1. JIRA Documentation . . . 10
1.1 JIRA 101 . . . 10
1.2 JIRA User's Guide . . . 16
1.2.1 JIRA Concepts . . . 18
1.2.1.1 What is a Project . . . 18
1.2.1.2 What is Workflow . . . 19
1.2.1.3 What is an Issue . . . 20
1.2.2 Getting Started . . . 22
1.2.2.1 Logging in to JIRA . . . 22
1.2.2.2 Exploring the JIRA Workspace . . . 22
1.2.2.3 Using Keyboard Shortcuts . . . 23
1.2.3 Working with Issues . . . 29
1.2.3.1 Attaching a File . . . 30
1.2.3.2 Attaching a Screenshot . . . 33
1.2.3.3 Cloning an Issue . . . 36
1.2.3.4 Commenting on an Issue . . . 37
1.2.3.5 Creating an Issue . . . 40
1.2.3.6 Creating a Sub-Task . . . 41
1.2.3.7 Editing an Issue . . . 43
1.2.3.8 Editing Rich-Text Fields . . . 44
1.2.3.9 Emailing an Issue . . . 46
1.2.3.10 Labelling an Issue . . . 48
1.2.3.11 Linking Issues . . . 50
1.2.3.12 Logging Work on an Issue . . . 54
1.2.3.13 Modifying Multiple ('Bulk') Issues . . . 62
1.2.3.14 Moving an Issue . . . 67
1.2.3.15 Scheduling an Issue . . . 69
1.2.3.16 Setting Security on an Issue . . . 70
1.2.3.17 Viewing an Issue's Change History . . . 71
1.2.3.18 Viewing an Issue's Crucible Reviews . . . 72
1.2.3.19 Viewing an Issue's FishEye Changesets . . . 73
1.2.3.20 Viewing the Bamboo Builds related to an Issue . . . 74
1.2.3.21 Watching and Voting on an Issue . . . 75
1.2.4 Searching for Issues . . . 76
1.2.4.1 Using Quick Search . . . 83
1.2.4.2 Performing Text Searches . . . 85
1.2.4.3 Advanced Searching . . . 89
1.2.4.4 Using the Issue Navigator . . . 155
1.2.4.5 Customising your Issue Navigator . . . 157

1.2.4.6 Saving Searches ('Issue Filters') . . . 158
1.2.4.7 Receiving Search Results as an RSS Feed . . . 164
1.2.4.8 Exporting Search Results to Microsoft Word . . . 166
1.2.4.9 Exporting Search Results to Microsoft Excel . . . 167
1.2.4.10 Displaying Search Results as a Chart . . . 169
1.2.4.11 Displaying Search Results in XML . . . 171
1.2.4.12 Receiving Search Results via Email . . . 176
1.2.4.13 Sharing a Search Result . . . 178
1.2.5 Generating Reports . . . 179
1.2.5.1 Workload Pie Chart Report . . . 180
1.2.5.2 User Workload Report . . . 182
1.2.5.3 Version Workload Report . . . 183
1.2.5.4 Time Tracking Report . . . 185
1.2.5.5 Single Level Group By Report . . . 187
1.2.5.6 Created vs Resolved Issues Report . . . 188
1.2.5.7 Resolution Time Report . . . 191
1.2.5.8 Pie Chart Report . . . 194
1.2.5.9 Average Age Report . . . 197
1.2.5.10 Recently Created Issues Report . . . 199
1.2.5.11 Time Since Issues Report . . . 202
1.2.6 Browsing a Project . . . 205
1.2.6.1 Browsing a Project's Summary . . . 206
1.2.6.2 Browsing a Project's Issues . . . 207
1.2.6.3 Browsing a Project's Road Map . . . 208
1.2.6.4 Browsing a Project's Change Log . . . 209
1.2.6.5 Browsing a Project's Popular Issues . . . 210
1.2.6.6 Browsing a Project's Versions . . . 211
1.2.6.6.1 Browsing a Version's Summary . . . 212
1.2.6.6.2 Browsing a Version's Issues . . . 213
1.2.6.6.3 Browsing a Version's Popular Issues . . . 214
1.2.6.6.4 Browsing a Version's Bamboo Builds . . . 214
1.2.6.7 Browsing a Project's Components . . . 216
1.2.6.7.1 Browsing a Component's Summary . . . 217
1.2.6.7.2 Browsing a Component's Issues . . . 218
1.2.6.7.3 Browsing a Component's Road Map . . . 219
1.2.6.7.4 Browsing a Component's Change Log . . . 220
1.2.6.7.5 Browsing a Component's Popular Issues . . . 220
1.2.6.8 Browsing a Project's Labels . . . 221
1.2.6.9 Browsing a Project's Bamboo Builds . . . 222
1.2.6.10 Browsing a Project's FishEye Changesets . . . 224
1.2.6.11 Browsing a Project's Crucible Reviews . . . 225
1.2.6.12 Viewing a Project's Burndown Chart . . . 226
1.2.7 Customising the Dashboard . . . 227
1.2.7.1 Managing Multiple Dashboard Pages . . . 231
1.2.7.2 Changing the Look and Behaviour of a Gadget . . . 237
1.2.7.3 Adding the Activity Stream Gadget . . . 241
1.2.7.4 Adding the Administration Gadget . . . 243
1.2.7.5 Adding the Assigned To Me Gadget . . . 244
1.2.7.6 Adding the Average Age Gadget . . . 245
1.2.7.7 Adding the Bamboo Charts Gadget . . . 247
1.2.7.8 Adding the Bamboo Plan Summary Chart Gadget . . . 248
1.2.7.9 Adding the Bamboo Plans Gadget . . . 251
1.2.7.10 Adding the Bugzilla ID Search Gadget . . . 252
1.2.7.11 Adding the Calendar Gadget . . . 253
1.2.7.12 Adding the Clover Coverage Gadget . . . 255
1.2.7.13 Adding the Created vs Resolved Gadget . . . 256
1.2.7.14 Adding the Crucible Charts Gadget . . . 258
1.2.7.15 Adding the Favourite Filters Gadget . . . 260
1.2.7.16 Adding the Filter Results Gadget . . . 261
1.2.7.17 Adding the FishEye Charts Gadget . . . 262
1.2.7.18 Adding the FishEye Recent Changesets Gadget . . . 263
1.2.7.19 Adding the In Progress Gadget . . . 264
1.2.7.20 Adding the Introduction Gadget . . . 265
1.2.7.21 Adding the Issue Statistics Gadget . . . 266
1.2.7.22 Adding the JIRA News Gadget . . . 267
1.2.7.23 Adding the Pie Chart Gadget . . . 267
1.2.7.24 Adding the Projects Gadget . . . 269
1.2.7.25 Adding the Quick Links Gadget . . . 271
1.2.7.26 Adding the Recently Created Issues Gadget . . . 271
1.2.7.27 Adding the Resolution Time Gadget . . . 274
1.2.7.28 Adding the Road Map Gadget . . . 276
1.2.7.29 Adding the Text Gadget . . . 277
1.2.7.30 Adding the Time Since Issues Gadget . . . 278
1.2.7.31 Adding the Two-Dimensional Filter Statistics Gadget . . . 280
1.2.7.32 Adding the Voted Issues Gadget . . . 282
1.2.7.33 Adding the Watched Issues Gadget . . . 283
1.2.7.34 Adding the Heat Map Gadget . . . 285
1.2.7.35 Adding the Labels Gadget . . . 285
1.2.8 Managing your User Profile . . . 286
1.2.8.1 Adding a User Avatar . . . 289

1.2.8.2 Allowing OAuth Access . . . 290
1.2.8.3 Changing your Password . . . 293
1.2.8.4 Choosing a Language . . . 294
1.2.8.5 Using Hover Profile . . . 295
1.2.8.6 Choosing a Time Zone . . . 296
1.3 JIRA Administrator's Guide . . . 297
1.3.1 Getting Help . . . 300
1.3.2 Configuring the Layout and Design . . . 301
1.3.2.1 Customising the Look and Feel . . . 302
1.3.2.2 Choosing a Default Language . . . 306
1.3.2.2.1 Translating JIRA . . . 307
1.3.2.3 Configuring the Default Issue Navigator . . . 307
1.3.2.4 Configuring the Default Dashboard . . . 308
1.3.2.4.1 Using Dashboard Gadgets . . . 309
1.3.2.4.2 Adding a Gadget to the Directory . . . 311
1.3.2.4.3 Subscribing to Another Application's Gadgets . . . 313
1.3.2.5 Configuring an Announcement Banner . . . 315
1.3.2.6 Enabling Logout Confirmation . . . 316
1.3.3 User and Group Management . . . 317
1.3.3.1 Managing Users . . . 317
1.3.3.2 Managing Groups . . . 322
1.3.3.3 Managing Project Roles . . . 325
1.3.3.4 Migrating User Groups to Project Roles . . . 328
1.3.3.5 Configuring User Directories . . . 334
1.3.3.5.1 Configuring the Internal Directory . . . 335
1.3.3.5.2 Connecting to an LDAP Directory . . . 336
1.3.3.5.3 Connecting to an Internal Directory with LDAP Authentication . . . 351
1.3.3.5.4 Connecting to Crowd or Another JIRA Server for User Management . . . 355
1.3.3.5.5 Managing Multiple Directories . . . 360
1.3.3.5.6 Synchronising Data from External Directories . . . 362
1.3.3.5.7 Managing Nested Groups . . . 363
1.3.3.5.8 Diagrams of Possible Configurations for User Management . . . 366
1.3.3.5.9 User Management Limitations and Recommendations . . . 372
1.3.3.5.10 Allowing Other Applications to Connect to JIRA for User Management . . . 375
1.3.3.6 Viewing User Sessions . . . 377
1.3.3.6.1 User access logging . . . 378
1.3.3.7 Clearing 'Remember my login' Tokens . . . 380
1.3.3.7.1 Disabling Remember My Login on this Computer . . . 382
1.3.3.8 Enabling Public Signup and CAPTCHA . . . 382
1.3.4 Project Management . . . 384
1.3.4.1 Defining a Project . . . 384
1.3.4.2 Managing Project Role Membership . . . 388
1.3.4.3 Defining a Component . . . 390
1.3.4.4 Managing Versions . . . 391
1.3.4.4.1 Running a Bamboo Build when Releasing a Version . . . 394
1.3.4.4.2 Creating Release Notes . . . 395
1.3.4.5 Configuring Project Keys . . . 396
1.3.5 Configuring Security . . . 397
1.3.5.1 Configuring Issue Level Security . . . 399
1.3.5.2 Managing Project Permissions . . . 402
1.3.5.3 Managing Global Permissions . . . 408
1.3.5.4 Configuring Secure Administrator Sessions . . . 411
1.3.5.5 Preventing Security Attacks . . . 413
1.3.5.6 JIRA Cookies . . . 415
1.3.6 Configuring Fields and Screens . . . 417
1.3.6.1 Configuring Built-in Fields . . . 419
1.3.6.1.1 Defining 'Issue Type' Field Values . . . 419
1.3.6.1.2 Defining 'Priority' Field Values . . . 425
1.3.6.1.3 Defining 'Resolution' Field Values . . . 427
1.3.6.1.4 Defining 'Status' Field Values . . . 427
1.3.6.1.5 Translating Resolutions, Priorities, Statuses and Issue Types . . . 429
1.3.6.2 Adding a Custom Field . . . 430
1.3.6.2.1 Configuring a Custom Field . . . 435
1.3.6.2.2 Creating Help for a Custom Field . . . 440
1.3.6.3 Specifying Field Behaviour . . . 441
1.3.6.3.1 Associating Field Behaviour with Issue Types . . . 447
1.3.6.3.2 Configuring Renderers . . . 450
1.3.6.4 Defining a Screen . . . 457
1.3.6.4.1 Associating a Screen with an Issue Operation . . . 463
1.3.6.4.2 Associating a Screen with an Issue Type . . . 467
1.3.7 Configuring Workflow . . . 471
1.3.7.1 Activating Workflow . . . 499
1.3.7.2 Adding a Custom Event . . . 501
1.3.7.3 Configuring the Initial Status . . . 504
1.3.7.4 Workflow Properties . . . 505
1.3.8 Configuring Email . . . 506
1.3.8.1 Configuring Email Notifications . . . 506
1.3.8.1.1 Configuring JIRA's SMTP Mail Server to Send Notifications . . . 507
1.3.8.1.2 Creating a Notification Scheme . . . 512
1.3.8.1.3 Customising Email Content . . . 517

1.3.8.2 Creating Issues and Comments from Email . . . 518
1.3.8.2.1 Configuring JIRA to Receive Email from a POP or IMAP Mail Server . . . 527
1.3.8.3 Using Gmail as a JIRA Mail Server . . . 529
1.3.9 Migrating from Other Issue Trackers . . . 531
1.3.9.1 Importing Data from Bugzilla . . . 532
1.3.9.2 Importing Data From A FogBugz Server . . . 535
1.3.9.3 Importing Data From FogBugz OnDemand . . . 538
1.3.9.4 Importing Data From Mantis . . . 541
1.3.9.5 Importing Data From Pivotal Tracker . . . 544
1.3.9.6 Importing Data From Trac . . . 547
1.3.9.7 Importing Data From CSV . . . 550
1.3.9.7.1 Commonly Asked CSV Questions and Known Issues . . . 556
1.3.9.7.2 How to Import CSV Data with PVCS Command . . . 557
1.3.10 Moving or Archiving Individual Projects . . . 558
1.3.10.1 Archiving a Project . . . 558
1.3.10.2 Splitting a JIRA instance . . . 559
1.3.11 Integrating with a Source Control System . . . 560
1.3.11.1 Integrating JIRA with FishEye . . . 560
1.3.11.2 Integrating JIRA with CVS and ViewCVS . . . 562
1.3.11.3 Integrating JIRA with Subversion . . . 566
1.3.11.4 Integrating JIRA with Perforce . . . 567
1.3.11.5 Integrating JIRA with ClearCase . . . 567
1.3.12 Integrating with a Build Management System . . . 567
1.3.12.1 Integrating JIRA with Bamboo . . . 567
1.3.13 Configuring Global Settings . . . 568
1.3.13.1 Configuring Time Tracking . . . 568
1.3.13.2 Configuring JIRA Options . . . 570
1.3.13.3 Setting Properties and Options on Startup . . . 576
1.3.13.3.1 Recognized System Properties for JIRA . . . 580
1.3.13.4 Advanced JIRA Configuration . . . 581
1.3.13.4.1 Changing the constraints on historical time parameters in gadgets . . . 583
1.3.13.4.2 Changing the Default Order for Comments from Ascending to Descending . . . 583
1.3.13.4.3 Limiting the number of issues returned from a search view such as an RSS feed . . . 583
1.3.13.5 Configuring File Attachments . . . 584
1.3.13.6 Configuring Application Links . . . 588
1.3.13.6.1 Adding an Application Link . . . 589
1.3.13.6.2 Configuring Authentication for an Application Link . . . 591
1.3.13.6.3 Editing an Application Link . . . 598
1.3.13.6.4 Making an Application Link the Primary Link . . . 599
1.3.13.6.5 Relocating an Application Link . . . 600
1.3.13.6.6 Upgrading an Application Link . . . 600
1.3.13.6.7 Deleting an Application Link . . . 603
1.3.13.6.8 Configuring Project Links across Applications . . . 603
1.3.13.7 Configuring Issue Cloning . . . 606
1.3.13.8 Configuring Issue Linking . . . 607
1.3.13.9 Configuring the Whitelist . . . 609
1.3.13.10 Configuring Sub-tasks . . . 609
1.3.13.11 Managing Shared Filters . . . 610
1.3.13.12 Managing Shared Dashboards . . . 612
1.3.14 Server Administration . . . 614
1.3.14.1 Increasing JIRA Memory . . . 614
1.3.14.2 Using the Database Integrity Checker . . . 617
1.3.14.3 Precompiling JSP pages . . . 618
1.3.14.4 Database Indexing . . . 619
1.3.14.5 Logging and Profiling . . . 620
1.3.14.5.1 Logging email protocol details . . . 623
1.3.14.6 Restoring Data . . . 624
1.3.14.6.1 Restoring a Project from Backup . . . 626
1.3.14.7 Optimising Performance . . . 634
1.3.14.8 Backing Up Data . . . 634
1.3.14.8.1 Automating JIRA Backups . . . 636
1.3.14.8.2 Preventing users from accessing JIRA during backups . . . 638
1.3.14.9 Search Indexing . . . 639
1.3.14.9.1 Re-Indexing after Major Configuration Changes . . . 639
1.3.14.10 Using robots.txt to hide from Search Engines . . . 640
1.3.14.11 Updating your JIRA License Details . . . 640
1.3.14.12 Viewing your System Information . . . 641
1.3.14.13 Generating a Thread Dump . . . 646
1.3.14.14 Performance Testing Scripts . . . 650
1.3.14.15 Finding the JIRA Support Entitlement Number (SEN) . . . 656
1.3.15 Appendix A - Extending JIRA . . . 658
1.3.15.1 Managing JIRA's Plugins . . . 659
1.3.15.2 Listeners . . . 665
1.3.15.3 Services . . . 667
1.3.15.4 Jelly Tags . . . 669
1.3.15.5 JIRA Toolkit (Customer Support Extensions) . . . 698
1.3.15.6 Developer Guides . . . 698
1.3.15.7 Building JIRA from Source . . . 698
1.3.15.7.1 How to Make a JIRA Patch . . . 701
1.3.15.8 API Documentation . . . 702

1.4 JIRA Installation and Upgrade Guide . . . 702
1.4.1 JIRA Requirements . . . 703
1.4.1.1 Installing Java . . . 705
1.4.2 Supported Platforms . . . 707
1.4.2.1 End of Support Announcements for JIRA . . . 708
1.4.2.2 Caveats in using Firefox 3.6.0 with JIRA . . . 712
1.4.3 Installing JIRA . . . 713
1.4.3.1 Installing JIRA on Windows . . . 713
1.4.3.1.1 Uninstalling JIRA from Windows . . . 715
1.4.3.2 Installing JIRA on Linux . . . 716
1.4.3.2.1 Uninstalling JIRA from Linux . . . 718
1.4.3.3 Installing JIRA from an Archive File on Windows, Linux or Solaris . . . 719
1.4.3.4 Installing JIRA WAR . . . 721
1.4.3.4.1 JIRA WAR Configuration Overview . . . 722
1.4.3.4.2 Installing JIRA on Tomcat 6.0 . . . 724
1.4.3.4.3 Installing JIRA on Tomcat 5.5 . . . 729
1.4.3.4.4 Switching Application Servers to Apache Tomcat . . . 735
1.4.3.4.5 Deploying Multiple Atlassian Applications in a Single Tomcat Container . . . 735
1.4.3.5 Configuring Your JIRA Installation . . . 735
1.4.3.5.1 Using the JIRA Configuration Tool . . . 735
1.4.3.5.2 Running JIRA as a Service . . . 739
1.4.3.5.3 Starting JIRA Automatically on Linux . . . 744
1.4.3.5.4 Starting JIRA automatically on FreeBSD . . . 746
1.4.4 Running the Setup Wizard . . . 747
1.4.4.1 Next Steps - Adding Users . . . 752
1.4.4.2 Next Steps - Creating a Project . . . 757
1.4.4.3 Next Steps - Creating an Issue . . . 761
1.4.5 Connecting JIRA to a Database . . . 762
1.4.5.1 Connecting JIRA to PostgreSQL . . . 763
1.4.5.2 Connecting JIRA to MySQL . . . 766
1.4.5.3 Connecting JIRA to Oracle . . . 769
1.4.5.4 Connecting JIRA to SQL Server 2005 . . . 772
1.4.5.5 Connecting JIRA to SQL Server 2008 . . . 776
1.4.5.6 Connecting JIRA to HSQLDB . . . 779
1.4.5.7 Switching Databases . . . 782
1.4.5.8 Surviving Connection Closures . . . 783
1.4.6 Upgrading JIRA . . . 784
1.4.6.1 Upgrading JIRA Manually . . . 788
1.4.6.2 Migrating JIRA to Another Server . . . 793
1.4.6.3 Disabling Auto-Export . . . 798
1.4.6.4 Rolling Back a JIRA Upgrade . . . 799
1.4.7 Important Directories and Files . . . 799
1.4.7.1 JIRA Installation Directory . . . 802
1.4.7.2 JIRA Home Directory . . . 803
1.4.7.3 Setting your JIRA Home Directory . . . 804
1.4.8 Tomcat security best practices . . . 806
1.4.9 Customising Your JIRA Installation . . . 809
1.4.9.1 Changing JIRA's TCP Ports . . . 809
1.4.9.2 Running JIRA in a Virtualised Environment . . . 810
1.4.9.3 Running JIRA over SSL or HTTPS . . . 814
1.4.9.4 Installing Confluence and JIRA Together . . . 820
1.4.9.5 Integrating JIRA with a Web Server . . . 820
1.4.9.5.1 Integrating JIRA with IIS . . . 820
1.4.9.5.2 Integrating JIRA with Apache . . . 825
1.4.9.6 Securing JIRA with Apache HTTP Server . . . 834
1.4.9.6.1 Using Apache to Limit Access to the JIRA Administration Interface . . . 834
1.4.9.6.2 Using Fail2Ban to limit login attempts . . . 846
1.4.10 Deployment Planning Activity . . . 847
1.5 JIRA Releases . . . 848
1.5.1 JIRA Release Summary . . . 848
1.5.2 Production Releases . . . 851
1.5.2.1 JIRA 5.0 Release Notes . . . 854
1.5.2.1.1 JIRA 5.0 Upgrade Notes . . . 864
1.5.2.2 JIRA 4.4 Release Notes . . . 867
1.5.2.2.1 JIRA 4.4 Upgrade Notes . . . 881
1.5.2.2.2 JIRA 4.4.5 Release Notes . . . 898
1.5.2.2.3 JIRA 4.4.4 Release Notes . . . 899
1.5.2.2.4 JIRA 4.4.3 Release Notes . . . 901
1.5.2.2.5 JIRA 4.4.2 Release Notes . . . 904
1.5.2.2.6 JIRA 4.4.1 Release Notes . . . 907
1.5.2.3 JIRA 4.3 Release Notes . . . 911
1.5.2.3.1 JIRA 4.3 Upgrade Guide . . . 924
1.5.2.3.2 JIRA 4.3.4 Release Notes . . . 935
1.5.2.3.3 JIRA 4.3.3 Release Notes . . . 937
1.5.2.3.4 JIRA 4.3.2 Release Notes . . . 939
1.5.2.3.5 JIRA 4.3.1 Release Notes . . . 939
1.5.2.4 JIRA 4.2 Release Notes . . . 941
1.5.2.4.1 JIRA 4.2 Upgrade Guide . . . 955
1.5.2.4.2 JIRA 4.2.4 Release Notes . . . 961
1.5.2.4.3 JIRA 4.2.3 Release Notes . . . 961

1.5.2.4.4 JIRA 4.2.2 Release Notes . . . 963
1.5.2.4.5 JIRA 4.2.1 Release Notes . . . 965
1.5.2.5 JIRA 4.1 Release Notes . . . 966
1.5.2.5.1 JIRA 4.1 Upgrade Guide . . . 978
1.5.2.5.2 JIRA 4.1.2 Release Notes . . . 982
1.5.2.5.3 JIRA 4.1.1 Release Notes . . . 985
1.5.2.6 JIRA 4.0 Release Notes . . . 988
1.5.2.6.1 JIRA 4.0 Upgrade Guide . . . 998
1.5.2.6.2 JIRA 4.0.2 Release Notes . . . 101
1.5.2.6.3 JIRA 4.0.1 Release Notes . . . 102
1.5.2.7 JIRA 3.13 Release Notes . . . 102
1.5.2.7.1 JIRA 3.13 Upgrade Guide . . . 102
1.5.2.7.2 JIRA 3.13.5 Release Notes . . . 103
1.5.2.7.3 JIRA 3.13.4 Release Notes . . . 103
1.5.2.7.4 JIRA 3.13.3 Release Notes . . . 103
1.5.2.7.5 JIRA 3.13.2 Release Notes . . . 103
1.5.2.7.6 JIRA 3.13.1 Release Notes . . . 103
1.5.2.8 JIRA 3.12 Release Notes . . . 103
1.5.2.8.1 JIRA 3.12 Upgrade Guide . . . 104
1.5.2.8.2 JIRA 3.12.3 Release Notes . . . 104
1.5.2.8.3 JIRA 3.12.2 Release Notes . . . 104
1.5.2.8.4 JIRA 3.12.1 Release Notes . . . 104
1.5.2.9 JIRA 3.11 Release Notes . . . 104
1.5.2.9.1 JIRA 3.11 Upgrade Guide . . . 105
1.5.2.10 JIRA 3.10 Release Notes . . . 105
1.5.2.10.1 JIRA 3.10 Upgrade Guide . . . 105
1.5.2.10.2 JIRA 3.10.2 Release Notes . . . 105
1.5.2.10.3 JIRA 3.10.1 Release Notes . . . 106
1.5.2.11 JIRA 3.9 Release Notes . . . 106
1.5.2.11.1 JIRA 3.9 Upgrade Guide . . . 106
1.5.2.11.2 JIRA 3.9.3 Release Notes . . . 106
1.5.2.11.3 JIRA 3.9.2 Release Notes . . . 106
1.5.2.11.4 JIRA 3.9.1 Release Notes . . . 106
1.5.2.12 JIRA 3.8 Release Notes . . . 106
1.5.2.12.1 Feedback for DHTML-loading of Issue screens . . . 106
1.5.2.12.2 JIRA 3.8 Upgrade Guide . . . 106
1.5.2.12.3 JIRA 3.8.1 Release Notes . . . 106
1.5.2.13 JIRA 3.6 Release Notes . . . 106
1.5.2.13.1 JIRA 3.6 Upgrade Guide . . . 107
1.5.2.13.2 JIRA 3.6.5 Release Notes . . . 108
1.5.2.13.3 JIRA 3.6.4 Release Notes . . . 108
1.5.2.13.4 JIRA 3.6.3 Release Notes . . . 108
1.5.2.13.5 JIRA 3.6.2 Release Notes . . . 108
1.5.2.13.6 JIRA 3.6.1 Release Notes . . . 108
1.5.2.14 JIRA 3.5 Release Notes . . . 109
1.5.2.14.1 JIRA 3.5 Upgrade Guide . . . 109
1.5.2.14.2 JIRA 3.5.3 Release Notes . . . 109
1.5.2.14.3 JIRA 3.5.2 Release Notes . . . 109
1.5.2.14.4 JIRA 3.5.1 Release Notes . . . 109
1.5.2.15 JIRA 3.4 and 3.4.1 Release Notes . . . 109
1.5.2.15.1 JIRA 3.4 and 3.4.1 Upgrade Guide . . . 110
1.5.2.15.2 JIRA 3.4.3 Release Notes . . . 110
1.5.2.15.3 JIRA 3.4.2 Release Notes . . . 111
1.5.2.15.4 JIRA 3.4.1 Release Notes . . . 111
1.5.2.16 JIRA 3.3 Release Notes . . . 111
1.5.2.16.1 JIRA 3.3 Upgrade Guide . . . 111
1.5.2.16.2 JIRA 3.3.3 Release Notes . . . 111
1.5.2.16.3 JIRA 3.3.2 Release Notes . . . 111
1.5.2.16.4 JIRA 3.3.1 Release Notes . . . 111
1.5.2.17 JIRA 3.2 Release Notes . . . 111
1.5.2.17.1 JIRA 3.2 Upgrade Guide . . . 112
1.5.2.17.2 3.2 performance benchmarks . . . 112
1.5.2.17.3 JIRA 3.2.3 Release Notes . . . 112
1.5.2.17.4 JIRA 3.2.2 Release Notes . . . 112
1.5.2.17.5 JIRA 3.2.1 Release Notes . . . 112
1.5.2.18 JIRA 3.0 Release Notes . . . 112
1.5.2.18.1 JIRA 3.0 Upgrade Notes . . . 113
1.5.2.18.2 JIRA 3.0.3 Release Notes . . . 113
1.5.2.18.3 JIRA 3.0.2 Release Notes . . . 113
1.5.2.18.4 JIRA 3.0.1 Release Notes . . . 113
1.5.2.19 All JIRA Release Notes (version 3.x and later) . . . 113
1.5.2.20 All JIRA Upgrade Guides (version 3.x and later) . . . 113
1.5.2.20.1 Aggregated JIRA 3.x Upgrade Guides . . . 113
1.5.2.21 JIRA 2.6 Release Notes . . . 115
1.5.2.22 JIRA 3.1 Release Notes . . . 115
1.5.2.22.1 JIRA 3.1 Upgrade Notes . . . 116
1.5.2.22.2 JIRA 3.1.1 Release Notes . . . 116
1.5.2.23 Important Version-Specific Upgrade Notes . . . 116
1.5.2.24 JIRA 3.7 Release Notes . . . 116
1.5.2.24.1 Issue Operations plugin . . . 116

1.5.2.24.2 JIRA 3.7 Upgrade Guide . . . 116
1.5.2.24.3 JIRA 3.7.4 Release Notes . . . 117
1.5.2.24.4 JIRA 3.7.3 Release Notes . . . 117
1.5.2.24.5 JIRA 3.7.2 Release Notes . . . 117
1.5.2.24.6 JIRA 3.7.1 Release Notes . . . 117
1.5.3 Beta Releases . . . 117
1.5.3.1 JIRA 5.0 RC 3 Release Notes . . . 117
1.5.3.2 JIRA 5.0 RC 2 Release Notes . . . 118
1.5.3.3 JIRA 5.0 RC 1 Release Notes . . . 119
1.5.3.4 JIRA 5.0 Beta 3 Release Notes . . . 120
1.5.3.5 JIRA 5.0 Beta 2 Release Notes . . . 121
1.5.3.6 JIRA 5.0 Beta 1 Release Notes . . . 122
1.5.3.7 JIRA 4.4 RC 1 Release Notes . . . 123
1.5.3.8 JIRA 4.4 Beta 1 Release Notes . . . 124
1.5.3.9 JIRA 4.3 RC1 Release Notes . . . 126
1.5.3.10 JIRA 4.3 Beta 1 Release Notes . . . 127
1.5.3.11 JIRA 4.2 RC1 Release Notes . . . 128
1.5.3.12 JIRA 4.2 Beta 3 Release Notes . . . 129
1.5.3.13 JIRA 4.2 Beta 2 Release Notes . . . 130
1.5.3.14 JIRA 4.2 Beta 1 Release Notes . . . 132
1.5.3.15 JIRA 4.0 RC1 Release Notes . . . 133
1.5.3.16 JIRA 4.0 Beta 5 Release Notes . . . 134
1.5.3.17 JIRA 4.0 Beta 4 Release Notes . . . 136
1.5.3.18 JIRA 4.0 Beta 3 Release Notes . . . 137
1.5.3.19 JIRA 4.0 Beta 2 Release Notes . . . 139
1.5.3.20 JIRA 4.0 Beta 1 Release Notes . . . 141
1.5.4 EAP Releases . . . 142
1.5.4.1 JIRA 5.0 EAP 5 Release Notes . . . 142
1.5.4.2 JIRA 5.0 EAP 4 Release Notes . . . 143
1.5.4.3 JIRA 5.0 EAP 3 Release Notes . . . 143
1.5.4.4 JIRA 5.0 EAP 2 Release Notes . . . 144
1.5.4.5 JIRA 4.4 EAP 6 Release Notes . . . 144
1.5.4.6 JIRA 4.4 EAP 5 Release Notes . . . 145
1.5.4.7 JIRA 4.4 EAP 4 Release Notes . . . 146
1.5.4.8 JIRA 4.4 EAP 3 Release Notes . . . 146
1.5.4.9 JIRA 4.4 EAP 2 Release Notes . . . 147
1.5.4.10 JIRA 4.3 EAP 5 Release Notes . . . 147
1.5.4.11 JIRA 4.3 EAP 3 Release Notes . . . 148
1.5.4.12 JIRA 4.3 EAP 2 Release Notes . . . 149
1.5.4.13 JIRA 4.3 EAP 1 Release Notes . . . 149
1.5.4.14 JIRA 4.2 EAP 4 Release Notes . . . 149
1.5.4.15 JIRA 4.2 EAP 3 Release Notes . . . 151
1.5.4.16 JIRA 4.2 EAP 2 Release Notes . . . 152
1.5.4.17 JIRA 4.2 EAP 1 Release Notes . . . 153
1.5.4.18 JIRA 3.11 EAP Release Notes . . . 154
1.5.5 Security Advisories . . . 154
1.5.5.1 JIRA Security Advisory 2007-12-24 . . . 154
1.5.5.2 JIRA Security Advisory 2008-02-21 . . . 155
1.5.5.3 JIRA Security Advisory 2008-08-26 . . . 155
1.5.5.4 JIRA Security Advisory 2008-10-29 . . . 155
1.5.5.5 JIRA Security Advisory 2008-12-09 . . . 155
1.5.5.6 JIRA Security Advisory 2009-04-02 . . . 155
1.5.5.7 JIRA Security Advisory 2010-04-16 . . . 156
1.5.5.7.1 Security Addendum 2010-04-16 - Determining if your public JIRA instance has been compromised . . . 156
1.5.5.7.2 Security Addendum 2010-04-16 - Preventing security attacks . . . 157
1.5.5.8 JIRA Security Advisory 2010-06-18 . . . 157
1.5.5.9 JIRA Security Advisory 2010-12-06 . . . 157
1.5.5.10 JIRA Security Advisory 2011-02-21 . . . 157
1.5.5.11 JIRA Security Advisory 2011-09-27 . . . 157
1.6 JIRA Resources . . . 158
1.6.1 Support Policies . . . 158
1.6.1.1 Bug Fixing Policy . . . 158
1.6.1.2 How to Report a Security Issue . . . 158
1.6.1.3 New Features Policy . . . 158
1.6.1.4 Patch Policy . . . 158
1.6.1.5 Security Advisory Publishing Policy . . . 158
1.6.1.6 Security Patch Policy . . . 158
1.6.1.7 Severity Levels for Security Issues . . . 158
1.6.2 Local JIRA documentation . . . 158
1.6.3 JIRA FAQ . . . 158
1.6.4 How JIRA Documentation Updates are Published . . . 158
1.7 JIRA Administrators FAQ . . . 159
1.7.1 Performance FAQ . . . 159
1.7.1.1 Is your JIRA Running Slowly . . . 159
1.7.2 Usage FAQ . . . 159
1.7.2.1 Modifying the JIRA Footer . . . 159
1.7.2.2 'Road Map', 'Change Log ', and 'Versions' Project Tabs Are Not Visible . . . 159
1.7.2.3 Why Do Linked Issues in JIRA Appear with a Strike-Through . . . 159
1.7.2.4 How to Enable the FishEye Plugin from the Plugin Administration Screen . . . 159
1.7.2.5 How to Add the Priority Field Into the Email Subject . . . 159

1.7.2.6 How to Change the Number of Rows Allowed in the Text Type Custom Field Renderer . . . 159
1.7.2.7 How to Make a Federated JIRA Instance . . . 159
1.7.2.8 How to Remove 'Unknown' Option from 'Component' and 'Fix Versions' . . . 159
1.7.2.9 Automatic Escalation of issues . . . 159
1.7.2.10 How to Activate Header Row for Subtask List in Issue Detail View . . . 159
1.7.2.11 How to Limit the Number of Characters Entered in a Summary Field . . . 159
1.7.2.12 JIRA's Timestamp Doesn't Match the System Time . . . 159
1.7.2.13 How to Attach a File During Issue Creation . . . 159
1.7.2.14 How to Remove Duplicate Entries for Names and Groups . . . 159
1.7.2.15 How to Export Users to CSV from JIRA . . . 159
1.7.2.16 How Come JIRA does Not Show Direct SQL Data Modifications . . . 159
1.7.2.17 How to get JIRA Pages to Render when URL Contains an Underscore . . . 159
1.7.2.18 Certain Action Words Show Incorrect Language Translations . . . 159
1.7.2.19 How to disable or enable the GOT FEEDBACK button . . . 160
1.7.2.20 Changing the number of users synchronised from LDAP to JIRA . . . 160
1.7.2.21 How do I reduce my user count in JIRA . . . 160
1.7.2.22 Adding custom content to the front page . . . 160
1.7.2.23 Allow editing of Closed Issues . . . 160
1.7.2.24 Allowing users to create issues anonymously . . . 160
1.7.2.25 Anonymising JIRA Data . . . 160
1.7.2.26 Appending Email Addresses to Comments Made by Anonymous Users when Using a Mail Handler . . . 160
1.7.2.27 Asking for an attachment on the Create Issue page . . . 160
1.7.2.28 Automatically Populate Timezone from a Created Issue . . . 160
1.7.2.29 Can I store customer details, like company, address and contact information, in JIRA? . . . 160
1.7.2.30 Change JIRA Browser Icon . . . 160
1.7.2.31 Changing Custom Field Types . . . 160
1.7.2.32 Changing Templates Used by Export to Excel from the Issue Navigator . . . 160
1.7.2.33 Changing the default attachment size limit . . . 160
1.7.2.34 Changing the default session timeout . . . 160
1.7.2.35 Changing the Default Tab Panel from Comments to All . . . 160
1.7.2.36 Changing the Due Date Input Format . . . 160
1.7.2.37 Changing the Project Key . . . 160
1.7.2.38 Changing the Project Key Format . . . 160
1.7.2.39 Changing the Size and Content of the Components Select List . . . 160
1.7.2.40 Changing the Size of the Fix Versions and Affects Versions Select List . . . 160
1.7.2.41 Changing the Size of the Text Area Custom Field . . . 161
1.7.2.42 Changing the Temporary Directory . . . 161
1.7.2.43 Changing Usernames in JIRA . . . 161
1.7.2.44 Configuring project specific security . . . 161
1.7.2.44.1 Controlling project visibility . . . 161
1.7.2.44.2 Using Project Level Security with Project Roles . . . 161
1.7.2.44.3 Using Project Level Security with User Groups . . . 161
1.7.2.45 Connecting to SSL services . . . 161
1.7.2.46 Creating a Custom Workflow . . . 162
1.7.2.47 Creating an Unassigned Issue . . . 162
1.7.2.48 Creating Issues via direct HTML links . . . 162
1.7.2.49 Current Reporter Browse Project Permission . . . 162
1.7.2.50 CVS ssh Jira Integration . . . 162
1.7.2.51 Disabling Form Token Checking . . . 163
1.7.2.52 Displaying a Field Based on Another Field Selection . . . 163
1.7.2.53 Editing a custom field option . . . 163
1.7.2.54 Escalating issues (or sending email notifications) when the set turnaround time is exceeded . . . 163
1.7.2.55 Field Layout Schemes in JIRA 3.x . . . 163
1.7.2.56 Fields Allowing Custom HTML or JavaScript . . . 163
1.7.2.57 Finding the Id for Issue Types . . . 163
1.7.2.58 Group Name Guidelines for JIRA . . . 163
1.7.2.59 How can I control the editing of issue fields via workflow? . . . 163
1.7.2.59.1 Using a Workflow to control edit of an issue by changing Workflow XML . . . 163
1.7.2.59.2 Using Templates to control edit of an issue . . . 163
1.7.2.60 How do I assign issues to multiple users . . . 163
1.7.2.61 How do I delete a user account? . . . 163
1.7.2.62 How do I disable Firebug for JIRA? . . . 163
1.7.2.63 How Do I Use an SSL Certificate Generated Using openssl? . . . 163
1.7.2.64 How the CreateOrCommentHandler works? . . . 163
1.7.2.65 How to change Multi Select Custom field size using script . . . 164
1.7.2.66 How to change the location of stdout and stderr logs . . . 164
1.7.2.67 How to clear the resolution field when the issue is reopened . . . 164
1.7.2.68 How to configure sub-task to have a specific screen? . . . 164
1.7.2.69 How to convert types using Jelly . . . 164
1.7.2.70 How to create a download link to a file . . . 164
1.7.2.71 How to disable the Resolve issue screen while resolving issues . . . 164
1.7.2.72 How to display a different format for the Number customfield . . . 164
1.7.2.73 How to ensure the Road Map tab is visible . . . 164
1.7.2.74 How to Get Unicode 'non-ASCII' Characters in HTTPS URL to Appear Correctly . . . 164
1.7.2.75 How to have long component version names display properly in the Issue Navigator . . . 164
1.7.2.76 How to Remove 'NONE' from a Select List Custom Field . . . 164
1.7.2.77 How to Rename the 'Priority' Field in the Issue Navigator . . . 164
1.7.2.78 How to re-order statuses . . . 164
1.7.2.79 How to re-order the list of issue operation in an issue . . . 164
1.7.2.80 How to resize Free Text Field customfield . . . 164

1.7.2.81 How to resize the 'Components' and 'Affects Versions' fields in the Issue Navigator . . . 164
1.7.2.82 How to Restrict the Subversion Commits Tab to Selected Projects or Users . . . 164
1.7.2.83 How to search by number range in the Issue Navigator . . . 164
1.7.2.84 How to show a transition only when the Assignee is different from the Current User . . . 164
1.7.2.85 Importing data . . . 164
1.7.2.86 Importing user from LDAP . . . 164
1.7.2.86.1 Neat JIRA LDAP tricks . . . 164
1.7.2.87 JIRA as a Support System . . . 164
1.7.2.87.1 Jelly Escalation . . . 165
1.7.2.87.2 Simple Escalation . . . 165
1.7.2.88 Letting customers only create issues . . . 165
1.7.2.89 Linking to local file under Firefox . . . 165
1.7.2.90 Login problems . . . 165
1.7.2.91 Mail error - Unable to relay . . . 165
1.7.2.92 Making JIRA login case insensitive for JIRA 3.13.x . . . 165
1.7.2.93 Outward Link Description and Inward Link Description . . . 165
1.7.2.94 Parsing utf-7 emails . . . 165
1.7.2.95 Project-specific email templates . . . 165
1.7.2.96 QuickSearch guesses the issue key prefix (sometimes) . . . 165
1.7.2.97 Receiving a Daily Summary of Updated Issues . . . 165
1.7.2.98 Recieving Notification for Select Issues or Updates . . . 165
1.7.2.99 Removing Commas for Values Held in Number Field Custom Field Type . . . 165
1.7.2.100 Removing invalid characters from XML backups . . . 166
1.7.2.101 Removing NONE from the Issue Security Drop-Down List . . . 166
1.7.2.102 Re-order workflow transactions . . . 166
1.7.2.103 Resolved issues appearing in Open issues filters . . . 166
1.7.2.104 Restricting the Visibility of Worklog on an Issue . . . 166
1.7.2.105 Retrieving the JIRA Administrator . . . 166
1.7.2.106 Scheme Entity Relations Map . . . 166
1.7.2.107 Sending JIRA Data to Support . . . 166
1.7.2.108 Setting Additional Fields for Issues Created from Email . . . 166
1.7.2.109 Setting a Default Value in the Description Field . . . 166
1.7.2.110 Setting Priority field value based on customfield value . . . 166
1.7.2.111 Showing Extended Timestamp in the Created Column of the Issue Navigator . . . 166
1.7.2.112 Single Sign-on . . . 166
1.7.2.113 Tracking the Time Taken for Each Workflow Transition . . . 166
1.7.2.114 Troubleshooting Issue Creation Via Email . . . 166
1.7.2.115 Using JIRA to Manage reusable modules . . . 167
1.7.2.116 Using validators to make custom fields required on transition screens . . . 167
1.7.2.117 We already have users & groups defined elsewhere - can JIRA make use of these? . . . 167
1.7.2.118 Where are the application server logs? . . . 167
1.7.2.119 Why doesn't JIRA have a Severity field like Bugzilla? . . . 167
1.7.2.120 Workflows Guidebook . . . 167
1.7.2.121 XML format for import & export files . . . 167
1.7.3 Installation Notes . . . 167
1.7.3.1 Configuring IIS with Tomcat . . . 167
1.7.3.2 Database Notes . . . 167
1.7.3.2.1 Incorrect database type specified . . . 167
1.7.3.2.2 Restarting JIRA from the Setup Wizard . . . 167
1.7.3.2.3 Database limitations on number of projects . . . 167
1.7.3.2.4 JIRA and HSQL . . . 167
1.7.3.2.5 JIRA and MS SQL Server 2005 . . . 167
1.7.3.2.6 JIRA and MS SQL Server 2008 . . . 168
1.7.3.2.7 JIRA and MySQL . . . 168
1.7.3.2.8 JIRA and Oracle . . . 169
1.7.3.2.9 JIRA and PostgreSQL . . . 169
1.7.3.3 How to Set Up SMTP Relay in Exchange 2007 . . . 169
1.7.3.4 Installation Troubleshooting Guide . . . 169
1.7.3.5 Installing a LDAP server on Debian Linux for use with JIRA . . . 169
1.7.3.6 Installing Java on Ubuntu or Debian . . . 170
1.7.3.7 Installing JIRA on Mac OS X . . . 170
1.7.3.7.1 Configure JIRA as service on Mac OS X . . . 170
1.7.3.8 Is Clustering or Load Balancing JIRA Possible . . . 170
1.7.3.9 java.lang.NoClassDefFoundError . . . 170
1.7.3.10 JVM and Appserver configuration info . . . 170
1.7.3.11 LicenseFactory error after upgrading JIRA . . . 170
1.7.3.12 Logging request headers . . . 170
1.7.3.13 Running multiple instances of JIRA on one machine . . . 170
1.7.3.14 Solaris ClassNotFoundException . . . 170
1.7.3.15 Windows cannot find -Xms128m . . . 171
1.8 Contributing to the JIRA Documentation . . . 171
1.8.1 Tips of the Trade . . . 171
1.8.2 Tips via Twitter . . . 171
1.8.3 JIRA Documentation in Other Languages . . . 171
1.9 GreenHopper for JIRA Guide . . . 171

JIRA Documentation

JIRA 5.0.x

User's Guide

The JIRA User's Guide is for project managers, developers, testers – anyone who uses JIRA. New to JIRA? Start by exploring the JIRA workspace and learning about issues, projects and workflow. Try creating some issues, then search, browse, export or report on them. Want to build up your skills from white belt (beginner) to JIRA master? Try our JIRA ninja tutorial.

Administrator's Guide

The JIRA Administrator's Guide is for people with JIRA administration rights. It will help you set up users and groups, projects, security and email. You may want to customise the look and feel of JIRA, and add your own fields, screens and workflows. Admin tasks such as backup are also covered. You may also find the Knowledge Base, FAQ and JIRA Forum useful.

Installation Guide

The JIRA Installation Guide is for people who are installing JIRA for the first time. Check the requirements and supported platforms, then download and install JIRA. Where to next? The JIRA 101 will help you get started, or you may like to import the JIRA Sample Files. If you are using other Atlassian products, take a look at the Integration Guide.

Upgrade Guide

The JIRA Upgrade Guide is for people who are upgrading their instance of JIRA. Start by reading the latest Release Notes and version-specific Upgrade Guide for the version to which you are upgrading, then download JIRA and follow the main Upgrade Guide.

Developer Resources

These resources are for software developers who want to create their own plugins for JIRA. Take a look at the JIRA Developer Documentation and the API Documentation. You may also find the JIRA Developers Forum useful.

JIRA 101

JIRA 101

Thank you for choosing JIRA to track your projects and issues. To help you get up and running quickly, we've compiled some easy instructions for configuring and using JIRA 4.4.

Getting Started

1. Installing JIRA

First things first. If you haven't already got JIRA up and running, carry out the following steps:

For Windows: (click to expand)

You may want to watch the video showing how to do this.

1. Download the JIRA Standalone Windows Installer (.EXE) file from the Atlassian Download Center.
2. Run the .EXE file, choose an installation directory, a home directory and a port ('8080' will do). We recommend that you choose to 'Run JIRA as a service'.
3. JIRA will start automatically when the Installer finishes, if you selected the option to launch JIRA at the end of the Installer.
4. To access JIRA, go to your web browser and type this address: http://localhost:8080. Windows 'Start' menu shortcuts will also be added which you can also use to start and stop JIRA.
5. Follow the Setup Wizard. This will guide you through the process of setting up your JIRA server, creating an Admin user and (optionally) setting up email.

For more help on the technical procedures in this section, see the JIRA Installation Guide.

If you need assistance, please create a support ticket.

Before using JIRA as a production system, you need to switch from the default HSQL database, which is provided for evaluation purposes only. Please see the documentation for details.

For Mac: (click to expand)

You may want to watch the video showing how to do this.

1. Download the JIRA Standalone TAR (.GZ) file from the Atlassian Download Center, and unzip it.
2. Edit the `jira-application.properties` file in the JIRA Installation Directory, add a 'jira.home' property and set it to your desired location for the JIRA home directory. Please use forward-slashes ("/"), not back-slashes ("\").
3. Run `bin/startup.sh` to start JIRA.
4. To access JIRA, go to your web browser and type this address: http://localhost:8080.
5. Follow the Setup Wizard. This will guide you through the process of setting up your JIRA server, creating an Admin user and (optionally) setting up email.

For more help on the technical procedures in this section, see the JIRA Installation Guide.

If you need assistance, please create a support ticket.

Before using JIRA as a production system, you need to switch from the default HSQL database, which is provided for evaluation purposes only. Please see the documentation for details.

For Linux: (click to expand)

1. Download the JIRA Standalone TAR (.GZ) file from the Atlassian Download Center, and unzip it.
2. Install Java and set JAVA_HOME.
3. Edit the `jira-application.properties` file in the JIRA Installation Directory, add a 'jira.home' property and set it to your desired location for the JIRA home directory. Please use forward-slashes ("/"), not back-slashes ("\").
4. Run `bin/startup.sh` to start JIRA.
5. To access JIRA, go to your web browser and type this address: http://localhost:8080.
6. Follow the Setup Wizard. This will guide you through the process of setting up your JIRA server, creating an Admin user and (optionally) setting up email.

For more help on the technical procedures in this section, see the JIRA Installation Guide.

If you need assistance, please create a support ticket.

Before using JIRA as a production system, you need to switch from the default HSQL database, which is provided for evaluation purposes only. Please see the documentation for details.

2. Adding Users

For each of your users, you will need to do the following: (click to expand)

1. Click '**Administration**' in the top navigation bar.
2. Select **'Users' > 'Users'** from the top menu, then click '**Add User**'.
3. Enter the **Username**, **Password**, **Full Name** and **Email Address**; and (optionally) tick the box to send the user an email containing their account details. Then click the **Create** button. For more details, please see the documentation.
4. The User Browser will be displayed. Locate the new user and click the '**Groups**' link in the '**Operations**' column.
5. If the user is going to need to work on issues, select the '**jira-developers**' group and click the '**Join**' button. (If the user is only going to log issues, and not work on them, then they don't need to belong to the '**jira-developers**' group.) For more about groups, please see the documentation.
    - Note: Adding your users to the '**jira-developers**' group will automatically add them to the Default Members for the '**Developers**' project role. For more about project roles, please see the documentation.

You may want to suggest to your users that they take a look at 'Mastering the Basics' (below). You may also want to point them to the documentation on:

- Changing your Password
- Exploring the JIRA Workspace

 Your users will need to access JIRA at **http://<JIRA machine name>:8080** (not http://localhost:8080).

3. Creating a Project

Before you can create issues, you need to create a project to contain them. Here's how: (click to expand)

1. Click '**Administration**' in the top navigation bar.
2. Select **'Projects' > 'Projects'** from the top menu, then click '**Add Project**'.
3. In the '**Name**' field, type a descriptive name for your project (typically two or three words, e.g. '**Purchase Orders**').
4. In the '**Key**' field, type a meaningful prefix for issues in your project (typically three or four characters, e.g.'**ORD**'). Note that this cannot be changed later.
5. In the '**Project Lead**' field, select the user to whom issues should be assigned by default.
6. If you chose to set up email when you installed JIRA (see above), change the '**Notification Scheme**' field from '**None**' to '**Default Notification Scheme**'. This will allow JIRA to automatically send emails to appropriate people when particular events occur (e.g. 'Issue Created', 'Issue Resolved'). For more about email, please see the documentation.
7. Leave the rest of the fields with their default values for now. Click the '**Add**' button.

Mastering the Basics

4. Creating an Issue

▼ To enter a new issue into JIRA: (click to expand)

1. Click the '**Create Issue**' link in the top navigation-bar.
2. Select the relevant Project and Issue Type, then click the '**Next**' button.
3. Type a short description of the issue in the '**Summary**' field, then click the '**Create**' button.

For more details, please see the documentation.

5. Logging Work on an Issue

▼ To record the time that you have spent working on an issue, and action you have taken: (click to expand)

1. Go to the issue, and select '**Log work**' from the '**Operations**' menu in the left column.
2. In the '**Time Spent**' field, enter the amount of time to be logged. Use 'w', 'd', 'h' and 'm' to specify weeks, days, hours or minutes (e.g. to enter two hours of work, type '2h').
3. In the '**Work Description**' field, type a description or comment about the work you have done
4. Click the '**Log**' button.

For more details about the other options on this screen, please see the documentation.

6. Resolving an Issue

▼ Resolving an issue sets its Status to 'Resolved', indicating that work is complete. To resolve an issue: (click to expand)

1. Go to the issue, and select '**Resolve Issue**' from the '**Available Workflow Actions**' menu in the left column.
2. In the '**Resolution**' field, select the 'Resolution' that best describes the outcome (e.g. 'Fixed').
3. (Optional) In the '**Comment**' field, type a description or comment about the issue's resolution.
4. Click the '**Resolve**' button.

For more information about how an issue moves from one Status to another, please see the documentation.

7. Searching for Issues

▼ To use Quick Search: (click to expand)

- Use the box in the top right corner of every page to quickly search JIRA. You can type an issue key (e.g. `TEST-1234`) to jump directly to an issue, or use syntax like `my open issues` to immediately return all issues which are assigned to you and haven't yet been resolved.

▼ To use Regular Search: (click to expand)

1. Click '**Issues**' on the top navigation bar to display JIRA's searching and filtering panel.
2. Select the Project, Issue Type, or any other issue attributes of interest. You can also perform comprehensive text searches.

▼ To use Advanced Search: (click to expand)

1. Click '**Issues**' on the top navigation bar, then click '**advanced**' to display the JQL (JIRA Query Language) panel.
2. Type your query (e.g. 'project=TEST') and click the '**Search**' button.

See the documentation for more information about saving your searches ('issue filters') and receiving search results via email.

8. Charting, Reporting and Exporting

▼ To generate a chart: (click to expand)

1. View your search results (see *'Searching'* above) in the Issue Navigator.
2. Click the '**Views**' menu and select the '**Charts**' option.
3. Choose your preferred type of chart, and enter any required configuration details. For more details, please see the documentation.

▼ To generate a report: (click to expand)

1. Click '**Projects**' in the top navigation bar.
2. Select the project you are interested in.
3. Click '**Reports**' at the right of the screen and select the report of interest.
4. Enter any required configuration details, then click '**Next**' to display your report (e.g. Workload Pie Chart Report).

▼ To export data to MS-Word, MS-Excel, XML or RSS: (click to expand)

1. View your search results (see *'Searching'* above) in the Issue Navigator.
2. Click the '**Views**' menu and select '**Word**', '**Excel**', or your preferred format. For more details, please see the documentation.

Customising JIRA

(Note that you need to be an Administrator to do the tasks in this section.)

Before you begin: (click to expand)

You may want to create a sample project named '**Purchase Orders**', in which to perform the tasks described in this section. For instructions, please see *'Creating a Project'* (above).

9. Adding a new Issue Type

Why would I do this? (click to expand)

The Issue Type is one of the first things a user must choose when they create an issue.
Depending on how your organisation is using JIRA, you might want to add a new Issue Type.
For example, if you are using JIRA to track purchase orders, the default Issue Types ('Bug', 'Improvement', 'New Feature', 'Task') might not be relevant. So you might want to add a new Issue Type called '**Order**'.

How do I do this? (click to expand)

To add a new Issue Type called '**Order**', and associate it with a project called '**Purchase Orders**':

1. Click **'Administration'** in the top navigation bar.
2. Select **'Issues' > 'Issue Types'**.
3. In the '**Add New Issue Type**' form, in the '**Name**' field, type '**Order**. In the '**Description**' field, type '**A purchase order**'. Then click the '**Add**' button'. (For more about adding Issue Types, and icons, please see the documentation.)
4. Click the '**Issue Types Scheme**' tab at the top of the '**Manage Issue Types**' screen.
5. In the '**Add New Issue Type Scheme**' form, in the '**Name**' field, type '**Purchase Order Issue Type Scheme**'. Then click the '**Add**' button'. (For more about Issue Type Schemes, please see the documentation.)
6. In the '**Available Issue Types**' list, click the Issue Type called '**Order**' and drag it into the '**Issue Types for Current Scheme**' list. Then click the '**Save**' button.
7. Click '**Projects**' in the left navigation column. Then in the '**Name**' column, click '**Purchase Orders**'. The project details will be displayed.
8. Click the '**Select**' link next to the '**Issue Type Scheme**' field, select '**Purchase Order Issue Type Scheme**' and click the '**Associate**' button.

To test what you have done, create an issue in the '**Purchase Orders**' project. The only available Issue Type should be '**Order**'.

10. Adding a new Screen

Why would I do this? (click to expand)

Depending on how your organisation is using JIRA, you might want to add a purpose-built screen that will be displayed for particular types of issues, or for particular projects or workflows.
For example, if you are using JIRA to track purchase orders, some of the normal issue fields (e.g. 'Affects Version', 'Fix Version', 'Environment') might not be relevant. So you might want to create a simplified screen that omits these fields.

How do I do this? (click to expand)

To add a new Screen called '**Purchase Order Screen**', and associate it with a project called '**Purchase Orders**':

1. Click '**Administration**' in the top navigation bar.
2. Select **'Issues' > 'Screens'**.
3. Click the '**Copy**' link next to '**Default Screen**'.
4. In the '**Name**' field, type '**Purchase Order Screen**'. Then click the '**Copy**' button'. (For more about Screens, please see the documentation.)
5. Click the '**Configure**' link next to '**Purchase Order Screen**'.
6. Tick the '**Remove**' box for the following fields: '**Affects Version**', '**Fix Version**', '**Environment**'. Then click the '**Remove**' button.
7. Click '**Screen Schemes**' in the left navigation column (under '**Issue Fields**').
8. In the '**Name**' field, type '**Purchase Order Screen Scheme**'. In the '**Default Screen**' field, select '**Purchase Order Screen**'. Then click the '**Add**' button'. (For more about Screen Schemes, please see the documentation.)
9. Click '**Issue Type Screen Schemes**' in the left navigation column (under '**Issue Fields**').
10. In the '**Name**' field, type '**Purchase Order Issue Type Screen Scheme**'. In the '**Screen Scheme**' field, select '**Purchase Order Screen Scheme**'. Then click the '**Add**' button'. (For more about Issue Type Screen Schemes, please see the documentation.)
11. Click '**Projects**' in the left navigation column. Then in the '**Name**' column, click '**Purchase Orders**'. The project details will be displayed.
12. Click the '**Select**' link next to the '**Issue Type Screen Scheme**' field, select '**Purchase Order Issue Type Screen Scheme**' and click the '**Associate**' button.

To test what you have done, view an issue in the '**Purchase Orders**' project. You shouldn't see the '**Affects Version**', '**Fix Version**' or '**Environment**' fields.

11. Adding a new Custom Field

Why would I do this? (click to expand)

Depending on how your organisation is using JIRA, you might need to add a 'custom' field that will be displayed for particular types of issues, or for particular projects.
For example, if you are using JIRA to track purchase orders, you might create a custom field called '**Supplier**'.

▼ How do I do this? (click to expand)

To create a new custom field called '**Supplier**' and put it on the '**Purchase Order Screen**':

1. Click **'Administration'** in the top navigation bar.
2. Select **'Issues' > 'Fields' > 'Custom Fields'**.
3. Click '**Add Custom Field**'.
4. On the '**Create Custom Field - Step 1**' screen, in the '**Field Type**' field, choose '**Select List'**. Then click the '**Next**' button.
5. On the '**Create Custom Field - Step 2**' screen:
    a. in the '**Name**' field, type '**Supplier'**.
    b. in the '**Description**' field, type '**Choose the supplier for this Purchase Order**'.
    c. under '**Choose applicable issue types**' select '**Order**'. Then click the '**Finish**' button.
6. On the '**Associate field Order to screens**' screen, tick the check-box for '**Purchase Order Screen**'. Then click the '**Update**' button.
7. On the '**View Custom Fields**' screen, click the '**Configure**' link next to '**Supplier**'. Then click '**Edit Options**'.
8. Add three options: '**ABC Pty Ltd**', '**ACME Pty Ltd**', '**XYZ Pty Ltd**'. Choose '**ACME Pty Ltd**' as the default. Then click the '**Done**' button.

To test what you have done, create an issue in the '**Purchase Orders**' project. You should see a field called '**Supplier**' that has a drop-down box containing your three options.

12. Adding a new Issue Status and Workflow

▼ Why would I do this? (click to expand)

Depending on how your organisation is using JIRA, you might need to add a new 'Status' , i.e. a new step in the issue ' workflow'.
For example, if you are using JIRA to track purchase orders, you might add a new Status called '**Purchase Approved**' .

▼ How do I do this? (click to expand)

To add a new Status called '**Purchase Approved**', and create a new workflow that has an extra step between '**Open**' and '**In Progress**':

1. Click '**Administration**' in the top navigation bar.
2. Click '**Statuses**' in the left navigation column (under '**Issue Settings**').
3. In the '**Name**' field, type '**Purchase Approved**'. Then click the '**Add**' button'. (For more about adding Statuses, and icons, please see the documentation.)
4. Click '**Workflows**' in the left navigation column.
5. Click the '**Copy**' link next to '**jira (Read-only System Workflow)**'.
6. In the '**Workflow Name**' field, type '**Purchase Order Workflow**'. Then click the '**Copy**' button'. (For more about Workflow, please see the documentation.)
7. Click the '**Steps**' link next to '**Purchase Order Workflow**'.
8. In the '**Add New Step**' form:
    a. in the '**Name**' field, type '**Purchase Approved**'.
    b. in the '**Linked Status**' field, select '**Purchase Approved**'.
    c. click the '**Add**' button.
9. Click the '**Add Transition**' link next to '**Open**':
    a. in the '**Name**' field, type '**Approve Purchase**'.
    b. in the '**Destination Step**' field, select '**Purchase Approved**'.
    c. click the '**Add**' button.
10. Click the '**Add Transition**' link next to '**Purchase Approved**':
    a. in the '**Name**' field, type '**Start Progress**'.
    b. in the '**Destination Step**' field, select '**In Progress**'.
    c. click the '**Add**' button.
11. Click the '**Delete Transitions**' link next to '**Open**'. Select '**Start Progress**' and click the '**Delete**' button.
12. Click '**Workflow Schemes**' in the left navigation column (under '**Schemes**'). Then click '**Add Workflow Scheme**'.
13. In the '**Name**' field, type '**Purchase Order Workflow Scheme**'. In the '**Default Screen**' field, select '**Purchase Order Screen**'. Then click the '**Add**' button'. (For more about Workflow Schemes, please see the documentation.)
14. Click the '**Workflows**' link next to '**Purchase Order Workflow Scheme**', click '**Assign Workflow**' and select '**Purchase Order Workflow**'. Then click the '**Add**' button.
15. Click '**Projects**' in the left navigation column. Then in the '**Name**' column, click '**Purchase Orders**'. The project details will be displayed.
16. Click the '**Select**' link next to the '**Workflow Scheme**' field, select '**Purchase Order Workflow Scheme**' and click the '**Associate**' button.

To test what you have done, create an issue in the '**Purchase Orders**' project. After you save the issue, the left column should contain a link called '**Approve Purchase**', but not a link called '**Start Progress**'.

Note: you may also want to watch the video on customising JIRA Workflow.

13. Using Permission Schemes, Groups and Project Roles

Why would I do this? (click to expand)

A Permission scheme allows you to grant people 'permission' to work on issues in a project. The new project that you created previously is using JIRA's **Default Permission Scheme**. If you end up creating lots of projects, you might need to grant different people permission to work on different projects.
For example, if your organisation requires all software development issues to be tested by a Quality Assurance person before being closed, you could create a permission scheme called '**Software Development Permission Scheme**' in which you assign the 'Close Issue' permission to the appropriate people. You would then associate your new permission scheme with all your software development projects.
There are a number of ways to do this, depending on your requirements:

- Project roles enable you to associate *different people* with particular functions, for particular projects.
- Groups enable you to associate *the same people* with a particular function, for all projects that use this permission scheme.

How do I do this? (click to expand)

To add a new permission scheme called '**Software Development Permission Scheme**', and a project role called '**Quality Assurance**':

1. Create a project role called '**Quality Assurance**':
    a. Click '**Administration**' in the top navigation bar.
    b. Select **'Users' > 'Roles'** from the top menu.
    c. In the '**Name**' field, type '**Quality Assurance**'. Then click the '**Add Project Role**' button.
2. Create a permission scheme called '**Software Development Permission Scheme**', in which you assign the 'Close Issue' permission to the '**Quality Assurance**' project role:
    a. Select **'Issues' > 'Permission Schemes'** from the top menu.
    b. Click the '**Copy**' link next to '**Default Permission Scheme**'. A new permission scheme called '**Copy of Default Permission Scheme**' will be created.
    c. Click the '**Edit**' link next to '**Copy of Default Permission Scheme**'. On the '**Edit**' screen,
        i. change the '**Name**' to '**Software Development Permission Scheme**'
        ii. change the '**Description**' to '**Permission scheme for software development projects**'. Then click the '**Update**' button.
    d. Click the '**Permissions**' link next to '**Copy of Default Permission Scheme**'. On the '**Edit Permissions**' screen,
        i. for the '**Close Issues**' permission, click the '**Delete**' link next to '**Project Role (Developers)'**.
        ii. for the '**Close Issues**' permission, click the '**Add**' link. Click '**Project Role**' and choose '**Quality Assurance**'. Then click the '**Add**' button.
3. Associate the '**Software Development Permission Scheme**' with all your software development projects. Do the following for each relevant project:
    a. Click **'Projects'** and select the project of interest. The project details will be displayed.
    b. Click the name of the project's current Permission Scheme, then select '**Software Development Workflow Scheme**' and click the '**Associate**' button.
4. For each software development project, add the appropriate people to the '**Quality Assurance**' project role:
    a. Click **'Projects'** and select the project of interest. The project details will be displayed.
    b. Click **'View Project Roles'** to display the 'People' screen
    c. Select the '**Edit**' link next to '**Quality Assurance**' and add the appropriate people.

Or, to add a new permission scheme called '**Software Development Permission Scheme'**, and a group called '**quality-assurance**':

1. Create a group called '**quality-assurance**', and add the appropriate people to it.
    a. Click '**Administration**' in the top navigation bar.
    b. Select **'Users' > 'Groups'** from the top menu.
    c. In the '**Name**' field at the bottom of the page, type '**quality-assurance**'. Then click the '**Add Group**' button.
2. Create a permission scheme called '**Software Development Permission Scheme**', in which you assign the 'Close Issue' permission to the '**quality-assurance**' group.
    a. Select **'Issues' > 'Permission Schemes'** from the top menu.
    b. Click the '**Copy**' link next to '**Default Permission Scheme**'. A new permission scheme called '**Copy of Default Permission Scheme**' will be created.
    c. Click the '**Edit**' link next to '**Copy of Default Permission Scheme**'. On the '**Edit**' screen,
        i. change the '**Name**' to '**Software Development Permission Scheme**'
        ii. change the '**Description**' to '**Permission scheme for software development projects**'. Then click the '**Update**' button.
    d. Click the '**Permissions**' link next to '**Copy of Default Permission Scheme**'. On the '**Edit Permissions**' screen,
        i. for the '**Close Issues**' permission, click the '**Delete**' link next to '**Project Role (Developers)'**.
        ii. for the '**Close Issues**' permission, click the '**Add**' link. Click '**Group**' and choose '**quality-assurance**'. Then click the '**Add**' button.
3. Associate the '**Software Development Permission Scheme**' with all your software development projects. Do the following for each relevant project:
    a. Click **'Projects'** and select the project of interest. The project details will be displayed.
    b. Click the name of the project's current Permission Scheme, then select '**Software Development Workflow Scheme**' and click the '**Associate**' button.

14. Installing Plugins

Why would I do this? (click to expand)

You can install plugins to add new functionality to JIRA (e.g. additional gadgets or reports), or to change the behaviour of existing features.

How do I do this? (click to expand)

To install a plugin from the Atlassian Plugin Exchange:

1. Select **'Administration' > 'System' > 'Plugins'**. The 'Universal Plugin Manager' (UPM) will be displayed, showing the plugins installed on your JIRA site.
2. Click the '**Install**' tab in the UPM. You will see a list of featured plugins. To search for other plugins, enter some keywords that describe the plugin in the 'Search the Plugin Exchange' search box and press 'Enter'.
3. Click the '**Install**' button for the desired plugin. A confirmation message and the plugin details will appear when the plugin is installed successfully.

   Note: You may need to restart JIRA for your change to take effect. The Universal Plugin Manager will inform you if this is the case.

   Note: Not all plugins can be automatically installed. Some require manual installation. These plugins will have a 'Download' button instead of an 'Install' button. In these cases, you should read and follow the plugin's installation instructions.

For more details please see the documentation.

Important Next Steps

(Note that you need to be an Administrator to do the tasks in this section.)

15. Connecting to an External Database

Before using JIRA as a production system, you need to switch from the default HSQL database, which is provided for evaluation purposes only. Please see the documentation for details.

16. Backing up Data

To back up your JIRA data, and establish processes for regular backups, please see the documentation.

Thank you for using JIRA.

Thanks for choosing JIRA. We're always happy to help. Feel free to email or call us with any questions you have.

JIRA User's Guide

This manual contains information on how to use JIRA, the issue tracking and project management system that you access from your web browser.

JIRA Concepts

- What is a Project
- What is Workflow
- What is an Issue

Getting Started

- Logging in to JIRA
- Exploring the JIRA Workspace
- Using Keyboard Shortcuts

Working with Issues

- Attaching a File
- Attaching a Screenshot
- Cloning an Issue
- Commenting on an Issue
- Creating an Issue
- Creating a Sub-Task
- Editing an Issue
- Editing Rich-Text Fields
- Emailing an Issue
- Labelling an Issue
- Linking Issues
- Logging Work on an Issue
- Modifying Multiple ('Bulk') Issues
- Moving an Issue
- Scheduling an Issue
- Setting Security on an Issue
- Viewing an Issue's Change History
- Viewing an Issue's Crucible Reviews
- Viewing an Issue's FishEye Changesets

- Viewing the Bamboo Builds related to an Issue
- Watching and Voting on an Issue

Searching for Issues

- Using Quick Search
- Performing Text Searches
- Advanced Searching
- Using the Issue Navigator
- Customising your Issue Navigator
- Saving Searches ('Issue Filters')
- Receiving Search Results as an RSS Feed
- Exporting Search Results to Microsoft Word
- Exporting Search Results to Microsoft Excel
- Displaying Search Results as a Chart
- Displaying Search Results in XML
- Receiving Search Results via Email
- Sharing a Search Result

Generating Reports

- Workload Pie Chart Report
- User Workload Report
- Version Workload Report
- Time Tracking Report
- Single Level Group By Report
- Created vs Resolved Issues Report
- Resolution Time Report
- Pie Chart Report
- Average Age Report
- Recently Created Issues Report
- Time Since Issues Report

Browsing a Project

- Browsing a Project's Summary
- Browsing a Project's Issues
- Browsing a Project's Road Map
- Browsing a Project's Change Log
- Browsing a Project's Popular Issues
- Browsing a Project's Versions
    - Browsing a Version's Summary
    - Browsing a Version's Issues
    - Browsing a Version's Popular Issues
    - Browsing a Version's Bamboo Builds
- Browsing a Project's Components
    - Browsing a Component's Summary
    - Browsing a Component's Issues
    - Browsing a Component's Road Map
    - Browsing a Component's Change Log
    - Browsing a Component's Popular Issues
- Browsing a Project's Labels
- Browsing a Project's Bamboo Builds
- Browsing a Project's FishEye Changesets
- Browsing a Project's Crucible Reviews
- Viewing a Project's Burndown Chart

Customising the Dashboard

- Managing Multiple Dashboard Pages
- Changing the Look and Behaviour of a Gadget
- Adding the Activity Stream Gadget
- Adding the Administration Gadget
- Adding the Assigned To Me Gadget
- Adding the Average Age Gadget
- Adding the Bamboo Charts Gadget
- Adding the Bamboo Plan Summary Chart Gadget
- Adding the Bamboo Plans Gadget
- Adding the Bugzilla ID Search Gadget
- Adding the Calendar Gadget
- Adding the Clover Coverage Gadget
- Adding the Created vs Resolved Gadget
- Adding the Crucible Charts Gadget
- Adding the Favourite Filters Gadget
- Adding the Filter Results Gadget
- Adding the FishEye Charts Gadget
- Adding the FishEye Recent Changesets Gadget
- Adding the In Progress Gadget

- Adding the Introduction Gadget
- Adding the Issue Statistics Gadget
- Adding the JIRA News Gadget
- Adding the Pie Chart Gadget
- Adding the Projects Gadget
- Adding the Quick Links Gadget
- Adding the Recently Created Issues Gadget
- Adding the Resolution Time Gadget
- Adding the Road Map Gadget
- Adding the Text Gadget
- Adding the Time Since Issues Gadget
- Adding the Two-Dimensional Filter Statistics Gadget
- Adding the Voted Issues Gadget
- Adding the Watched Issues Gadget
- Adding the Heat Map Gadget
- Adding the Labels Gadget

Managing your User Profile

- Adding a User Avatar
- Allowing OAuth Access
- Changing your Password
- Choosing a Language
- Using Hover Profile
- Choosing a Time Zone

Note: for information on configuring JIRA, please see the *JIRA Administrator's Guide*.

JIRA Concepts

The following pages contain information on key concepts in JIRA:

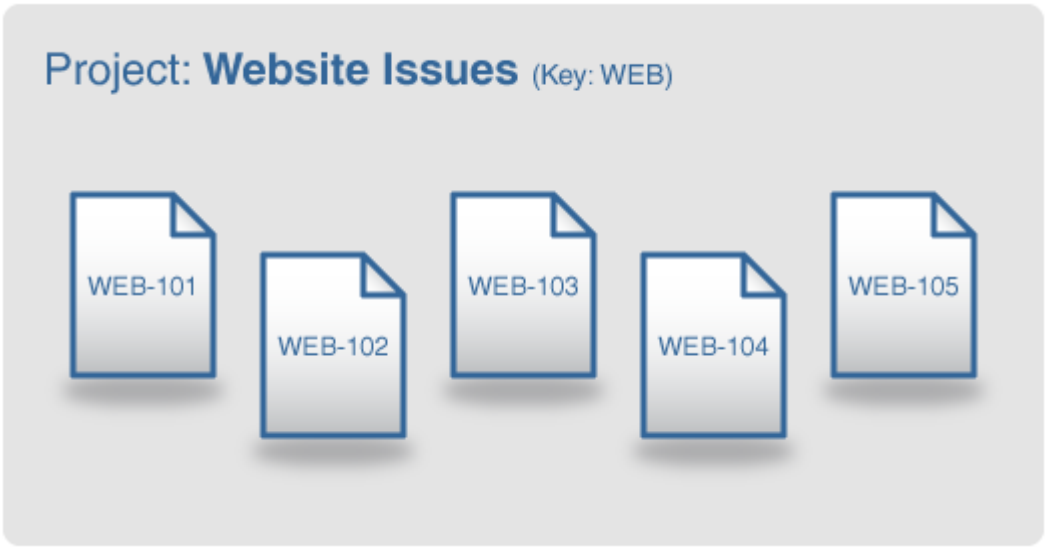
- What is a Project
- What is Workflow
- What is an Issue

What is a Project

A JIRA project is a collection of issues, and is defined according to your organisation's requirements. For example, a JIRA project could be:

- a software development project
- a marketing campaign
- a helpdesk system
- a leave request management system
- a website enhancement request system

Every issue belongs to a project. Each project has a *name* (e.g. **Website Issues**) and a *key* (e.g. **WEB**). The project key becomes the first part of that project's *issue keys,* e.g. **WEB-101, WEB-102,** etc:

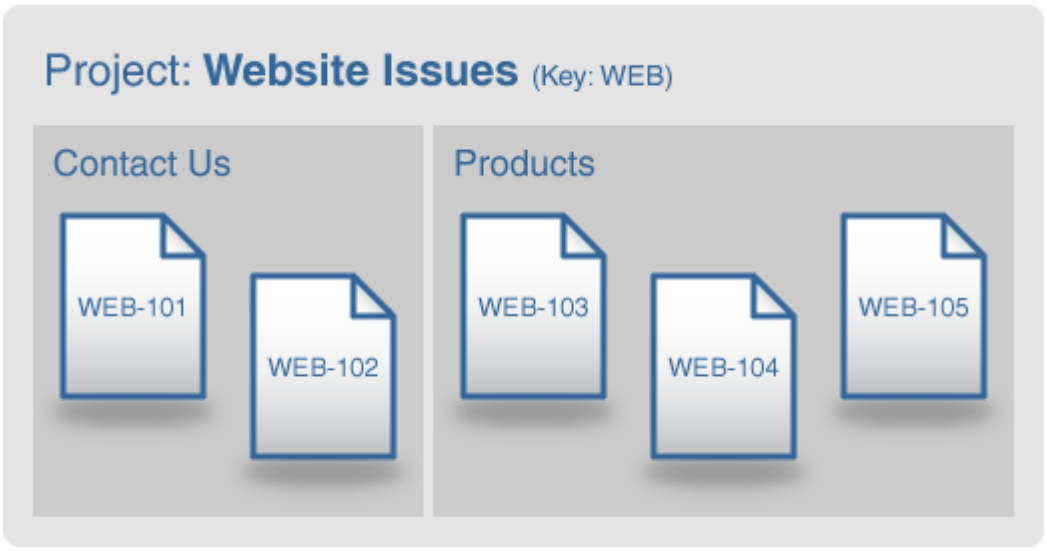


What is a component?

A project component is a logical grouping of issues within a project. Each project may consist of various components (or none), depending on

your organisation's needs.

For example, a software development project could consist of components called 'Documentation', 'Backend', 'Email Subsystem', 'GUI'. A website enhancement request system might consist of components called 'Products', 'Contact Us', etc:



An issue can belong to zero, one or multiple components within a project.

What is a version?

For some types of projects, particularly software development, it is useful to be able to associate an issue with a particular project *version* (e.g. 1.0 beta, 1.0, 1.2, 2.0).

Issues have two fields that relate to versions:

- **Affects Version(s)** — this is the version(s) in which the issue is manifesting. For instance, a software bug might affect versions 1.1 and 1.2.
- **Fix Version(s)** — this is the version(s) in which the issue was (or will be) fixed. For instance, the bug affecting versions 1.1 and 1.2 might be fixed in version 2.0. Note that issues which do not have a **Fix Version** are classified as 'Unscheduled', as shown in the screenshot above.

Versions can be in one of three states: **Released, Unreleased** or **Archived.** Versions can also have a *Release Date,* and will automatically be highlighted as 'overdue' if the version is Unreleased when this date passes.

Additional Resources

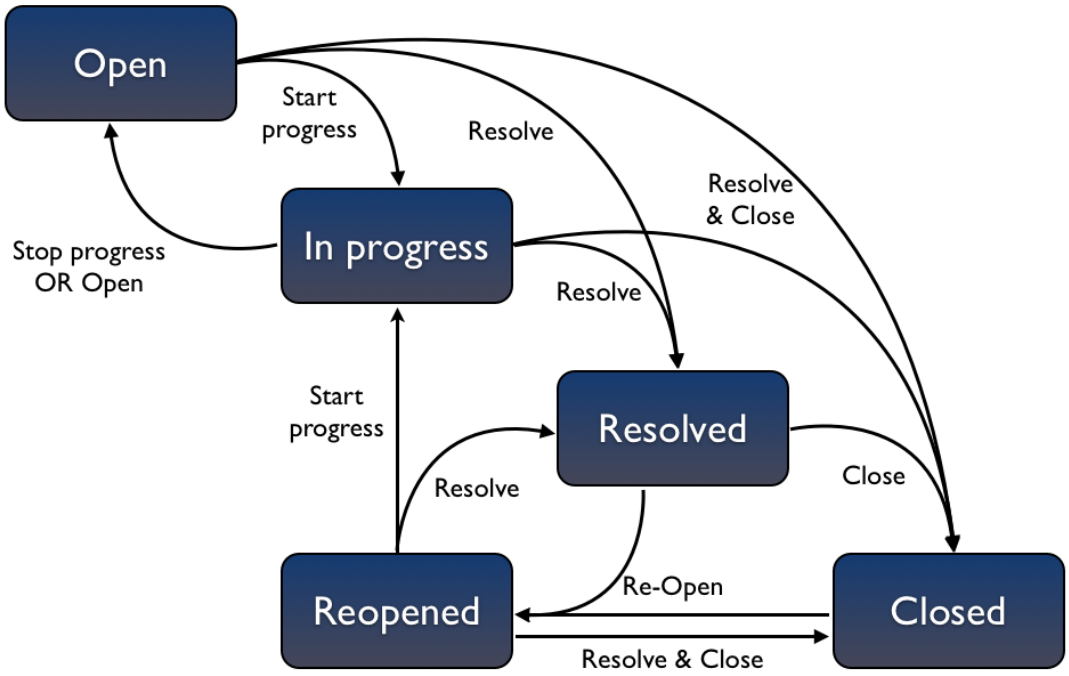
- See 'Browsing a Project' for information on looking up a project's structure and issues.
- See the *JIRA Administrator's Guide* for information on defining projects, components and versions.

What is Workflow

Workflow is the movement (or **transition**) of an issue through various **Statuses** during its lifecycle.

The following diagram shows JIRA's default workflow, where:

- each **status** is represented by a blue box
- each **transition** is indicated by an arrow.



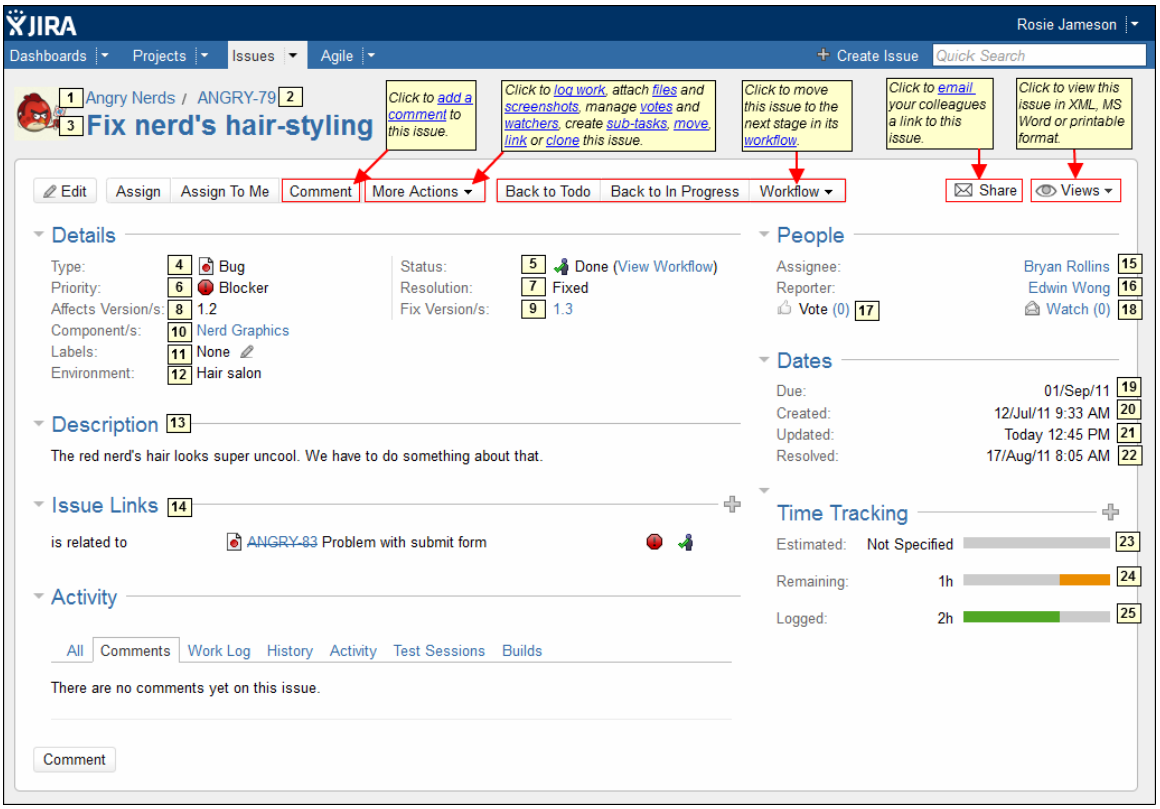
JIRA's workflow can also be customised by your JIRA administrator.

What is an Issue

Different organisations use JIRA to track different kinds of issues. Depending on how your organisation is using JIRA, an issue could represent a software bug, a project task, a helpdesk ticket, a leave request form, etc.

You can access an issue in JIRA from a search result or from a dashboard gadget that provides access to issues.

A JIRA issue typically looks like this:



> Your JIRA issues may look different to the above screenshot if your administrator has customised JIRA for your organisation.

The numbered fields shown in the above screenshot are:

1. **Project** — the 'parent' project to which the issue belongs.
2. **Key** — a unique identifier for this issue. (The characters to the left of the hyphen represent the project to which this issue belongs.)
3. **Summary** — a brief one-line summary of the issue.
4. **Type** — see below for a list of types.
5. **Status** — the stage the issue is currently at in its lifecycle ('workflow'). See below for a list of statuses.
6. **Priority** — the importance of the issue in relation to other issues. (See below for a list of priorities).
7. **Resolution** — a record of the issue's resolution, if the issue has been resolved or closed. (See below for a list of resolutions).
8. **Affects Version(s)** *(if applicable)* — project version(s) for which the issue is (or was) manifesting.
9. **Fix Version(s)** *(if applicable)* — project version(s) in which the issue was (or will be) fixed.
10. **Component(s)** *(if applicable)* — project component(s) to which this issue relates.
11. **Labels(s)** *(if applicable)* — labels to which this issue relates.
12. **Environment** *(if applicable)* — the hardware or software environment to which the issue relates.
13. **Description** — a detailed description of the issue.
14. **Links** — a list of links to related issues. (Strikethrough text, ~~like this~~, indicates that an issue has been resolved.)
15. **Assignee** — the person to whom the issue is currently assigned.
16. **Reporter** — the person who entered the issue into the system.
17. **Votes** — the number shown in brackets indicates how many votes this issue has.
18. **Watchers** — the number shown in brackets indicates how many people who are watching this issue.
19. **Due** *(if applicable)* — the date by which this issue is scheduled to be completed.
20. **Created** — the time and date on which this issue was entered into JIRA.
21. **Updated** — the time and date on which this issue was last edited.
22. **Resolved** — the time and date on which this issue was resolved.
23. **Estimate** — the **Original Estimate** of the total amount of time required to resolve the issue, as estimated when the issue was created.
24. **Remaining** — the **Remaining Estimate**, i.e. the current estimate of the remaining amount of time required to resolve the issue.
25. **Logged** — the sum of the **Time Spent** from each of the individual work logs for this issue.

Some of the most important fields — 'Type', 'Priority', 'Status' and 'Resolution' — are described as follows:

Issue Type

JIRA can be used to track many different types of issues. The default types are listed below, but please note that your JIRA administrator may have customised this list to suit your organisation.

Bug — A problem which impairs or prevents the functions of the product.

Improvement — An enhancement to an existing feature.

New Feature — A new feature of the product.

Task — A task that needs to be done.

Custom Issue — A custom issue type, as defined by your organisation if required.

Priority

An issue's priority indicates its relative importance. The default priorities are listed below; note that both the priorities and their meanings can be customised by your JIRA administrator to suit your organisation.

Blocker — Highest priority. Indicates that this issue takes precedence over all others.

Critical — Indicates that this issue is causing a problem and requires urgent attention.

Major — Indicates that this issue has a significant impact.

Minor — Indicates that this issue has a relatively minor impact.

Trivial — Lowest priority.

Status

Each issue has a status, which indicates where the issue currently is in its lifecycle ('workflow'). An issue starts as being 'Open', then generally progresses to 'Resolved' and then 'Closed'; or, depending on circumstances, it may progress to other statuses. Please also note that your JIRA administrator may have customised the available statuses to suit your organisation.

Open — This issue is in the initial 'Open' state, ready for the assignee to start work on it.

In Progress — This issue is being actively worked on at the moment by the assignee.

Resolved — A Resolution has been identified or implemented, and this issue is awaiting verification by the reporter. From here, issues are either 'Reopened' or are 'Closed'.

Reopened — This issue was once 'Resolved' or 'Closed', but is now being re-examined. (For example, an issue with a *Resolution* of 'Cannot Reproduce' is Reopened when more information becomes available and the issue becomes reproducible). From here, issues are either marked *In Progress, Resolved* or *Closed.*

Closed — This issue is complete.

Resolution

An issue can be resolved in many ways, only one of them being 'Fixed'. The default resolutions are listed below; note that your JIRA administrator may have customised these to suit your organisation.

Fixed — A fix for this issue has been implemented.
Won't Fix — This issue will not be fixed, e.g. it may no longer be relevant.
Duplicate — This issue is a duplicate of an existing issue. Note: it is recommended you create a link to the duplicated issue.
Incomplete — There is not enough information to work on this issue.
Cannot Reproduce — This issue could not be reproduced at this time, or not enough information was available to reproduce the issue. If more information becomes available, please reopen the issue.

Note that once an issue has been resolved (that is, the issue's Resolution field is not empty), textual references to that issue will show the key in strikethrough text.

Getting Started

The following pages contain information to help you get started using JIRA:

- Logging in to JIRA
- Exploring the JIRA Workspace
- Using Keyboard Shortcuts

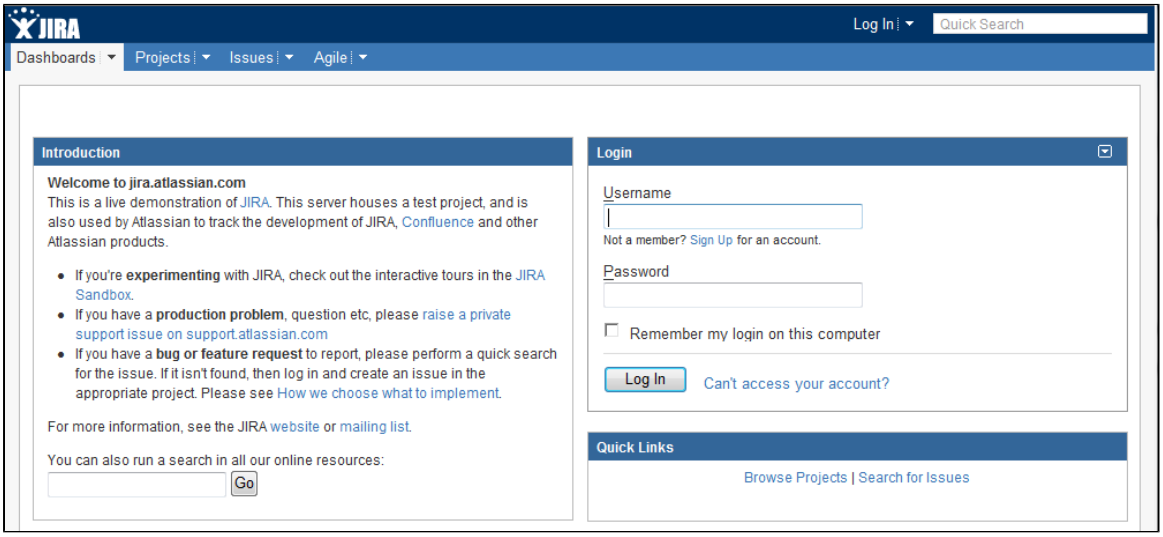
Logging in to JIRA

Many JIRA instances will have permissions implemented that restrict issues and issue actions to certain users and user groups. Some JIRA instances may not permit anonymous access. In these scenarios, you will be prompted to log in to JIRA.

The **Login** panel will be displayed if you have not logged in to JIRA. There are three things you can do here:

1. **Log in to JIRA:** To log in to JIRA, enter your **'Username'** and **'Password'** and click the **'Log In'** button.
   *Selecting the **'Remember my login on this computer'** check box will prevent you from being automatically logged out of JIRA. However, your session will not be preserved, e.g. last search, current project, etc.*
2. **Reset your password:** To reset your password, click the **'Forgot Password'** link. The reset password page will display. Enter your **'Username'** and a new password will be emailed to the email address specified in your JIRA user profile. If you have forgotten your email address, you will need to contact your JIRA administrator for help.
3. **Sign up for an account:** If you do not have a user account and your JIRA administrator has enabled public signup, you can create your own user account by clicking the **'Sign up'** link in the 'Not a member? Sign up for an account' text. Enter your details and click the **'Sign up'** button to create your account.

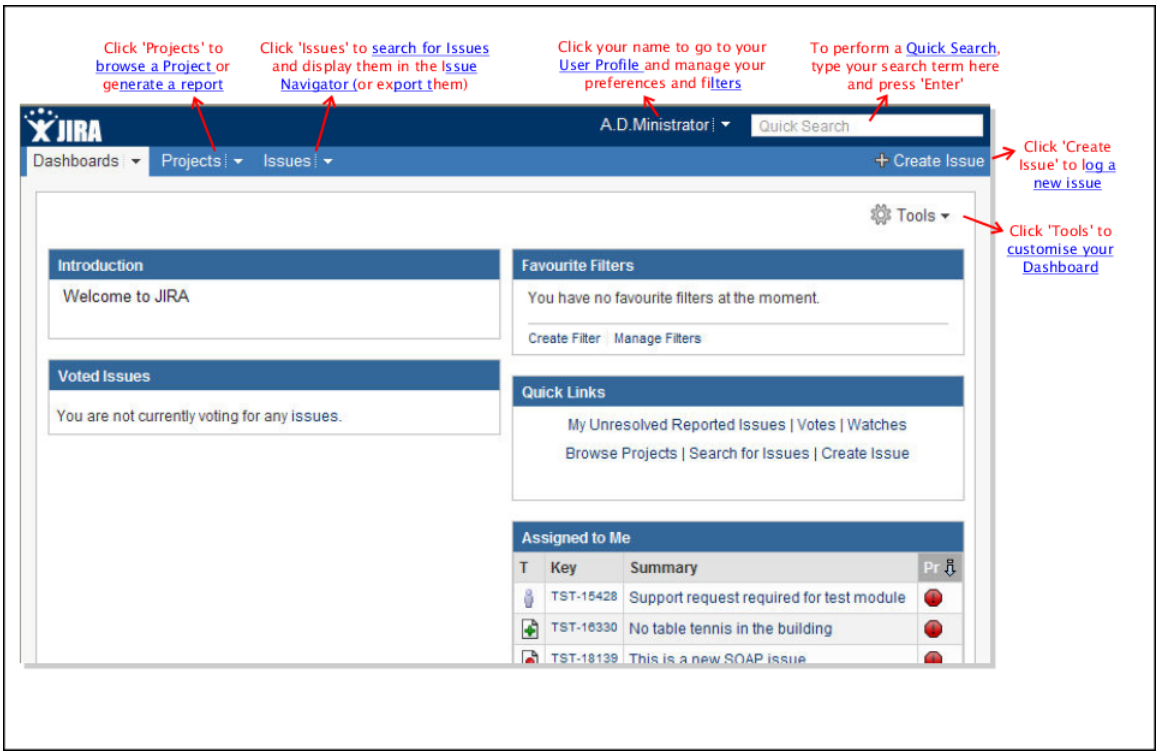
Please note that your JIRA screen may look different from the following screenshot as the dashboard and colours may have been customised by your JIRA administrator. The links, however, will be the same.



Exploring the JIRA Workspace

The **Dashboard** is the first screen you see when you login to JIRA.

- The navigation bar (at the top of the screen) is the same on every screen in JIRA. It contains links which give you quick access to many of JIRA's most useful functions.
- The white area of the screen, below the top navigation bar, can be customised to display 'gadgets' showing many different types of information, depending on your areas of interest.



Please note that your JIRA screen may look different from this screenshot, as the logo and colours may have been customised by your JIRA administrator. The links in the navigation bar, however, will be the same.

Using Keyboard Shortcuts

Keyboard shortcuts provide a quick and easy way of navigating though JIRA and performing fundamental actions on issues without having to take your fingers off the keyboard.

You may not have permission to perform all the keyboard shortcut actions described on this page. This depends on how your JIRA administrator(s) have configured permissions for your user account.

Keyboard Shortcuts

Global Shortcuts

These shortcuts are available from any JIRA screen.

Operation	Keyboard Shortcut	Description
Go to Dashboard	'g' then 'd'	Directs you to the Dashboard screen.
Browse to a Project	'g' then 'p'	Directs you to your current project browser screen.
Find Issues	'g' then 'i'	Opens the Issue Navigator, where you can search for issues using either the Simple Search or Advanced Search features.

Quick Search	'/'	Directs your cursor to the Quick Search text field in the top right-hand corner. In 'Administration' mode, this keyboard shortcut directs your cursor to the 'Administration Quick Search' text field (also in the top right-hand corner), whose functionality is identical to the Administration Search Dialog Box (below).
Create an Issue	'c'	Opens the Create Issue page or the project and issue type selection bubble for creating an issue.
Open shortcut help	'?'	Opens the keyboard shortcuts dialog box (described below). To close this dialog box, press the 'Esc' key or click '**Close**' in the lower-right of the box.
Dashboards drop-down menu	Modifier key(s) + 'd'	Opens the 'Dashboards' drop-down menu in the top navigation bar. You can then use the arrow keys to navigate to an item, then press 'Enter' to select it. Modifier Keys are specific to each combination of browser and operating system. Refer to the Modifier Keys section below for more details.
Projects drop-down menu	Modifier key(s) + 'p'	Opens the 'Projects' drop-down menu in the top navigation bar. You can then use the arrow keys to navigate to an item, then press 'Enter' to select it.
Issues drop-down menu	Modifier key(s) + 'i'	Opens the 'Issues' drop-down menu in the top navigation bar. You can then use the arrow keys to navigate to an item, then press 'Enter' to select it.
Administration Search Dialog Box (Only available if you have the 'JIRA administrators' global permission.)	'g' then 'g'	Opens the **Administration Search** dialog box. • Press the 'Cursor Down' key to show a list of all Administration options, then: 1. use the cursor keys to select an Administration option, and 2. press 'Enter' to choose your selected option. • Type one to a few letters of the Administration option's name to restrict the list down to options whose names match the series of letters you entered, then: 1. if there is more than one option in the restricted list, use the cursor keys to select one, and 2. press 'Enter' to choose your selected option.

Keyboard shortcuts are not available on JIRA's **Workflow Designer** page.

On this page:

- Keyboard Shortcuts
  - Global Shortcuts
  - 'View Issue' and 'Issue Navigator' Shortcuts
  - Form Shortcuts
- Modifier Keys
- Accessing the Keyboard Shortcuts Dialog Box
- Disabling and Re-enabling Keyboard Shortcuts
- RELATED TOPICS

'View Issue' and 'Issue Navigator' Shortcuts

These shortcuts are available from JIRA's View Issue and Issue Navigator screens only. These shortcuts allow you to:

- navigate through a list of issues and between individual items on an issue, and
- perform actions and other operations on an issue.

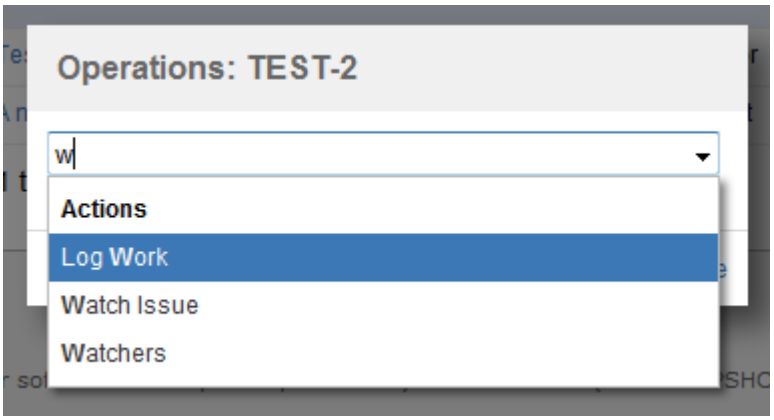
Operation	Keyboard Shortcut	Description	Availability
View selected Issue	'o' or 'Enter'	Opens the currently selected issue on the **Issue Navigator**.	Issue Navigator
Next Issue	'j'	Navigates to the next issue. This keyboard shortcut only applies to a **View Issue** screen if you got there via the **Issue Navigator**.	View Issue and Issue Navigator

Previous Issue	'k'	Navigates to the previous issue. This keyboard shortcut only applies to a **View Issue** screen if you got there via the **Issue Navigator**.	View Issue and Issue Navigator
Back to the Navigator	'u'	Returns to the **Issue Navigator** from the **View Issue** screen. This keyboard shortcut only applies to a **View Issue** screen if you got there via the **Issue Navigator**.	View Issue
Hide/Show Left Section	'[' (left square bracket)	Hides or shows the left section on the **Issue Navigator**.	Issue Navigator
Next Activity	'n'	Navigates to the next item in the activity section of the **View Issue** screen.	View Issue
Previous Activity	'p'	Navigates to the previous item in the activity section of the **View Issue** screen.	View Issue
Focus search field	'f'	Focuses the cursor in the Advanced Search text field or the Simple Search's Query text field.	Issue Navigator
Escape Field	'Esc'	Escapes the cursor out of the current text field so that you can use more keyboard shortcuts. Pressing the 'Esc' key can also be used to **Close** or **Cancel** JIRA's dialog boxes. If the auto-complete dropdown list is visible on the Advanced Search text field, you must press the 'Esc' key twice to escape the cursor out of this text field.	View Issue and Issue Navigator
Edit Issue	'e'	Opens the **Edit Issue** page (if you have appropriate permission), where you can edit the issue.	View Issue and Issue Navigator
Assign Issue	'a'	Opens the **Assign** dialog box (if you have appropriate permission), where you can assign the issue to another JIRA user.	View Issue and Issue Navigator
Comment on Issue	'm'	On the **View Issue** screen, this opens the comment panel at the top of the page and focuses on the comment text box. On the **Issue Navigator**, this opens the **Add Comment** dialog box for adding a comment to the currently selected issue.	View Issue and Issue Navigator
Edit Issue Labels	'l' (lower-case 'L')	Opens the **Labels** dialog box, where you can edit the labels associated with the issue.	View Issue and Issue Navigator
Share Issue	's'	Opens the **Share** bubble, which allows you to send a link to an issue or search result (with an optional note) to the email address of any JIRA user, or any arbitrary email address. See Sharing an Issue or Sharing a Search Result for details.	View Issue and Issue Navigator
Operations Dialog Box	'.' (full-stop/period)		View Issue and Issue Navigator

Opens the **Operations** dialog box, from which you can perform any issue operation on the current JIRA issue by doing either of the following:

- Press the 'Cursor Down' key, then:
    1. use the cursor keys to select an issue operation, and
    2. press 'Enter' to choose your selected operation.
- Type one to a few letters of the issue operation's name to restrict the list down to operations whose names match the series of letters you entered, then:
    1. if there is more than one operation in the restricted list, use the cursor keys to select one, and
    2. press 'Enter' to choose your selected operation.

Screenshot: The Operations Dialog Box



The list of issue operations available in the drop-down menu are split into the following categories:

- **Workflow** Transitions — Choose the appropriate workflow transition to change the issue's status. The workflow transitions available depend on the current issue's status and on how your JIRA administrator has customised your JIRA workflow. The default JIRA workflow transitions include:
  - **Start Progress** — Set the issue's Status to **In Progress**.
  - **Resolve issue** — Set the issue's Status to **Resolved** and select the appropriate Resolution.
  - **Close issue** — Set the issue's Status to **Closed** and if the issue has not already been **Resolved**, select the appropriate Resolution.
  - **Reopen issue** — Set a **Resolved** or **Closed** issue's Status to **Reopened**.
- **Actions** — Choose the appropriate action to perform on the issue.
  - **Edit** — Edit the issue's details (**Summary, Description,** etc).
  - **Assign** — Select an asignee for the issue.
  - **Assign To Me** — Assign the issue to yourself.
  - **Comment** — Add a comment to the issue.
  - **Log Work** — Record the work done and time spent on the issue. This option is only available if Time Tracking has been activated on your JIRA site.
  - **Attach Files** — Select a file, upload it and attach it to the issue.
  - **Attach Screenshot** — Select a file, upload it and attach it to the issue.
  - **Voters** — Opens the Voters list of the issue, where you can manage your vote and see others who have voted on the issue too.
  - **Add Vote** — Adds your vote to the issue. (This option is only available if you did not create the issue.)
  - **Watch Issue** — Become a watcher of the issue.
  - **Stop Watching** — Stop watching the issue. (This option is only available on issues you are currently watching.)
  - **Watchers** — Opens the Watchers List, where you can manage watchers of the issue.
  - **Create Sub-Task** — Create a new issue which is a sub-task of the issue.
  - **Convert to Issue** — If the issue is a sub-task, convert it to a standalone issue.
  - **Convert to Sub-Task** — If the issue is a standalone issue, convert it to a sub-task.
  - **Move** — Move the issue to a different project.
  - **Link** — Create a link between the issue and another issue. This option is only available if Issue Linking has been enabled on your JIRA site.
  - **Clone** — Create a new issue which is an identical copy of the issue.
  - **Labels** — Edit the issue's labels.
  - **Delete** — Permanently remove the issue.

Note that some options in the **Actions** menu will only be available if you have the necessary permissions, or if certain features have been enabled by your JIRA administrator.

Form Shortcuts

These shortcuts are available on JIRA forms, including those involved in editing an issue's fields, such as the Create Issue or Edit Issue forms and JIRA login forms.

Operation	Keyboard Shortcut	Description
Form Submit	Modifier key(s) + 's' (Alt + 's' only in Chrome on Windows or Linux/Solaris)	Submits any currently open form in JIRA. The keyboard shortcuts dialog box will show which modifier key (or keys) are required for your combination of web browser and operating system. Modifier Keys are specific to each combination of browser and operating system. Refer to the Modifier Keys section below for more details.
Cancel Form	Modifier key(s) + ` (backquote)	Cancels any currently open form in JIRA.
Escape Field	'Esc'	Escapes the cursor out of any field on the currently open form, so that you can use more keyboard shortcuts. Pressing the 'Esc' key can also be used to **Close** or **Cancel** JIRA's dialog boxes.

Back	'u'	Returns to the **View Issue** screen from any form that results from an action performed on that issue. You must have escaped out of all fields on the form (by pressing 'Esc') before using this keyboard shortcut.
'Remember my login' check box	Modifier key(s) + 'r'	Automatically selects the 'Remember my login on this computer' check box. This shortcut only applies to JIRA's login forms.

Modifier Keys

If a keyboard shortcut requires modifier keys, one or two of these modifier keys (for example, 'Shift', 'Alt' or 'Ctrl') must be pressed simultaneously, along with a single 'action' key. In the shortcuts dialog box, these keystrokes are indicated as 'modifier (+ modifier) + x', where 'x' is an action key, with the exception of a 'Shift + x' key combination.

Modifier keys differ depending on your combination of operating system and web browser. For example, when running Firefox on Mac OS X, you will need to press 'Ctrl' + 'S' to submit a form, while on Windows, you will need to press 'Alt' + 'S'. The following table identifies the modifier keys for the various combinations of supported web browsers and operating systems:

Web Browser	Mac OS X	Windows	UNIX/Linux
Firefox	Ctrl	Alt + Shift	Alt + Shift
Internet Explorer		Alt	
Safari	Ctrl + Alt/Option	Ctrl	
Chrome	Ctrl + Alt/Option	Alt + Shift	Alt + Shift

Please note:

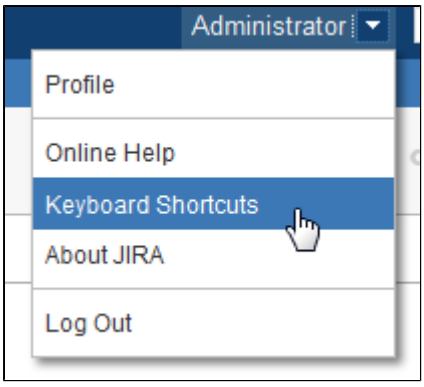
- **Modifier key shortcuts** differ from **two-key shortcuts**. For the latter, one discrete keystroke follows another (without the keys being pressed simultaneously). In the shortcuts dialog box, **two-key shortcuts** are indicated as 'x then y', where 'x' is the first keystroke and 'y' is the second.
- In Internet Explorer, typing a 'Modifier key shortcut' that leads to a link results in the link being highlighted only. Hence, after typing a modifier key shortcut, you will then need to press the 'Enter' key to complete the action - for example, to open a link's drop-down menu. The actions of modifier key shortcuts that lead to buttons, however, are fully completed.
- In Firefox, it is possible to customise 'Modifier key shortcuts'. Please read Mozilla's Ui.key.contentAccess documentation for more information.

Accessing the Keyboard Shortcuts Dialog Box

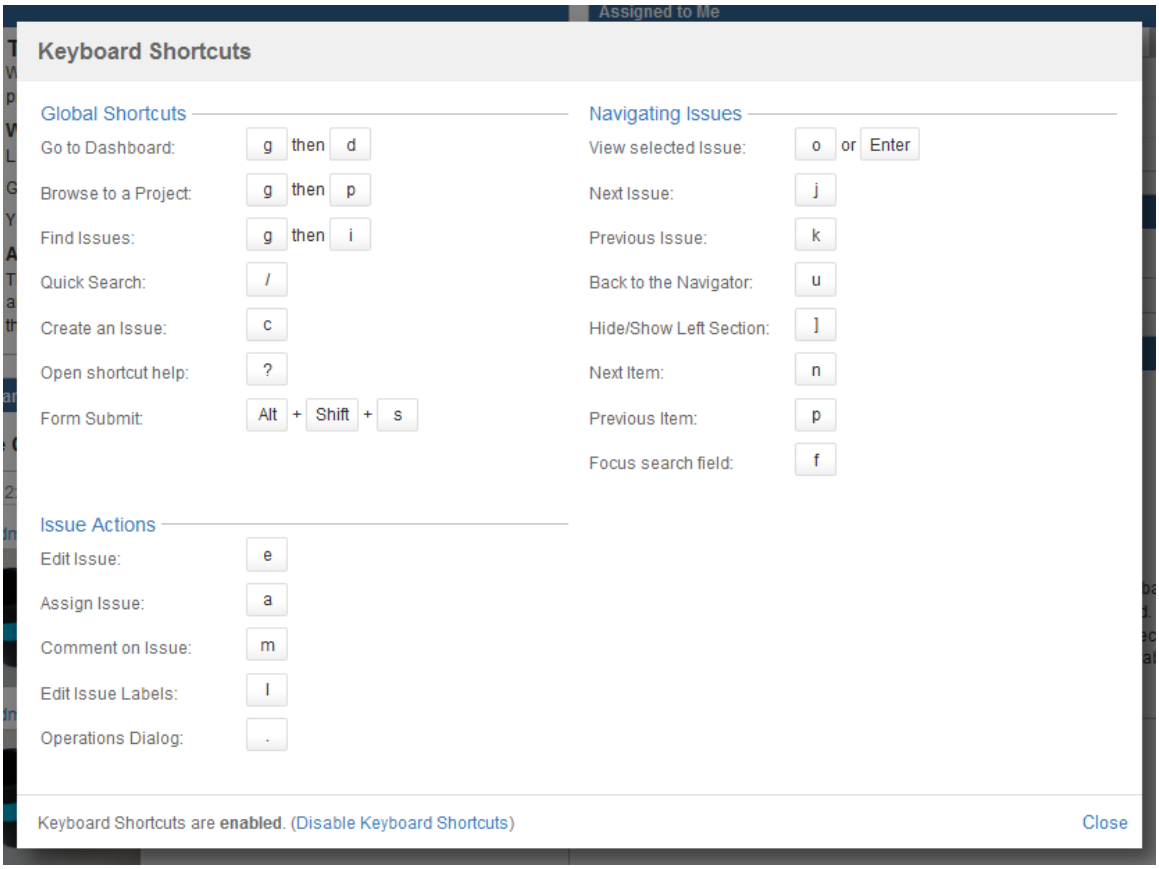
The keyboard shortcuts dialog box shows an overview of JIRA actions that are available as keyboard shortcuts and the combination of keystrokes required to perform them.

You can quickly open this dialog box by pressing '?' ('Shift + /') on your keyboard, or by clicking your username's dropdown and selecting '**Keyboard Shortcuts**' from the list.

Screenshot: Accessing 'Keyboard Shortcuts' from Your Username's Dropdown



Screenshot: Keyboard Shortcuts Dialog Box in Firefox on Windows



Please Note:

- Be aware that when you press '?', the keyboard shortcuts dialog box will not appear if your cursor is already focused inside any JIRA text entry field. Press 'Esc' first to escape from a field.
- If you have GreenHopper installed, you will see a series of **Agile** keyboard shortcuts in the lower-right of this dialog box (and some additional **Global** keyboard shortcuts specific to GreenHopper in the upper-left section). However, the keyboard shortcuts in the **Agile Shortcuts** section only function in GreenHopper and not in a JIRA context.

Disabling and Re-enabling Keyboard Shortcuts

Keyboard shortcuts are enabled by default. However, you can disable them on a per-user basis via the Keyboard Shortcuts dialog box.

To disable or re-enable keyboard shortcuts:

1. Ensure you are logged in to JIRA and open the Keyboard Shortcuts dialog box (described above).
2. At the bottom of the Keyboard Shortcuts dialog box, click '**Disable Keyboard Shortcuts**' or '**Enable Keyboard Shortcuts**' to respectively disable or re-enable keyboard shortcuts for the currently logged in user.

   Alternatively, you can disable or re-enable keyboard shortcuts by editing the **Preferences** section of the Summary Tab of your User Profile.

RELATED TOPICS

- GreenHopper Keyboard Shortcuts
- Keyboard Shortcut Plugin Module

Working with Issues

The following pages contain information on working with issues:

- Attaching a File
- Attaching a Screenshot
- Cloning an Issue
- Commenting on an Issue
- Creating an Issue
- Creating a Sub-Task
- Editing an Issue
- Editing Rich-Text Fields
- Emailing an Issue
- Labelling an Issue
- Linking Issues
- Logging Work on an Issue

- Modifying Multiple ('Bulk') Issues
- Moving an Issue
- Scheduling an Issue
- Setting Security on an Issue
- Viewing an Issue's Change History
- Viewing an Issue's Crucible Reviews
- Viewing an Issue's FishEye Changesets
- Viewing the Bamboo Builds related to an Issue
- Watching and Voting on an Issue

Attaching a File

JIRA allows you to attach files to an issue.

> To be able to attach files, your JIRA administrator must have enabled file attachments.

On this page:

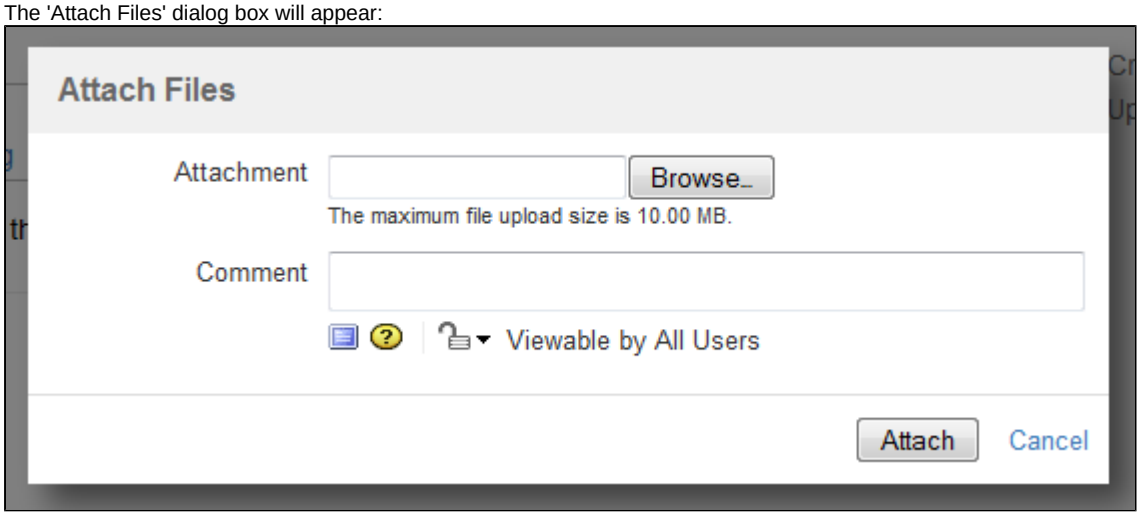
- Attaching a File to an Issue
- Viewing an Image Gallery
- Sorting Attachments
- Accessing ZIP-format File Contents
- Exporting All Attachments as a ZIP File
- Removing a File Attachment from an Issue

Attaching a File to an Issue

To attach files to an issue, you need the **Create Attachments** project permission in that issue's project.

To attach a file to a JIRA issue:

1. Open the JIRA issue to which you wish to attach a file.
2. From the 'More Actions' menu, select **Attach File**.
3. The 'Attach Files' dialog box will appear:



4. Click the **Browse** button to search for your files.

> - You can attach more than one file at a time: after you select a file, it will appear at the top of the 'Attach Files' dialog box, followed by each subsequent file you add via the 'Browse' button.
> - If you use Firefox 3.6 or later or Chrome browsers, you can select multiple files in the browser dialog box which appears after clicking the 'Browse' button.
> - If you select a file by mistake, clear its check box to prevent the file being attached to the issue.
> - By default, the maximum size of any one file is 10MB, although this limit can be customised by your JIRA administrator. See Configuring File Attachments for more information.
> - File names cannot contain any of these characters: `'\', '/','\"', '%', ':', '$', '?', '*'`.

5. (Optional) Enter a comment about the files(s) you are attaching.
   If you enter a comment, then you can also set the security level for the comment by selecting the appropriate value from the padlock icon dropdown. The security level for the comment is '`All Users`' by default.
6. Click the **Attach** button. All selected files will be attached to the issue.

Viewing an Image Gallery

By default, JIRA displays image files (ie. GIFs, JPGs, PNGs) attached to an issue, including any screenshots, as 'thumbnails' on the issue.

Screenshot: Viewing attached image files on an issue

If your 'JIRA administrator' has disabled 'Thumbnails' in JIRA's Attachment Settings, then JIRA will not display thumbnails of image files on an issue and instead, they will be part of the list of attached files.

You can click on an image's thumbnail to preview a larger version of it in a popup. If multiple images are attached to an issue, click the left and right arrows at the sides of an image preview to see previews of the adjacent attached images.

Screenshot: Previewing attached image files on an issue

Sorting Attachments

You can sort the list of attachments on JIRA issues in ascending or descending order according to the attachment file name or date of attachment to the issue.

To sort your list of attachments:

1. Open a JIRA issue with attachments.
2. Click the down-arrow icon to the right of the **Attachments** section and select the required sort criteria or order options provided in the list.

Your final sort criteria and order options will also apply to image thumbnails and all subsequent issues viewed.

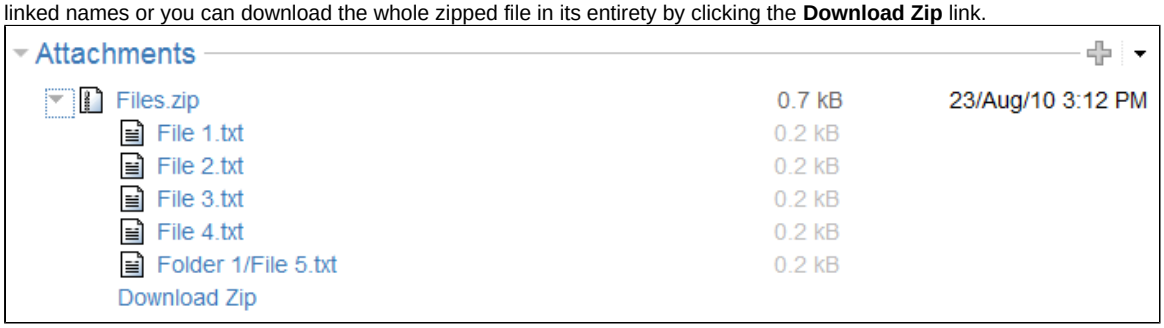
Accessing ZIP-format File Contents

When viewing an issue, JIRA allows you to browse and access the contents of any ZIP-format file (such as those with a '`.zip`' or '`.jar`' file name extension) attached to an issue.

To browse the contents of a zipped attachment and access its files:

1. Open a JIRA issue with an attached zipped file.
2. Click the right-arrow icon to the left of the zipped file's name.

3. The contents of the zipped file are listed. From this expanded list, you can access the zipped file's individual files by clicking their linked names or you can download the whole zipped file in its entirety by clicking the **Download Zip** link.



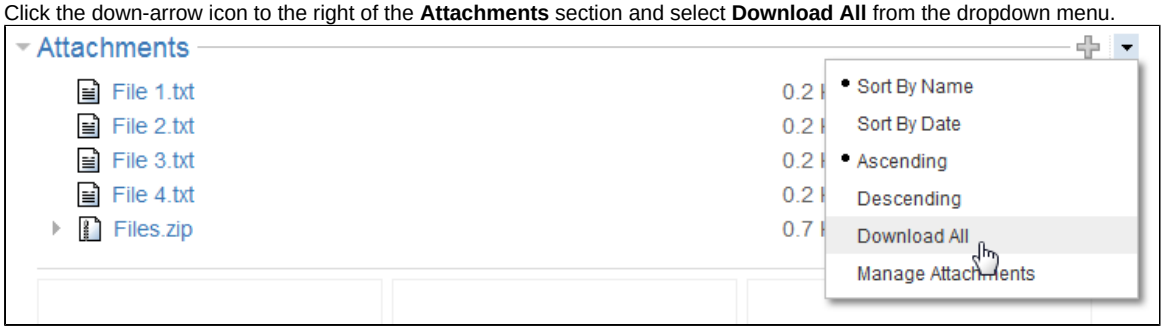
If a file is located within a subdirectory of the zipped file, then the path to that file is indicated in the content of the zipped file. For example, the content of **Files.zip** listed in the screenshot above shows that **File 5.txt** is located within the **Folder 1** subdirectory of **Files.zip**.

If your 'JIRA Administrator' has disabled 'ZIP support' in JIRA's Attachment Settings, then this feature will not be available and you must download the zip file to your computer before accessing its individual files.

Exporting All Attachments as a ZIP File

To download all the files attached to an issue (including any screenshots) as a single ZIP file:

1. Open the JIRA issue from which you wish to export all attachments as a zip file.
2. Click the down-arrow icon to the right of the **Attachments** section and select **Download All** from the dropdown menu.



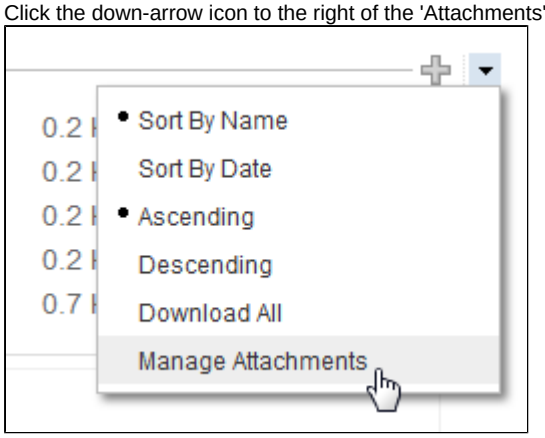
Removing a File Attachment from an Issue

To remove attachments from an an issue, you need either of the following the project permissions in that issue's project:

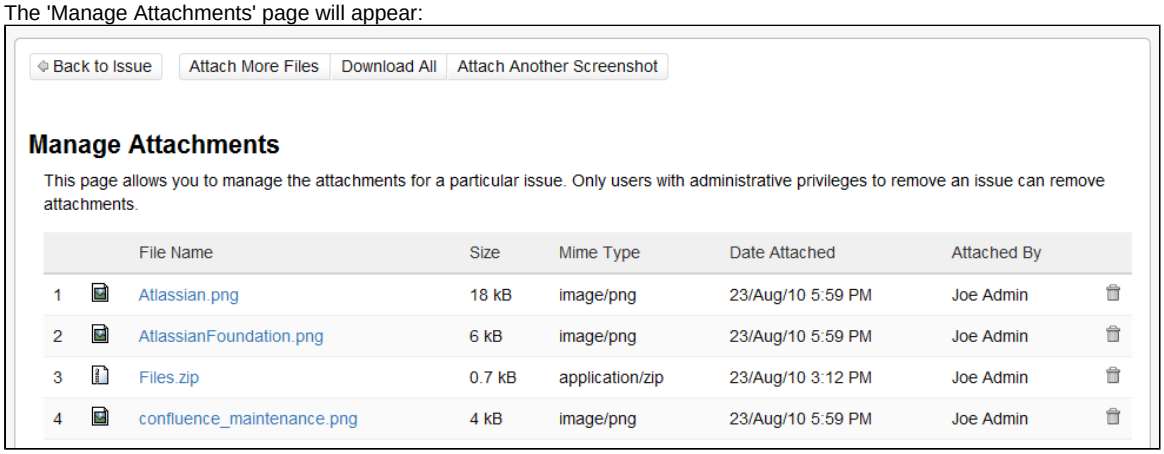
- 'Delete Own Attachments' — to delete files which you have added to the issue.
- 'Delete All Attachments' — to delete files which anyone has added to the issue.

To remove a file attachment from a JIRA issue:

1. Open the JIRA issue from which you wish to remove a file.
2. Click the down-arrow icon to the right of the 'Attachments' section and select **Manage Attachments** from the dropdown menu.



The 'Manage Attachments' page will appear:



Note: "Only users with administrative privileges to remove an issue can remove attachments" indicates that any user with the appropriate 'Delete Issues' project permissions can remove all attachments of an issue they delete, even if they do not have the specific project permissions (above) to delete these attachments via the 'Manage Attachments' page.

3. Locate the file you wish to delete and click the **Delete Attachment** icon:

Attaching a Screenshot

JIRA allows you to attach screenshots you have captured to an issue, if the administrator has configured JIRA and your permissions appropriately.

This feature is only available if:

1. Your JIRA administrator has file attachments enabled. (You will also need the 'Create Attachments' permission in the appropriate projects.)
2. You are using a Windows or Mac client. (If you use another operating system, you can attach a screenshot using the file attachment feature instead.)
3. The computer you are using to access JIRA uses a Java version 1.6+ platform. (JIRA uses a Java applet to run the **'Attach Screenshot'** functionality in a separate browser window.)

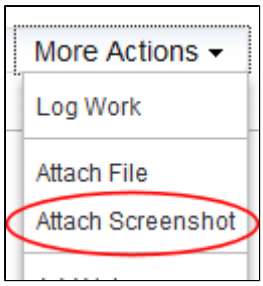
On this page:

- Attaching a Screenshot
- Capturing Screenshots
  - Capturing a screenshot on Windows
  - Capturing a screenshot on Mac OSX

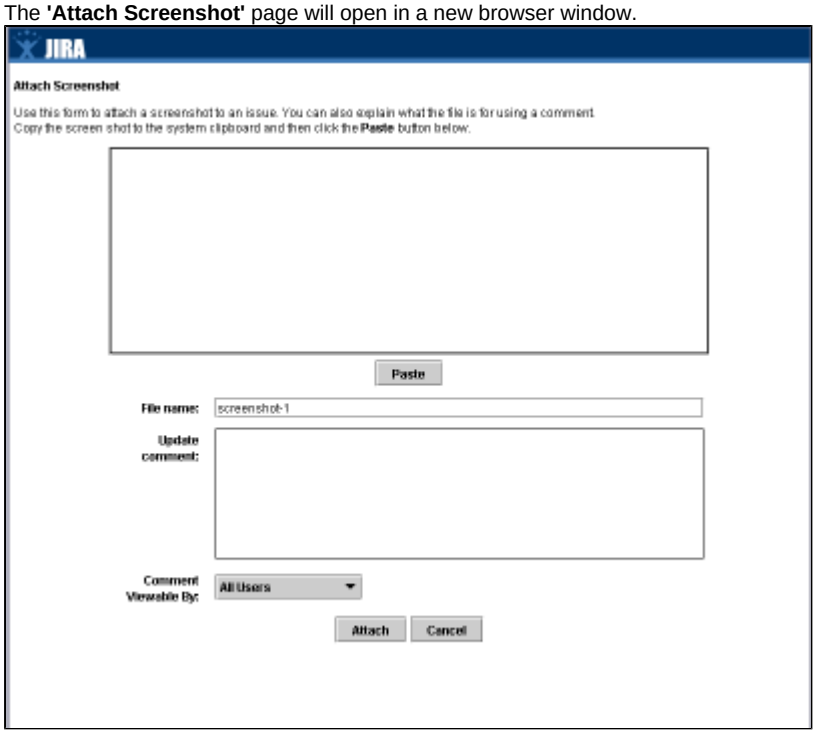
Attaching a Screenshot

To attach a screenshot:

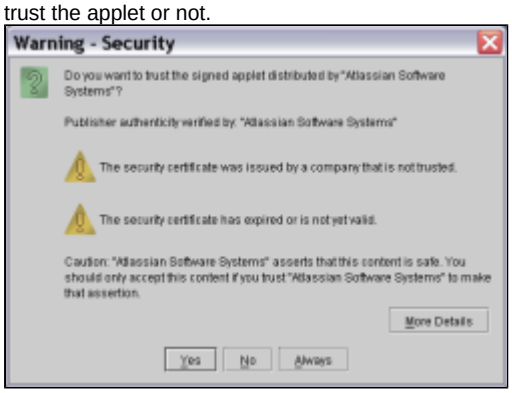
1. Open the JIRA issue to which you wish to attach a file.
2. From the **'More Actions'** menu, select **'Attach Screenshot'**:



3. The **'Attach Screenshot'** page will open in a new browser window.

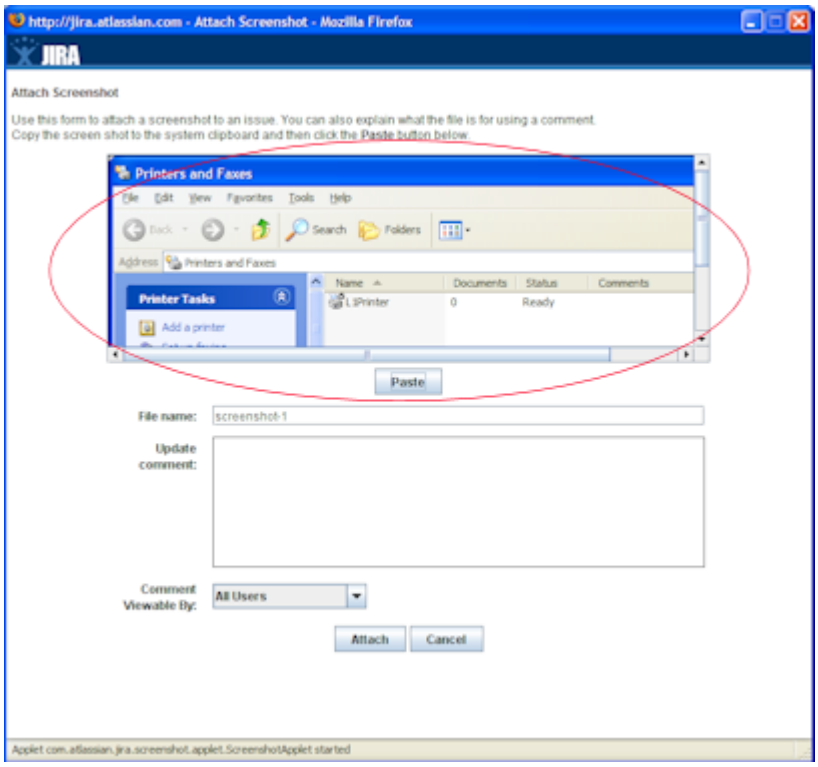


If this is the first time you have used this function, a security warning will also display in a dialog box asking you whether you want to trust the applet or not.



Choose the **'Yes'** option to trust the applet and access the **'Attach Screenshot'** page.

4. Ensure that you have captured an image to your operating system's clipboard and click the **'Paste'** button to paste the image. Your captured image should display in the blank area above the **'Paste'** button.

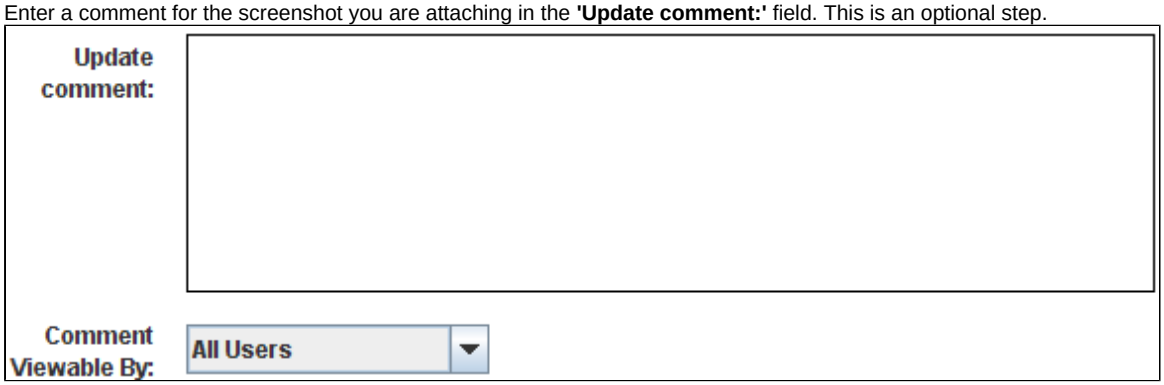


Please note that clicking the **'Attach'** button before an image has been pasted will not attach anything to the JIRA issue.

5. Enter a file name for the screenshot you are attaching in the **'File name:'** field. The file name will be initially defaulted to '`screenshot-1`'.

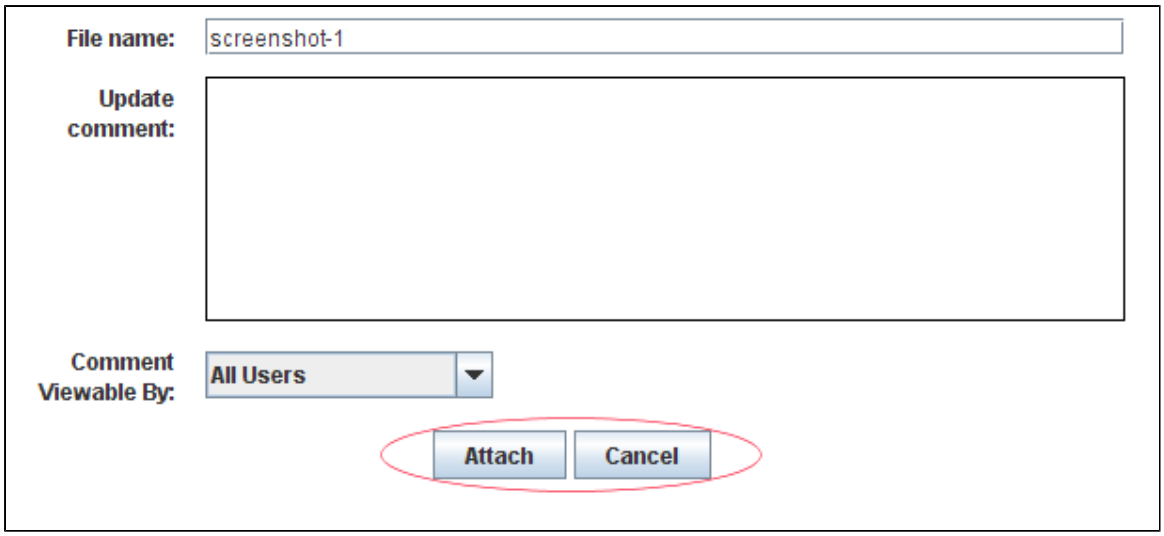
A valid file name cannot contain any of these characters: `'\', '/','\"', '%', ':', '$', '?', '*'`.
If an invalid file name is entered, an error message will display when the **'Attach'** button is clicked and the screenshot will not be attached to the JIRA issue.

6. Enter a comment for the screenshot you are attaching in the **'Update comment:'** field. This is an optional step.



If you have entered a comment, then you can also set the security level for the comment by selecting the appropriate value from the **'Comment Viewable By:'** dropdown. The security level for the comment will initially be defaulted to '`All Users`'.
The comment entered will be added to the JIRA issue, with the selected security level, when the screenshot is attached.

7. Click the **'Attach'** button to attach the captured image to your JIRA issue. The window will close and you will be returned to your original JIRA issue. At any time you may click the **'Cancel'** button to close the window without attaching anything to the JIRA issue.



Capturing Screenshots

The method for capturing screenshots differs on each operating system, as described below:

Capturing a screenshot on Windows

- New screenshot capture — To capture a screenshot into the system clipboard, use either of the following keyboard combinations:
  - press **ALT-PRINTSCREEN** to capture your currently selected window; or
  - press **CTRL-ALT-PRINTSCREEN** to capture the whole desktop
- Existing image — Open your existing image in your favourite imaging application and select the copy option from the appropriate menu to capture the image into the system clipboard.

Capturing a screenshot on Mac OSX

- New screenshot capture — To capture a screenshot into the system clipboard, use either of the following keyboard combinations:
  - press **CTRL-APPLE-SHIFT-4** to capture your currently selected window; or
  - press **CTRL-APPLE-SHIFT-3** to capture the whole desktop
- Existing image — Open your existing image in your favourite imaging application and select the copy option from the appropriate menu to capture the image into the system clipboard.

Cloning an Issue

'Cloning' (copying) an issue allows you to quickly create a duplicate of an issue within the same project. The clone issue is a replica of the original issue, containing the same information stored in the original issue — e.g. Summary, Affects Versions, Components, etc. The clone issue can also be linked to the original issue using a 'clone' link.

A clone issue is a separate entity from the original issue. Operations on the original issue have no effect on the clone issue and vice versa. The only connection is a link (if created) between the original and the clone issue.

On this page:

- Creating a Clone Issue
- Cloned Issue Linking Behaviour
- Cloned Issue Summary Field Prefix
- Cloning and Sub-Tasks

A clone issue retains the following information:

- Summary — with optional prefix that can be customised by your JIRA system administrator. (See Configuring Issue Cloning for details.)
- Description
- Assignee
- Environment
- Priority
- Issue Type
- Security
- Reporter (Note that if you do not have the **'Modify Reporter'** permission, the clone issue will be created with you as the Reporter.)
- Components
- Affects Versions
- Fix For Versions
- Issue Links (optional)

- Attachments (optional)
- Project (Note that once the clone has been saved, you can move it to another project as described in Moving an Issue.)

Details such as time tracking and comments are not cloned.

Creating a Clone Issue

To clone an issue,

1. Open the JIRA issue which you wish to clone.
2. From the **'More Actions'** menu, select **'Clone'**. The '**Clone Issue**' screen will appear.
    - You can edit the clone issue's Summary if you wish.
    - If the issue contains links to other issue(s), you can select whether or not to include the links in the new clone issue.
    - If the issue contains sub-tasks, you can select whether or not to create the sub-tasks in the new clone issue.
    - If the issue contains attachments, you can select whether or not to include the attachments in the new clone issue.
3. Click the '**Create**' button.

Cloned Issue Linking Behaviour

By default, when an issue is cloned, JIRA will automatically create a link between the original and cloned issue using the pre-existing link type name 'Cloners'.

Your JIRA system administrator can customise this default behaviour by either preventing links from being created or changing the name of the link type. See Configuring Issue Cloning for details.

Cloned Issue Summary Field Prefix

By default, the 'Summary' field of a cloned issue is prefixed with the string 'CLONE - ' to indicate that the issue is a clone.

Your JIRA system administrator can customise this default behaviour by either changing the prefix string or preventing the addition of prefixes on cloned issues. See Configuring Issue Cloning for details.

Cloning and Sub-Tasks

Sub-Tasks can be cloned in the same manner as other issue types.

If the original issue has associated sub-tasks, that issue's sub-tasks will also be cloned. The summary of a cloned sub-task will also include the prefix specified in the properties file.

Commenting on an Issue

Adding comments to an issue is a useful way to record additional detail about an issue, and collaborate with team members. Comments are shown in the **Comments** tab of the **Activity** section when you view an issue.

Note:

- When adding a comment, you can set the comment to be **Viewable by** members of a particular project role or user group only; or you can allow all users to view it.
- For users to view a comment, they must have the **Browse Project** project permission to view the issue and for each comment, they must be a member of the **Viewable by** users (see Adding a Comment below).

On this page:

- Adding a comment
- Collapsing and expanding a comment
- Editing a comment
- Deleting a comment
- Linking to a comment

Adding a comment

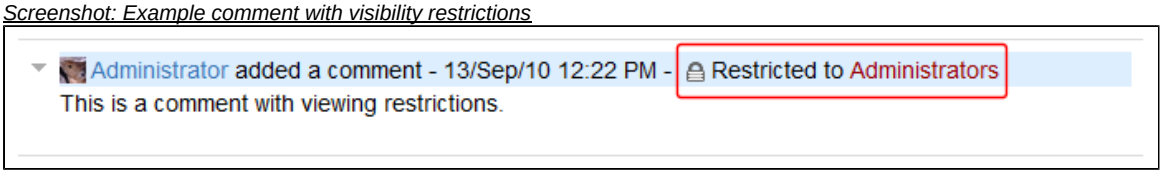
To add comments to an issue, i.e. to see the **Comment** button, you must have both of the following permissions for the issue's relevant project:

- **Browse Project** project permission — to view the issue to be commented on
- **Add Comments** project permission — to add a comment to the issue.

To add a comment:

1. Open the issue on which to add your comment.
2. Click the **Comment** button (located at either the top or bottom of the page).
   Keyboard shortcut: **m**
3. In the **Comment** text box, type your comment, using as many lines as you require. You can use wiki markup in this text box if you wish.
   - **To apply viewing restrictions to a comment:** click the open padlock icon next to **Viewable by...** and select which users will be able to view this comment.
     The **Viewable by...** list also includes all project roles and groups to which you belong. (Note that **All Users** means everybody who uses JIRA, while **Users** means everybody who is a member of the **Users** project role in this project.) Depending on how your JIRA administrator has configured 'Comment visibility', the **Viewable by...** list may include groups as well as project roles.
     Once viewing restrictions have been applied to a comment, the padlock icon closes and **Viewable by...** is replaced by **Restricted to...** and indicates the user, project role or group, who can view this comment, for example:
     Restricted to Users
   - **To email other users about your comment:** simply mention these users in the **Comment** text box. An email message will be sent to the user's email address (registered with their JIRA account) upon clicking the **Update** button. See Emailing an issue to users by mentioning them for details on the correct syntax.
4. Click the **Add** button to save the comment.

Screenshot: Example comment with visibility restrictions

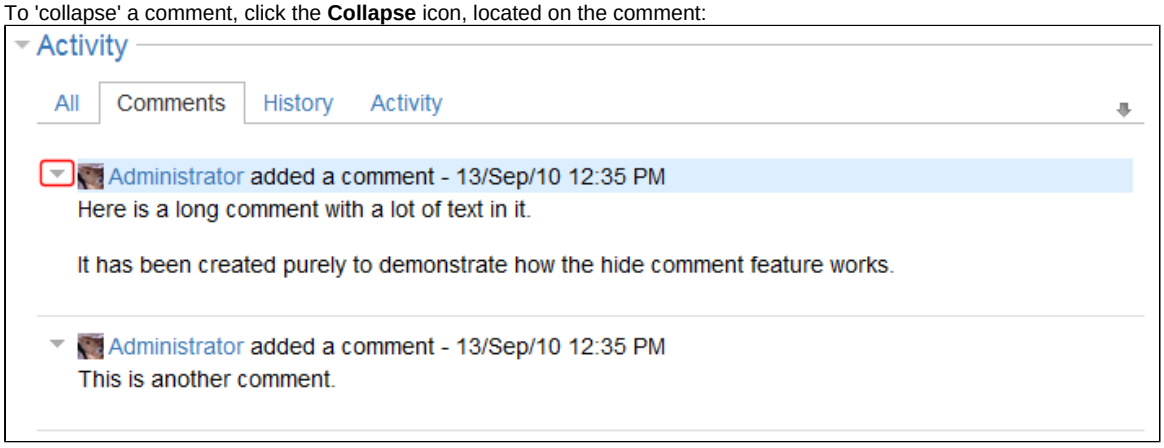


For each comment on an issue, a small version of the comment author's user avatar appears to the left of their full name.

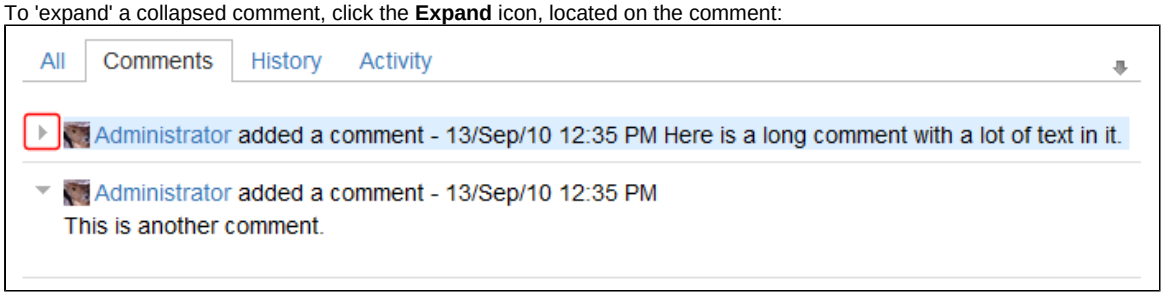
Collapsing and expanding a comment

To collapse or expand a comment:

1. Locate the comment in the **Activity** section at the bottom of the issue.
2. Browse to the comment you wish to collapse/expand.
3. To 'collapse' a comment, click the **Collapse** icon, located on the comment:



4. To 'expand' a collapsed comment, click the **Expand** icon, located on the comment:



'Collapsing' or 'Expanding' a comment does not relate in any way to a comment's 'viewing restrictions' applied when adding or editing a comment.

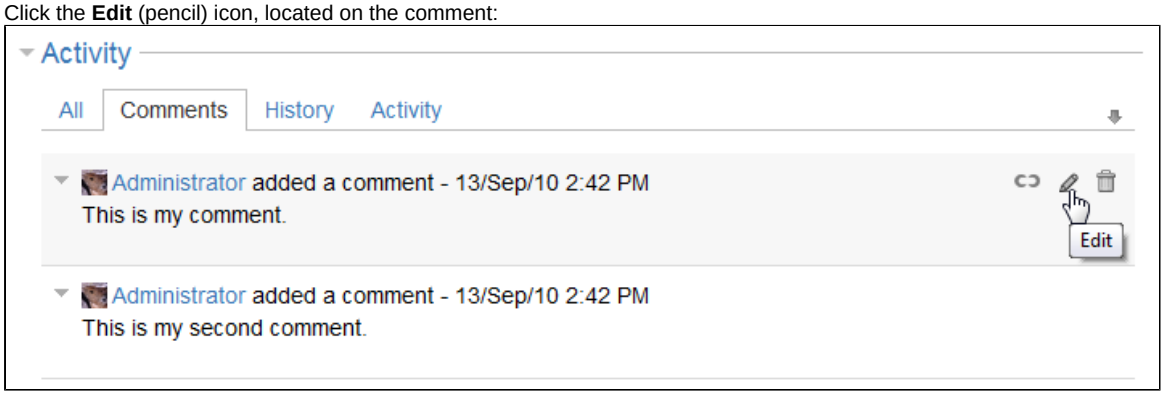
Editing a comment

You can edit your own comments if you have been granted the **Edit Own Comments** project permission.

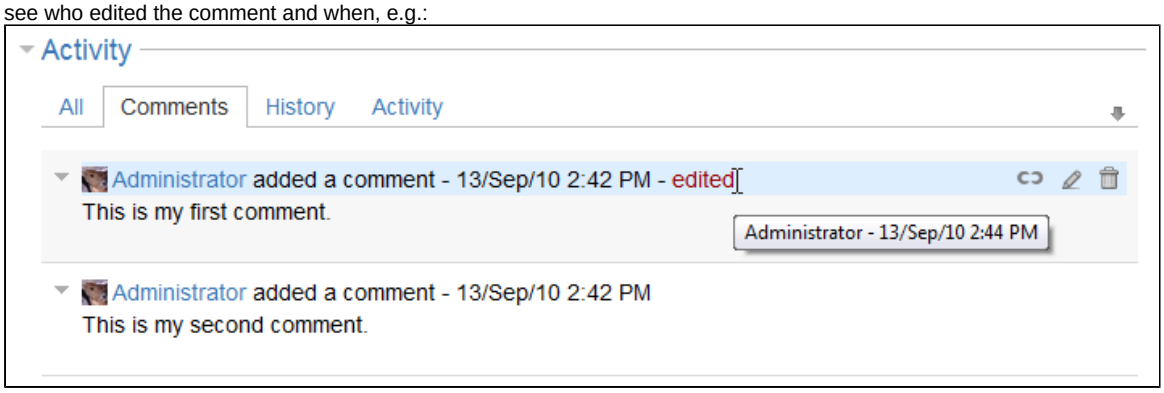
You can edit other people's comments if you have been granted the **Edit All Comments** project permission.

To edit a comment:

1. Locate the comment in the **Activity** section at the bottom of the issue.
2. Browse to the comment you wish to edit.
3. Click the **Edit** (pencil) icon, located on the comment:



4. Edit the comment's text and/or **Viewable by** list as required.
5. Click the **Save** button.
6. The word 'edited' will be displayed to indicate that the comment has been edited. You can hover your mouse over the word 'edited' to see who edited the comment and when, e.g.:



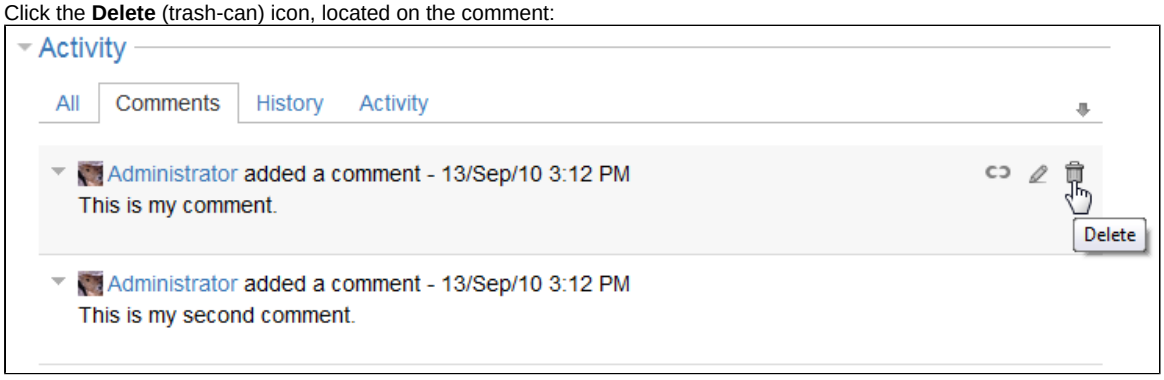
Deleting a comment

You can delete your own comments if you have been granted the **Delete Own Comments** project permission.

You can delete other people's comments if you have been granted the **Delete All Comments** project permission.

To delete a comment:

1. Hover your mouse over the comment you wish to delete.
2. Click the **Delete** (trash-can) icon, located on the comment:



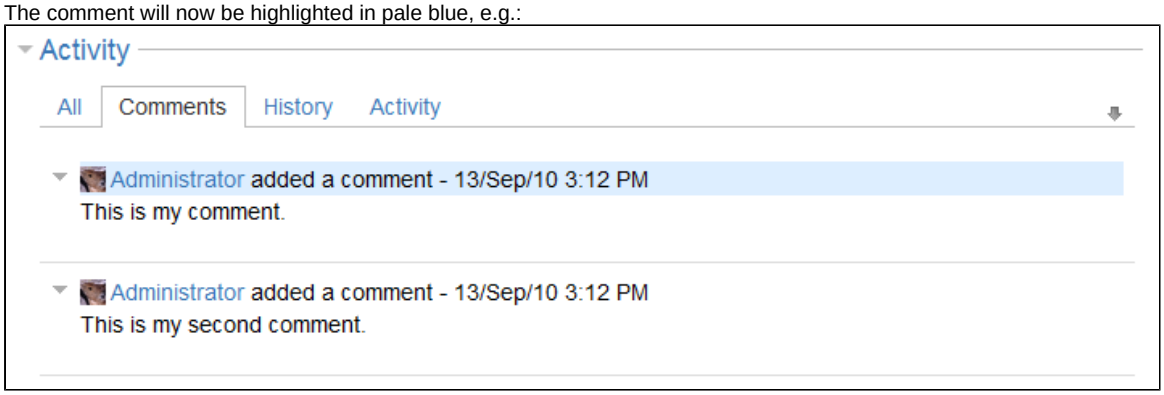
3. Confirm the deletion by clicking the **Delete** button.

Linking to a comment

Sometimes you may want to link to a specific comment within a JIRA issue. To do this,

1. Browse to the comment you wish to link to.
2. Click the **Permlink** (link) icon, located on the comment:

3. The comment will now be highlighted in pale blue, e.g.:



If your JIRA issue contains an extensive list of comments, the issue page will automatically be scrolled down so that the linked comment is visible.

4. The URL in your browser's address bar will now look something like this:

   http://jira.atlassian.com/browse/TST-123?focusedCommentId=94796#action_94796
5. Copy the URL from your browser's address bar and paste it into wherever you want to link from (e.g. an email).

Related topics

Emailing an Issue

Creating an Issue

To create a JIRA issue, you need the **Create Issue** project permission for the issue's relevant project. If you do not have this permission, please contact your JIRA administrator.

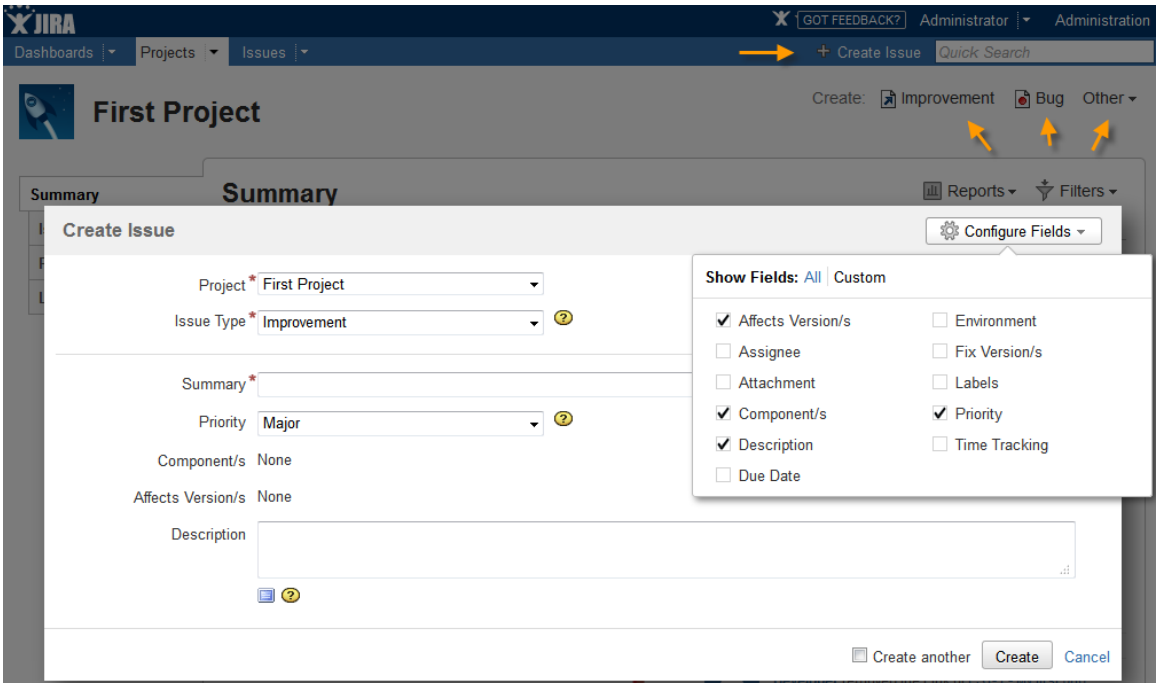
To create a new JIRA issue:

1. Click the **Create Issue** link (at the top-right of the JIRA user interface) to open the **Create Issue** dialog box.
   Keyboard shortcut: **c**
   While browsing a project, you can click one of the **issue type** icons such as **Bug**, **New Feature**, **Task** or **Improvement**.
2. Select the relevant **Project** and **Issue Type** on the **Create Issue** dialog box.
3. Type a **Summary** for the issue and complete any appropriate fields — at least required ones which are marked by an asterisk.
   If you want to access fields which are not shown on this dialog box or you want to hide existing fields:
   a. Click the **Configure Fields** button.
   b. Click **Custom** and select the fields you want to show or hide by selecting or clearing the relevant check boxes, respectively, or click **All** to show all fields.
      When you next create an issue, JIRA remembers your last choice of selected fields.
4. *(Optional)* If you want to create a series of similar issues (with the same **Project** and **Issue Type**), select the **Create another** check box.
5. Click the **Create** button to create the issue.
   If you selected the **Create another** check box (above), a new **Create Issue** dialog box will appear after your issue is created, automatically pre-populated with your previous issue details, while leaving the **Summary** field blank.

Tips:

- You can mention other users in the **Description** or **Comment** field so that an email message will be sent to the user's email address (registered with their JIRA account) upon clicking the **Update** button. See Emailing an issue to users by mentioning them for details.
- To see a list of all issues that you have created, which have not yet been resolved, go to your user name and select **Profile** and on your profile, click **Filters** > **Reported & Open**.
- With appropriate configuration by your JIRA administrator, it is also possible to create issues via email.

Screenshot: Example 'Create Issue' dialog box



Related topics

Sharing a Search Result

Creating a Sub-Task

Sub-task issues are useful for splitting up a parent issue into a number of tasks which can be assigned and tracked separately. Splitting issues into smaller tasks often provides a better picture of the progress on the issue, and allows each person involved in resolving the issue to better understand what part of the process they are responsible for.

All the sub-tasks related to a parent issue are summarised on the parent issue's main screen (see 'Working with Sub-Tasks' below). Sub-tasks always belong to the same project as their parent issue.

Sub-tasks have all the same fields as standard issues, e.g. Summary, Description, Reporter, Assignee, Status. Note that sub-tasks have a different set of issue types from the standard issue types.

Sub-tasks cannot have sub-tasks of their own. However, if you need to break up a sub-task into smaller sub-tasks, you could achieve this by first converting the sub-task to a standard issue. You would then be able to create sub-tasks for it.

On this page:

- Creating a sub-task
- What a sub-task looks like
- Working with sub-tasks
- Searching for sub-tasks
  - Adding the 'Sub-Tasks' column to your Issue Navigator
- Converting a standard issue to a sub-task
- Converting a sub-task to a standard issue

Creating a sub-task

To create sub-tasks, you will need to have the 'Create Issue' permission in the parent issue's project. Also note that sub-tasks can only be created if your JIRA administrator has enabled sub-tasks and added the sub-task issue type to the project's issue type scheme.

To create a sub-task:

1. Navigate to the issue which you would like to be the parent issue of the sub-task you are about to create.
2. From the '**More Actions**' drop-down menu, select '**Create Sub-Task**'. You will see the '**Create Subtask**' screen.
3. Fill in the details as needed, and then click the '**Create**' button at the bottom of the page.

Tip: You can customise the **'Create Subtask'** dialog to show fields you use most often. To do this, at the top right corner of the dialog, click the **'Configure Fields'** button, and use the **'All'** and **'Custom'** links to switch between the default screen and your custom settings. Your changes will be saved for future use.

Note that there is no option to set security on a sub-task, as sub-tasks inherit their parent issue's security levels if any have been set.

What a sub-task looks like

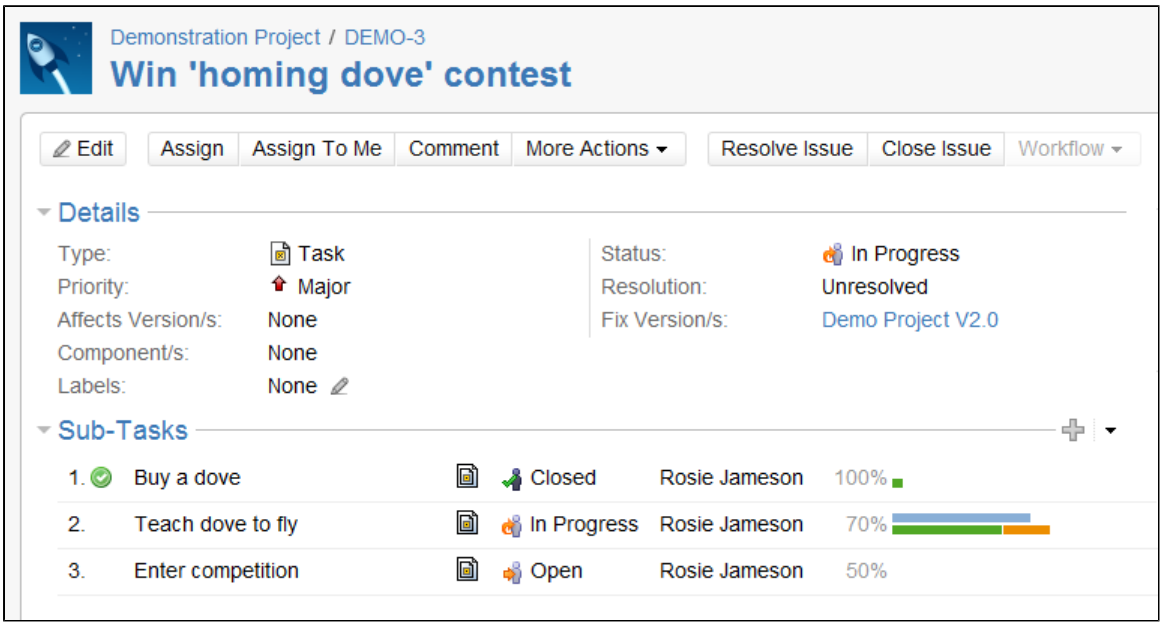
In the following screenshot, the sub-task is **DEMO-6 (Enter competition)** and its parent issue is **DOVE-3 (Win 'homing dove' contest).** Both belong to the **Demonstration** project.

Screenshot: Example sub-task

Working with sub-tasks

If an issue has sub-tasks, then the issue screen will show a list of all the issue's sub-tasks:

Screenshot: An Issue's Sub-Tasks



- **Show open sub-tasks only.** – The sub-task list has two *views:* 'All' and 'Open'. The *All* view lists all sub-tasks, regardless of status, while the *Open* view only shows sub-tasks that have not been resolved (i.e. do not have a Resolution). You can switch views by clicking the 'All' and 'Open' links.
- **Time-tracking**. – The coloured bars show the time-tracking data for the issue and its sub-tasks.
- **Reorder sub-tasks.** – You can reorder sub-tasks, for example, organise the list in the order of intended execution or priority. Hover your mouse over the sub-task you wish to move, and use the 'up' and 'down' arrows that will appear.
- **Perform actions on sub-tasks**. – To preform *actions* on the sub-tasks, e.g. 'Assign Issue', 'Resolve Issue', 'Close Issue', 'Reopen Issue', hover your mouse over the sub-task you wish to action, and click the 'Actions' drop-down menu link that appears.
- **Quickly create sub-tasks.** – Once an issue has one or more sub-tasks, you can quickly create additional sub-tasks by clicking the '+' icon.

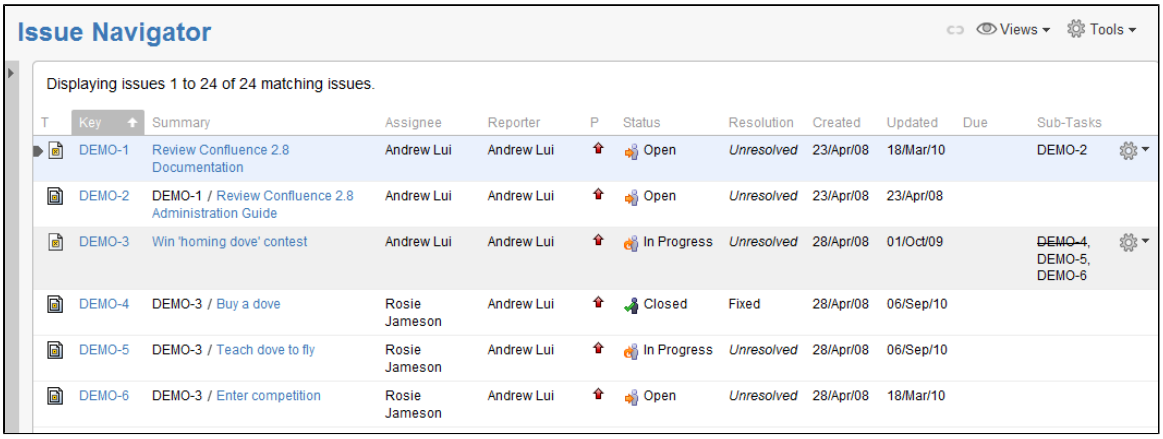
Searching for sub-tasks

When sub-tasks are enabled, two extra entries will appear in the 'Issue Type' drop-down list in the Issue Navigator's search form. These entries are: 'Standard Issue Types' and 'Sub-Task Issue Types'.

- To search standard issues only, click the *Standard Issue Types* entry.
- To search sub-task issues only, click *Sub-Task Issue Types* entry.
- To search for one specific type of issue or sub-issue, select just one Issue Type or one Sub-Task Issue Type.

If no entries are selected from the 'Issue Type' drop-down then the search will return all the standard issues *and* sub-task issues that meet the rest of the search criteria.

Once you have added the 'Sub-Tasks' column to your Issue Navigator, the search results indicate sub-task issues by displaying the parent issue's issue key above the sub-task's summary, as shown below:



Adding the 'Sub-Tasks' column to your Issue Navigator

To add the '**Sub-Tasks**' column to your Issue Navigator as shown in the above screenshot, please see Customising your Issue Navigator columns.

Converting a standard issue to a sub-task

1. Navigate to the issue which you would like convert to a sub-task.
2. From the '**More Actions**' drop-down menu, select '**Convert to sub-task**'.
3. The '**Step 1. Select Parent Issue and Sub-Task Type**' screen will be displayed. Type or select the appropriate parent issue type, select the new issue type (i.e. a sub-task issue type) and click the 'Next' button.
4. If the issue's current status is not an allowed status for the new issue type, the '**Step 2. Select New Status**' screen will be displayed. Select a new status and click the '**Next**' button.
5. The '**Step 3. Update Fields**' screen will be displayed. If the new issue type requires any additional fields, you will be prompted to enter them (otherwise you will see the message 'All fields will be updated automatically'). Click the '**Next**' button.
6. The '**Step 4. Confirmation**' screen will be displayed. If you are satisfied with the new details for the issue, click the '**Finish**' button.
7. The issue will be displayed. You will see that it is now a sub-task, that is, its parent's issue number is now displayed at the top of the screen (see 'Creating a sub-task' above).

Note that you will not be able to convert an issue to a sub-task if the issue has sub-tasks of its own. You will first need to convert the issue's sub-tasks to standalone issues (see below); you can then convert them to sub-tasks of another issue if you wish. Sub-tasks cannot be moved directly from one issue to another — you will need to convert them to standard issues, then to sub-tasks of their new parent issue.

Converting a sub-task to a standard issue

1. Navigate to the sub-task issue which you would like convert to a standard issue.
2. Select '**Convert to issue**' from the '**More Actions**' drop-down menu.
3. The '**Step 1. Select Issue Type**' screen will be displayed. Select a new issue type (i.e. a standard issue type) and click the '**Next**' button.
4. If the sub-task's current status is not an allowed status for the new issue type, the '**Step 2. Select New Status**' screen will be displayed. Select a new status and click the '**Next**' button.
5. The '**Step 3. Update Fields**' screen will be displayed. If the new issue type requires any additional fields, you will be prompted to enter them (otherwise you will see the message 'All fields will be updated automatically'). Click the '**Next**' button.
6. The '**Step 4. Confirmation**' screen will be displayed. If you are satisfied with the new details for the issue, click the '**Finish**' button.
7. The issue will be displayed. You will see that it is no longer a sub-task, that is, there is no longer a parent issue number displayed at the top of the screen.

Editing an Issue

To edit a JIRA issue, you need the **Edit Issue** project permission for the issue's relevant project. If you do not have this permission, please contact your JIRA administrator.

To edit an existing JIRA issue:

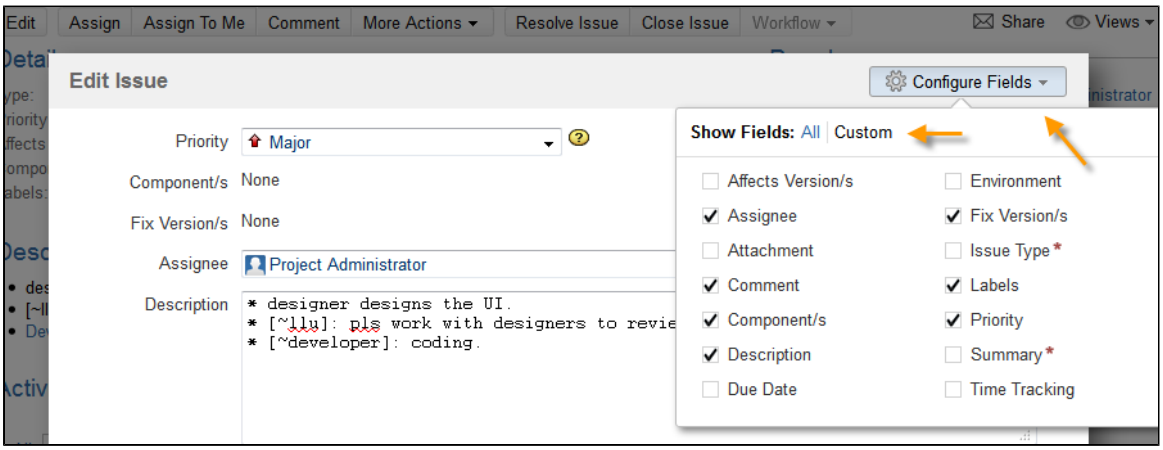
1. Locate the issue you want to edit.
2. Click the **Edit** button (at the top-left of the 'view issue' page) to open the **Edit Issue** dialog box.
   Keyboard shortcut: **e**
3. Modify your issue's details in the appropriate fields of the **Edit Issue** dialog box.
   If you want to access fields which are not shown on this dialog box or you want to hide existing fields:
   a. Click the **Configure Fields** button.
   b. Click **Custom** and select the fields you want to show or hide by selecting or clearing the relevant check boxes, respectively, or click **All** to show all fields.

When you next edit an issue for a given project, JIRA remembers your last choice of selected fields.

4. Click the **Update** button to save your changes.

Tip: You can mention other users in the **Description** or **Comment** field so that an email message will be sent to the user's email address (registered with their JIRA account) upon clicking the **Update** button. See Emailing an issue to users by mentioning them for details.

Screenshot: Customising the fields on the Edit Issue dialog



Related topics

Linking Issues

Emailing an Issue

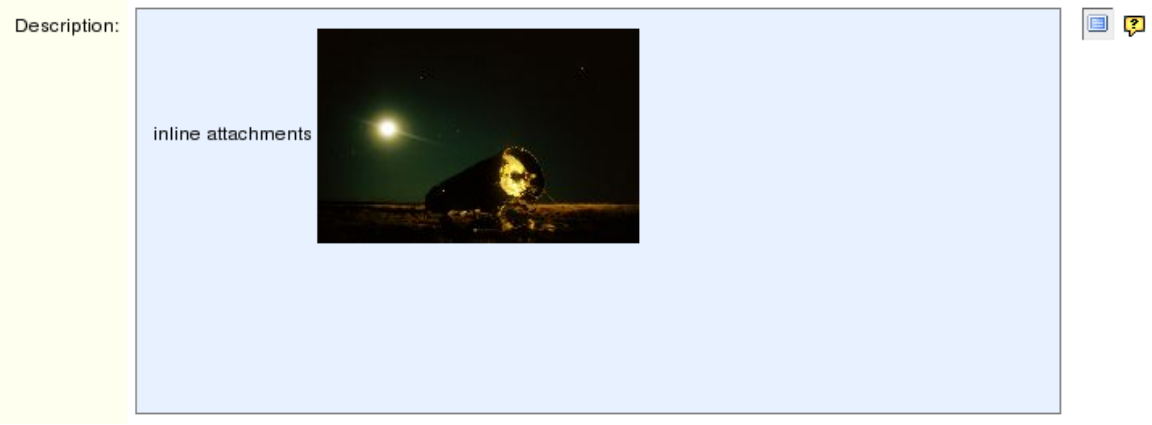
Editing Rich-Text Fields

When you create, edit or comment on a JIRA issue, some fields may display two small icons at the right of the text area: a blue screen (the **'Preview'** icon) and a yellow question-mark (the **'Help'** icon). The presence of these icons indicates that this field supports JIRA's Text Formatting Notation, allowing you to use rich-text features such as:

- Italic, bold, underlined text.
- Multiple levels of headings.
- Bullets, numbered lists, tables and quotations.
- Images.
- Macros (see below).

For example, to include an image in the field, you would first attach the image to the issue, then type the following into the field:

To preview what the field will look like after you save it, click the **'Preview'** icon:



Click the **'Help'** icon to see a popup window containing the Text Formatting Notation Help.

Using Macros

> The JIRA Text Formatting Notation and macros will only be available if your JIRA administrator has configured the relevant renderers.

JIRA ships with the following macros:

Macro	Description	Enabled by default

Anchor Macro	Create an anchor that allows people to link to a specific point in a page. Usage:
	`{anchor:bookmark1}`
	`... text here ...`
	`[#bookmark1]`
		yes

Code Macro	Format blocks of source-code or XML. The default language is Java but you can specify JavaScript, ActionScript, XML, HTML and SQL too. Usage:
	`--- Java example ---`
	`{code:title=Bar.java|borderStyle=solid}`
	`// Some comments here`
	`public String getFoo()`
	`{`
	`  return foo;`
	`}`
	`{code}`
	`*--- XML example ---*`
	`{code:xml}`
	`<test>`
	`  <another tag="attribute"/>`
	`</test>`
	`{code}`
		yes

Quote Macro	Generate blockquotes that may contain multiple paragraphs or complex markup. Usage:
	`{quote}`
	`This is text from another source`
	`{quote}`
		yes

No Format Macro	Create blocks of text where other wiki formatting is not applied. Usage:
	`{noformat}`
	`This is text  \#&@!@(*!&*@()#*@\ \\`
	`{macros} _wont_work_here`
	`{noformat}`
		yes

Panel Macro	Draw a panel with the following optional parameters:
	• title: Title of the panel
	• borderStyle: The style of the border this panel uses (solid, dashed and other valid CSS border styles)
	• borderColor: The color of the border this panel uses
	• borderWidth: The width of the border this panel uses
	• bgColor: The background color of this panel
	• titleBGColor: The background color of the title section of this panel
	`{panel}Some text{panel}`
	`{panel:title=My Title}Some text with a title{panel}`
	`{panel:title=My Title| borderStyle=dashed| borderColor=#ccc|`
	`titleBGColor=#F7D6C1| bgColor=#FFFFCE}`
	`a block of text surrounded with a *panel*`
	`yet _another_ line`
	`{panel}`
		true

Colour Macro	Change the colour of the contained text. Usage:
	`{color:red}`
	`look ma, red text!`
	`{color}`
	You can use any of the sixteen standard HTML colours ('aqua','black','blue','fuschia','green','gray','lime', 'maroon','navy','olive','purple','red','silver','teal','white','yellow'), or any hexadecimal colour code (e.g. '#336699').
		yes

Lorem Ipsum Macro	Insert paragraphs of "lorem ipsum" space-filler text. Usage: `{loremipsum}`	yes

HTML Macro	Use HTML code within a Jira Issue. Usage:
	`{html}`
	`<p>You'll find a lot more in <A href="chapter2.html">chapter two</a>.`
	`See also this <a href="../images/forest.gif">map of the enchanted`
	`forest.</A></p>`
	`{html}`
		no

Emailing an Issue

You can easily email other JIRA users a link to an issue either by sharing the issue with them or by mentioning them in an issue's **Description** or **Comment** field, when creating, editing or commenting on an issue.

✅ When 'sharing' an issue, you can also email the issue any email address — not only just JIRA users. This is useful when on JIRA sites where one or more projects can be accessed anonymously.

On this page:

- Sharing an issue
- Emailing an issue to users by mentioning them

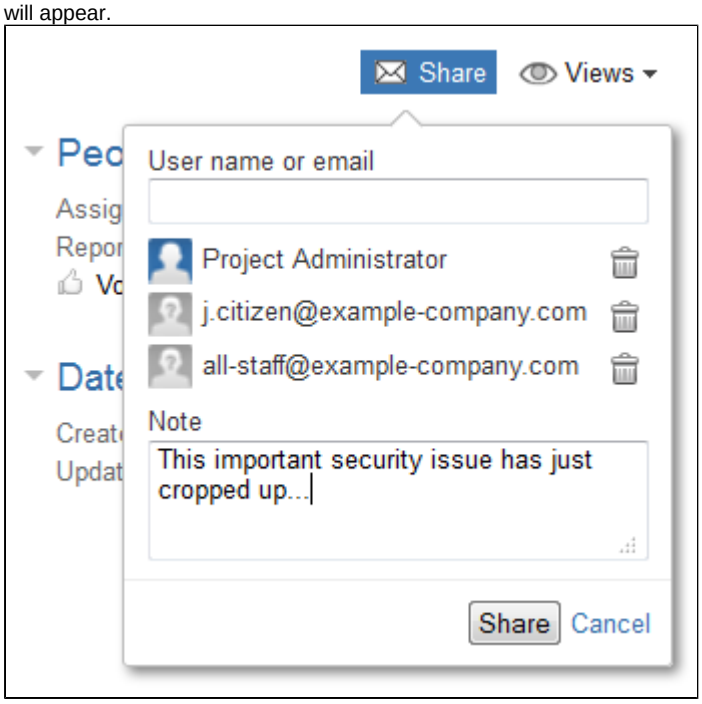
Sharing an issue

You can easily email other JIRA users (including any email address) a link to an issue by 'sharing' the issue with them. You can also add an optional note to the email message.

To access the **Share** button, your **JIRA System Administrator** must first have configured JIRA's SMTP mail server. Additionally, you also require the **Browse Users** global permission.

To share an issue with one or more JIRA users or any email addresses:

1. View the issue you want to share.
2. Click the **Share** button at the top-right.
   Keyboard shortcut: **s**
3. Specify JIRA users (by typing their usernames or part/all of their full names as registered with their JIRA user accounts) or type *any* email addresses of people you want to share the issue with.
   When you begin typing a JIRA user's username or name, or a previously specified email address, an autocomplete dropdown will appear.



4. Add an optional **Note**.
5. Click the **Share** button.

Please Note:

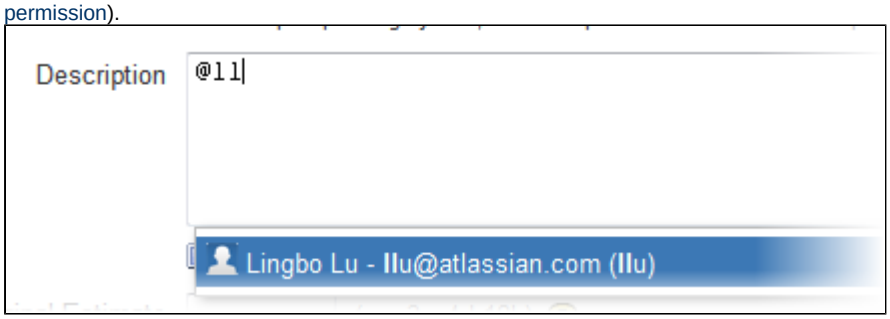
- Recipients specified in the **User name or email** field will receive an email message whose body contains the content of the **Note** (if one was specified) as well as a link to the issue.
- A shared issue sent to JIRA users specified in the **User name or email** field will be sent to the email addresses registered with these user's respective JIRA accounts.
- The subject line of the email message will specify you as the JIRA user who 'shared' the issue with the recipients.
- You can also share a search result from the issue navigator. For details, see Sharing a Search Result.

Emailing an issue to users by mentioning them

When creating, editing or commenting on an issue, you can easily email other JIRA users a link to the issue by mentioning these users in the issue's **Description** or **Comment** field.

To use this feature, your **JIRA System Administrator** must first have configured JIRA's SMTP mail server. You need the **Browse Users** global permission to access the 'suggested users' dropdown list when 'mentioning' a user. However, if you know the username of a JIRA user, you can still mention them without this permission.

To mention a user on an issue:

1. Create, edit or comment on an issue.
2. In the issue's **Description** or the **Comment** field, type '@' and then the first few characters of the JIRA user's username or part or all of their full name (registered with their JIRA user account).
   As you start typing, a list of suggested users will appear in a dropdown list below the field (if you have the **Browse Users** global permission).
   

3. Complete typing the JIRA user's username or choose from the list of suggested users in the dropdown list.
   When you submit the field, JIRA will send that user an email message indicating that you mentioned them on that issue.

Please Note:

- Upon submitting your **Description** or **Comment** field that mentions one or more JIRA users, an email message will be sent to the email addresses registered with these user's respective JIRA accounts.
- If either the **Description** or **Comment** field does not use the **Wiki Style Renderer**, you can still mention users (who will receive email notifications) but once the field is submitted, the users' usernames will not be converted to links leading to their profile pages. Instead, any mentioned users will appear 'literally' in wiki markup style, i.e. `[~username]`.
- JIRA will only send an email message to JIRA users upon each unique mention. Hence, if you edit an issue's **Description** or **Comment** and do not mention a user a second time, that user will not receive a notification once the field is submitted.
- When you mention users, they will not be automatically added to the watcher list. If you mention users who are already watchers of the issue, they receive one email notification only when you save the edits that have their names mentioned.

Related topics

Watching and Voting on an Issue

Sharing a Search Result

Labelling an Issue

On this page:

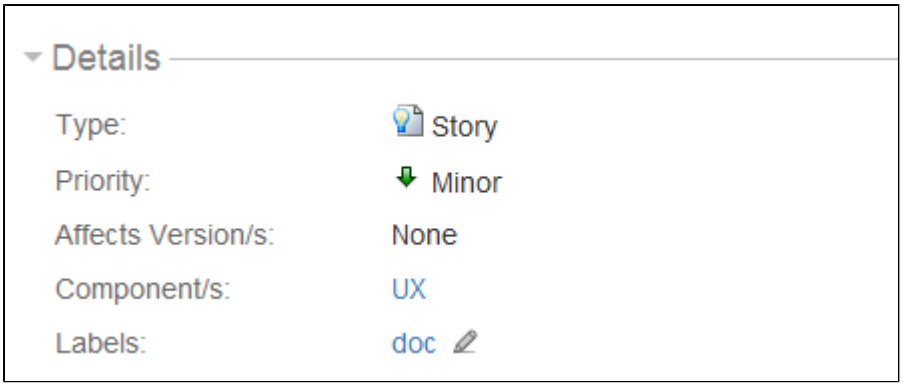
- About labels
- Adding a label to an issue
- Removing a label from an issue
- See also

About labels

Labelling allows you to categorise an issue(s) in a more informal way than assigning it to a version or component. You can then search for issues that have been given a particular label.

When viewing an issue, the issue's labels appear in the 'Details' section:

Screenshot: the 'Details' section within an issue



If your JIRA administrator has added any custom fields of type 'Label', they will be shown below the '**Labels**' field. For example, if you are using GreenHopper, a field called '**Epic**' will be shown (see the GreenHopper documentation on Working with Epics.

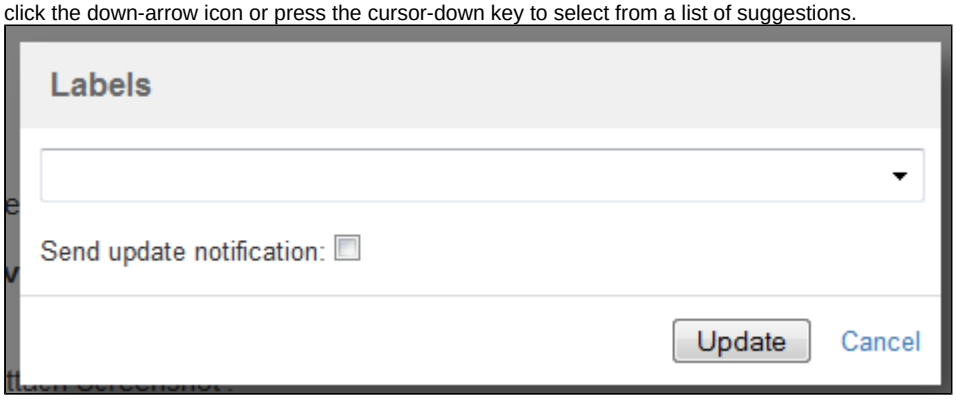
You can click a label (e.g. '**doc**' in the above screenshot) to jump to the Issue Navigator and see a list of all issues which have that label.

You can also:

- browse for labelled issues in a particular project (see Browsing a Project's Labels).
- search for issues that have been given a particular label (use the 'Labels' field in a Simple Search or an Advanced Search), and create saved filters.
- add the Labels Gadget to your dashboard, and/or use the 'Labels' field with any field-focused gadget (e.g. Heat Map, Issue Statistics , Filter Results, 2D Filter Statistics).

Adding a label to an issue

1. View the issue which you want to label.
2. Click the 'edit' icon which appears to the right of the word '**Labels**':
3. In the pop-up dialog box that appears, either:
    - type your label and press the **<Enter>** key, or
    - click the down-arrow icon or press the cursor-down key to select from a list of suggestions.



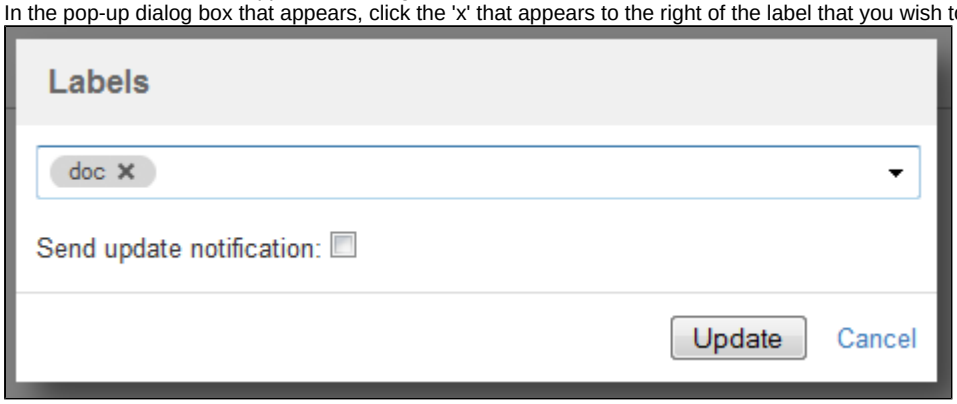
    You can type multiple labels, separated by spaces.
4. Normally when you edit and save an issue, certain people are notified via email.
    - If you want the normal notification email to be sent, select the '**Send update notification**' check-box.
    - If you don't want the normal notification email to be sent, leave the '**Send update notification**' check-box blank.

    Note that the issue's Change History will always be updated when a label is added.
5. Click the '**Update**' button. Your new label will be saved.

You can also add a label when you create or edit an issue.

Removing a label from an issue

1. View the issue which you want to label.
2. Click the 'edit' icon which appears to the right of the word '**Labels**':
3. In the pop-up dialog box that appears, click the 'x' that appears to the right of the label that you wish to remove.



4. Normally when you edit and save an issue, certain people are notified via email.
    - If you want the normal notification email to be sent, select the '**Send update notification**' check-box.
    - If you don't want the normal notification email to be sent, leave the '**Send update notification**' check-box blank.

Note that the issue's Change History will always be updated when a label is removed.

5. Click the '**Update**' button. Your change will be saved.

You can also remove a label when you edit an issue.

See also

- Browsing a Project's Labels
- Adding the Labels Gadget

Linking Issues

About issue linking

Issue linking allows you to create an association between two issues on either the same or different JIRA servers. For example:

- An issue may *relate to* another.
- An issue may *duplicate* another.
- An issue may *block* another.

(Your JIRA administrator can customise the types of links that you can create.)

Issue linking also allows you to:

- Create an association between a JIRA issue and a Confluence page.
- Link a JIRA issue to any other web page.

Within an issue, issue links look like this:

Screenshot: the 'Issue Links' section within an issue

Note that resolved issues will be shown in strike-through font, e.g. ~~DEMO-1~~.

On this page:

- About issue linking
- Creating a link to another issue
- Creating a link to an issue on another JIRA site
- Creating a link to a Confluence page
- Creating a link to any web page URL
- Deleting a link
- Searching for linked issues

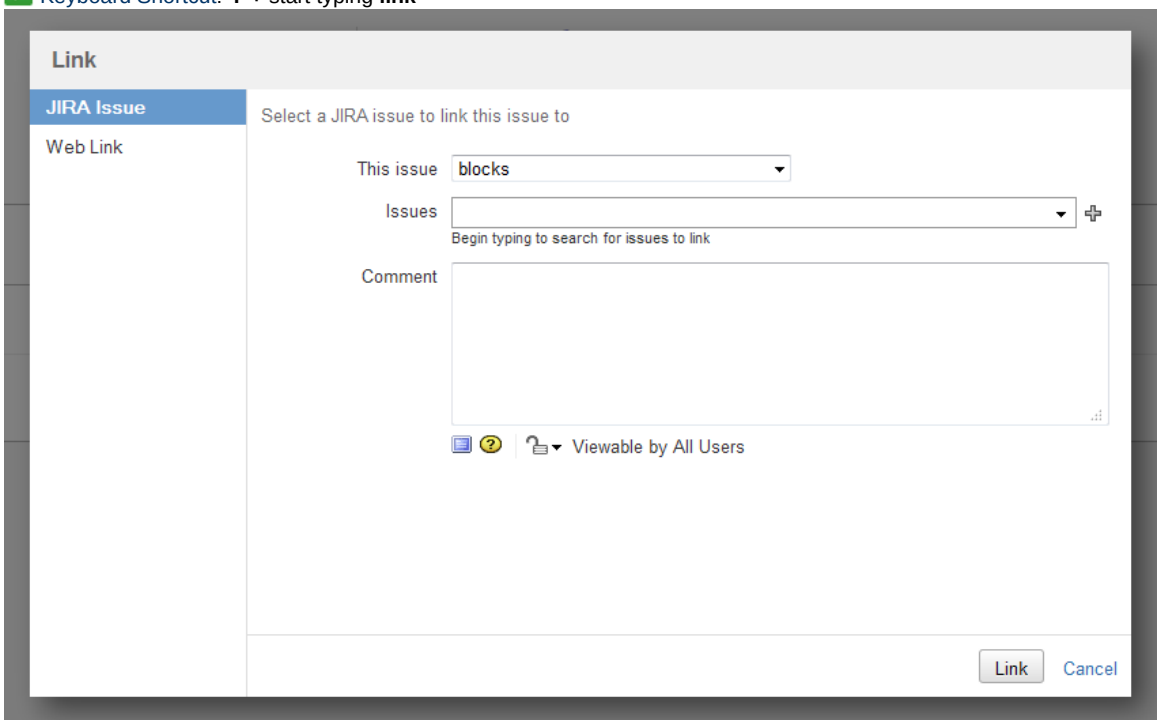
To create links on JIRA issues, you will need to have the Link Issues permission in the project(s) to which the issues belong.

Creating a link to another issue

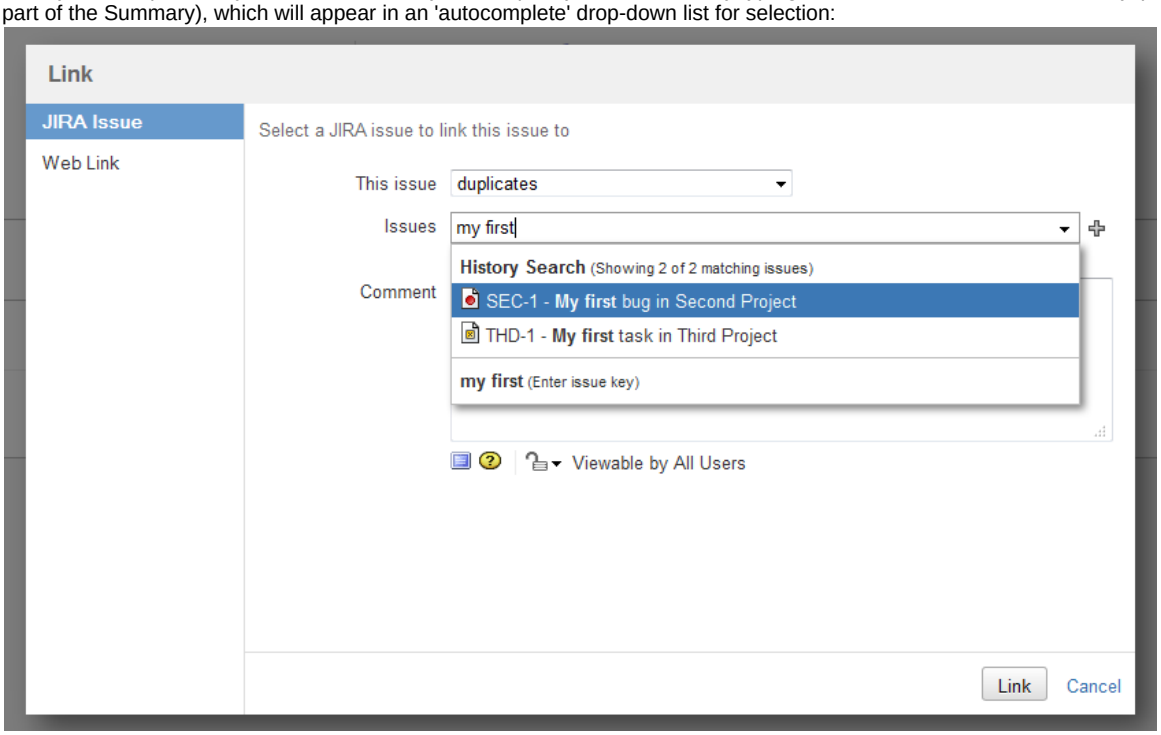
To create a link to another issue:

1. View the issue from which you want to create a link. Alternatively, move to the this issue on the issue navigator.
2. On the 'view issue' page, select **Link** from the **More Actions** menu or click the **Link** (plus) icon at the right of the **Issue Links** section. Otherwise, using the issue navigator, click the **Actions** (cog) icon to the right of this issue. The **Link** dialog box will be

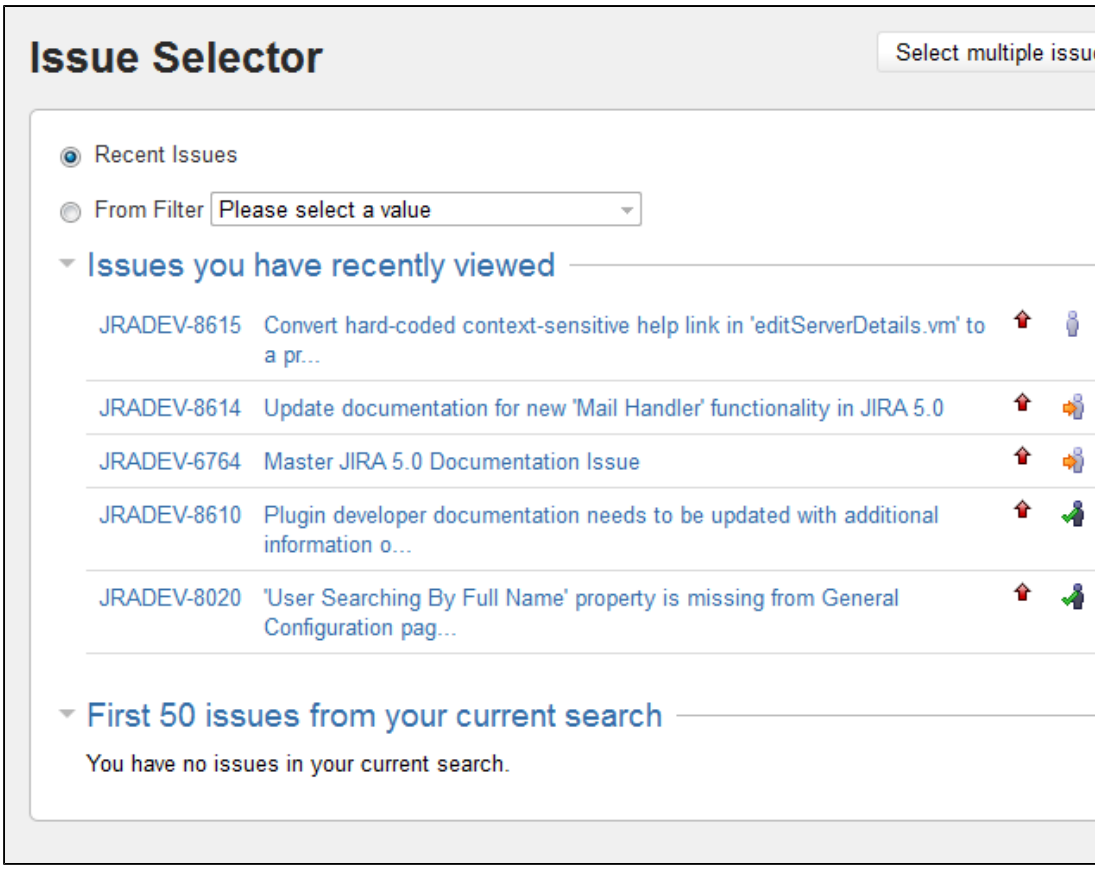
displayed.
Keyboard Shortcut: **'.' +** start typing **link**



3. Ensure that the **JIRA Issue** item is selected at the left of the dialog box and then choose the type of link to be created.
4. In the **Issues** field, specify the issue(s) to be linked to your currently viewed/selected issue. There are two ways to do this:
   - Type the full issue key (e.g. **ABC-123**) — or to link to multiple issues, press the 'Enter' key between each typed issue key.
     If you have previously browsed an issue, you can quickly find the issue by typing the first few letters of the issue key (or part of the Summary), which will appear in an 'autocomplete' drop-down list for selection:



   OR:
   - Click the **Select Issue** (plus) icon to use the **Issue Selector** popup, which allows you to select either from issues recently viewed, or from issues returned from your chosen saved search filter (click the **Please select a value**drop-down to choose a saved search filter).
     - To select a single issue, click the issue key or summary.



- To select multiple issues:
    a. Click the **Select multiple issues** button. Check boxes will appear next to each issue.
    b. Select the check boxes of the issues you want to link to.
    c. Click the **Select issues** button to close the popup and return to the **Link Issue** form.

5. Type an optional **Comment** - for example, to describe more information about why you are creating these links.
6. Click the **Link** button.

Creating a link to an issue on another JIRA site

To create this type of link, your JIRA system administrator ideally needs to have configured *fully reciprocal application links* between your JIRA site and the other JIRA site containing the issue(s) you want to link to.

To create a link to an issue on another JIRA site:

1. View the issue from which you want to create a link. Alternatively, move to the this issue on the issue navigator.
2. On the 'view issue' page, select **Link** from the **More Actions** menu or click the **Link** (plus) icon at the right of the **Issue Links** section. Otherwise, using the issue navigator, click the **Actions** (cog) icon to the right of this issue. The **Link** dialog box will be displayed.
   Keyboard Shortcut: **'.' +** start typing **link**
3. Click the **Remote JIRA Issue** option at the left of the dialog box and then choose the type of link to be created.
   Note:
   - This option will not be available if your JIRA system administrator has not configured an application link between your JIRA site and the remote JIRA site.
   - If, after selecting this option, you are prompted for authorisation, you may be required to log in to the remote JIRA site, which will allow your JIRA site to access the remote JIRA site *on behalf of your account on the remote JIRA site*.
     This behaviour means the application links configured between your JIRA site and the remote JIRA site use OAuth authentication.
4. Type the issue key of the issue on the remote JIRA site that you want to link to.
   Note:
   - You can link to any issue on the remote JIRA site to which you have access on that site.
   - This field does not use 'autocompletion'. Hence, you need to know the issue's issue key beforehand to create this link.
5. Select the **Create reciprocal link** check box to create the complementary link on the remote issue you are linking to, back to your issue. For example, if you create a **blocks** link type to a remote issue, the reciprocal link generated on the remote issue will be a **is blocked by** link type back to your local issue.
6. *(Optional)* Type a **Comment**, for example, to describe more information about why you are creating this link.
7. Click the **Link** button.

Troubleshooting

Problem: If you selected the **Create reciprocal link** check box but after clicking the **Link** button, discover that a reciprocal link from the remote JIRA issue back to your JIRA issue has not been created, then your JIRA system administrator has most likely created only a

one-way link from your JIRA site to the remote JIRA site.

Solution: Ask your JIRA system administrator to configure *fully reciprocal application links* between your JIRA site and the remote JIRA site.

Problem: If you attempted to create a reciprocal link but received the following message:

'**A reciprocal link from issue 'XYZ-123' back to this issue was not created as the remote JIRA server returned the following error: No Link Issue Permission for issue 'XYZ-123'.**' (where 'XYZ-123' is the issue key on the remote JIRA site),

then a reciprocal link on the remote JIRA site will not have been created, because the user account through which you authenticated on the remote JIRA site (at step 3 above) does not have the Link Issues project permission.

Solution:

- Ask the JIRA project administrator(s) on the remote JIRA site to grant your user account the Link Issues project permission for the relevant project(s) to which you need to create issue links.
- Alternatively, if the application link between your JIRA site and the remote JIRA site use OAuth authentication and you suspect you may have authenticated on the remote site with another user account that does not have the Link Issues project permission, repeat the procedure above but during the authorisation step (at step 3), authenticate on the remote site with a user account which has this permission.
  If you are not prompted for authentication during authorisation, try clearing your browser's cookies first and repeat the procedure again.

Creating a link to a Confluence page

To create this type of link, your JIRA system administrator needs to have configured an *application link* between your JIRA site and the Confluence site containing the pages you want to link to.

To create a link to a Confluence page:

1. View the issue from which you want to create a link. Alternatively, move to the this issue on the issue navigator.
2. On the 'view issue' page, select **Link** from the **More Actions** menu or click the **Link** (plus) icon at the right of the **Issue Links** section. Otherwise, using the issue navigator, click the **Actions** (cog) icon to the right of this issue. The **Link** dialog box will be displayed.
   Keyboard Shortcut: **'.' +** start typing **link**
3. Click the **Confluence Page** option at the left of the dialog box.
   This option will not be available if your JIRA system administrator has not configured an application link between your JIRA site and the Confluence site.
4. If more than one application link has been configured between your JIRA site and other Confluence sites, then choose the appropriate Confluence site from the **Server** dropdown list.
5. Specify the Confluence page to be linked to your currently viewed issue. There are two ways to do this:
   - In the **Page URL** field, enter the URL of a page on the Confluence site you want to link to. For example:

     ```
     http://<confluence-server>/display/ds/Welcome+to+the+Confluence+Demonstration+Space
     ```

   - Click the **search for a page** link. The **Link** dialog box is replaced by the **Find a Confluence page** dialog box.
     If you are prompted for authorisation, you may be required to log in to the Confluence site, which will allow your JIRA site to access the Confluence site *on behalf of your account on the Confluence site*.
     This behaviour means the application links configured between your JIRA site and the remote Confluence site use OAuth authentication.
     a. In the first **Search** field, specify one or more search terms that appear in the page you want to link to. This field is mandatory.
     b. *(Optional)* In the second **Search** field, select the Confluence space to further narrow down the search.
     c. Click the **Search** button and then the title of the page you want to link to.
6. *(Optional)* Type a **Comment**, for example, to describe more information about why you are creating this link.
7. Click the **Link** button.

Troubleshooting

Problem: If Confluence page links you create show **Failed to load** on the JIRA issue or if you attempted to search for a Confluence page but received the following message:

'**Content on the Confluence site could not be accessed because the Confluence server's 'Remote API' feature is disabled. The Confluence system administrator must enable this 'Remote API' feature for JIRA to successfully access this content.**'

then JIRA was unable to communicate with the Confluence server to either:

- retrieve information about the link or
- conduct a Confluence page search in the 'Find a Confluence page' dialog box.

Solution:

Ask the Confluence system administrator to enable the **Remote API (XML-RPC & SOAP)** feature, since this Confluence feature is disabled by default. See Enabling the Remote API in the Confluence documentation for details.

Creating a link to any web page URL

To create a link to any web page URL:

1. View the issue from which you want to create a link. Alternatively, move to the this issue on the issue navigator.
2. On the 'view issue' page, select **Link** from the **More Actions** menu or click the **Link** (plus) icon at the right of the **Issue Links** section. Otherwise, using the issue navigator, click the **Actions** (cog) icon to the right of this issue. The **Link** dialog box will be displayed.
   Keyboard Shortcut: **'.' +** start typing **link**
3. Click the **Web Link** option at the left of the dialog box.
4. Specify the **URL** of the web page you want to link to.
5. Specify the **Link Text** that will appear in the **Issue Links** section of the 'view issue' page and will be hyperlinked to your URL.
6. *(Optional)* Type a **Comment**, for example, to describe more information about why you are creating this link.
7. Click the **Link** button.

Deleting a link

To delete a link:

1. Go to an issue that contains links, and locate the **Issue Links** section (see screenshot above).
2. Hover your mouse over the link you wish to delete, and click the **Delete** (trashcan) icon that appears.

Searching for linked issues

You can search for issues that are linked to a particular issue. For details, please see the documentation on Advanced Searching.

Be aware that this functionality does not extend to issues on a remote JIRA server.

Logging Work on an Issue

On this page:

- About time-tracking
- Specifying time estimates
- Logging work on an issue
  - Logging work when viewing an issue
  - Logging work while resolving or closing an issue
- Editing a work log entry
- Deleting a work log entry
- Customised JIRA installations
  - Logging work *and/or* specifying time estimates on the same JIRA screen
- See also

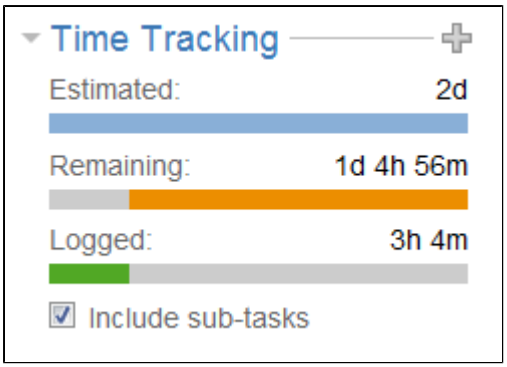
About time-tracking

> You can only specify time estimates and log work on an issue if your JIRA administrator has granted you the **Work On Issues** permission in the project to which the issue belongs. **Note:** Anyone with the **Browse Project** permission can view an issue's time tracking information.

If an issue (or its sub-tasks) has had work logged and/or an **Original Estimate** value specified, three coloured bars will be displayed representing the following amounts of time:

- **Original Estimate** (blue) — the amount of time originally anticipated to resolve the issue. (This is indicated as **Estimated** when viewing an issue.)
- **Remaining Estimate** (orange) — the remaining amount of time currently anticipated to resolve the issue. (This is indicated as **Remaining** when viewing an issue.)
- **Time Spent** (green) — the amount of time logged working on the issue so far. (This is indicated as **Logged** when viewing an issue.)

Screenshot: the Time Tracking section of an issue



If your issue has one or more sub-tasks, then:

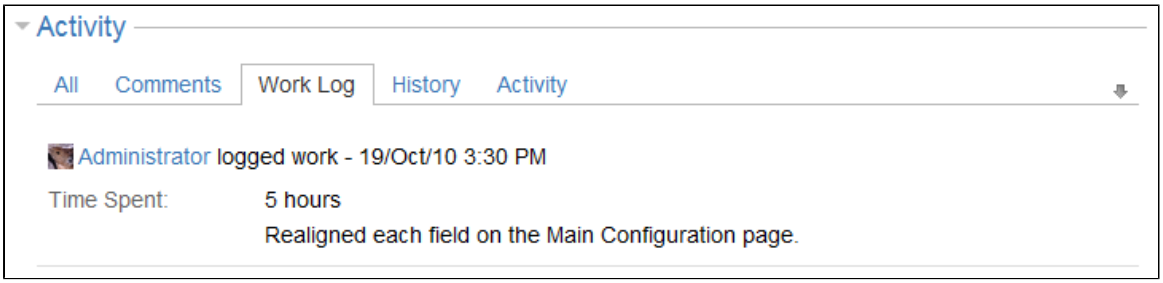
- To see aggregated times for your issue plus all of its sub-tasks, ensure that the **Include sub-tasks** check box is selected.
- To see times for just your issue only, ensure that the **Include sub-tasks** check box is cleared.

When you *log work* on an issue (see below), you:

1. Log the time you have spent, in weeks/days/hours/minutes (you can use fractions if you wish, e.g. `'5.5h'`).
   This time will be added to the issue's total **Time Spent**.
2. Enter a description of the work you have done.
3. Adjust the **Remaining Estimate** value (i.e. the *remaining amount of time* you think the issue will take to resolve).

The work logged on an issue is shown in the **Work Log** tab of the **Activity** section when you view an issue:

Screenshot: an issue's Work Log



Additionally, once work has been logged on an issue, various reports based on the time-tracking information become available.

Specifying time estimates

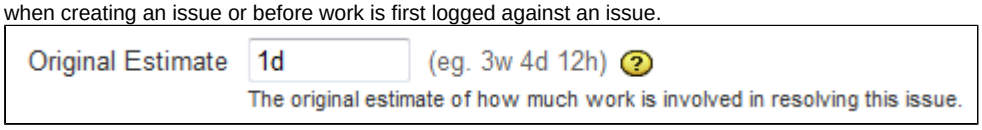
Prior to logging work on an issue, you may want to specify an **Original Estimate** for an issue (i.e. the *total amount of time* you think it will take to resolve the issue).

When work is first logged against the issue, the **Time Spent** is subtracted from the **Original Estimate** and the resulting value is automatically presented in the **Remaining Estimate**. When subsequent work is logged, any **Time Spent** is subtracted away from the **Remaining Estimate**.

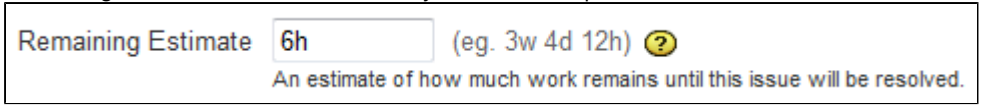
Throughout the lifecycle of an issue, however, you can modify the **Original Estimate** and/or **Remaining Estimate** values manually if you wish. To do this:

1. Navigate to an existing issue, view it and click the '**Edit**' button at the top of the screen to edit that issue.
   OR
   Create a new issue by clicking '**Create Issue**' at the top right of the screen and complete the required field details.

   Edit the following Time Tracking fields:
   - **Original Estimate** — the amount of time you originally believe is required to resolve the issue. Typically, this is specified when creating an issue or before work is first logged against an issue.

   - **Remaining Estimate** — the amount of time you believe is required to resolve the issue in its current state.

You may see only one of these fields if JIRA's Time Tracking feature is running in Legacy Mode.

If JIRA's Time Tracking feature is in Legacy Mode and work has not yet been logged on the issue, you will see the **Original Estimate** field. However, once work has been logged, you will only see the **Remaining Estimate** field.

2. Enter or modify your time tracking details in the **Original Estimate** and/or **Remaining Estimate** fields. Use 'w', 'd', 'h' and 'm' to specify weeks, days, hours or minutes. For example, to specify 'six hours', type '6h'.
   If both of these fields are available and are mandatory (indicated by red asterisks), you can just enter one value and leave the other value blank. When you submit the form, the field with a value will be copied to the field that was left blank.
3. Click the '**Update**' button at the bottom of the screen.

If you cannot change **Original Estimate** values on issues after work has been logged on them but you wish to do so, request that your JIRA administrator disables Legacy Mode on Time Tracking.

If your JIRA administrator has added the ability to specify/modify time estimates on other workflow transition screens in JIRA's default workflow (or a customised workflow), you will be able to specify/modify time estimates during those workflow transitions too. Additionally, if your JIRA administrator has added this ability to JIRA screens used by other issue types, you can also specify time estimates when creating or editing these types of issues.

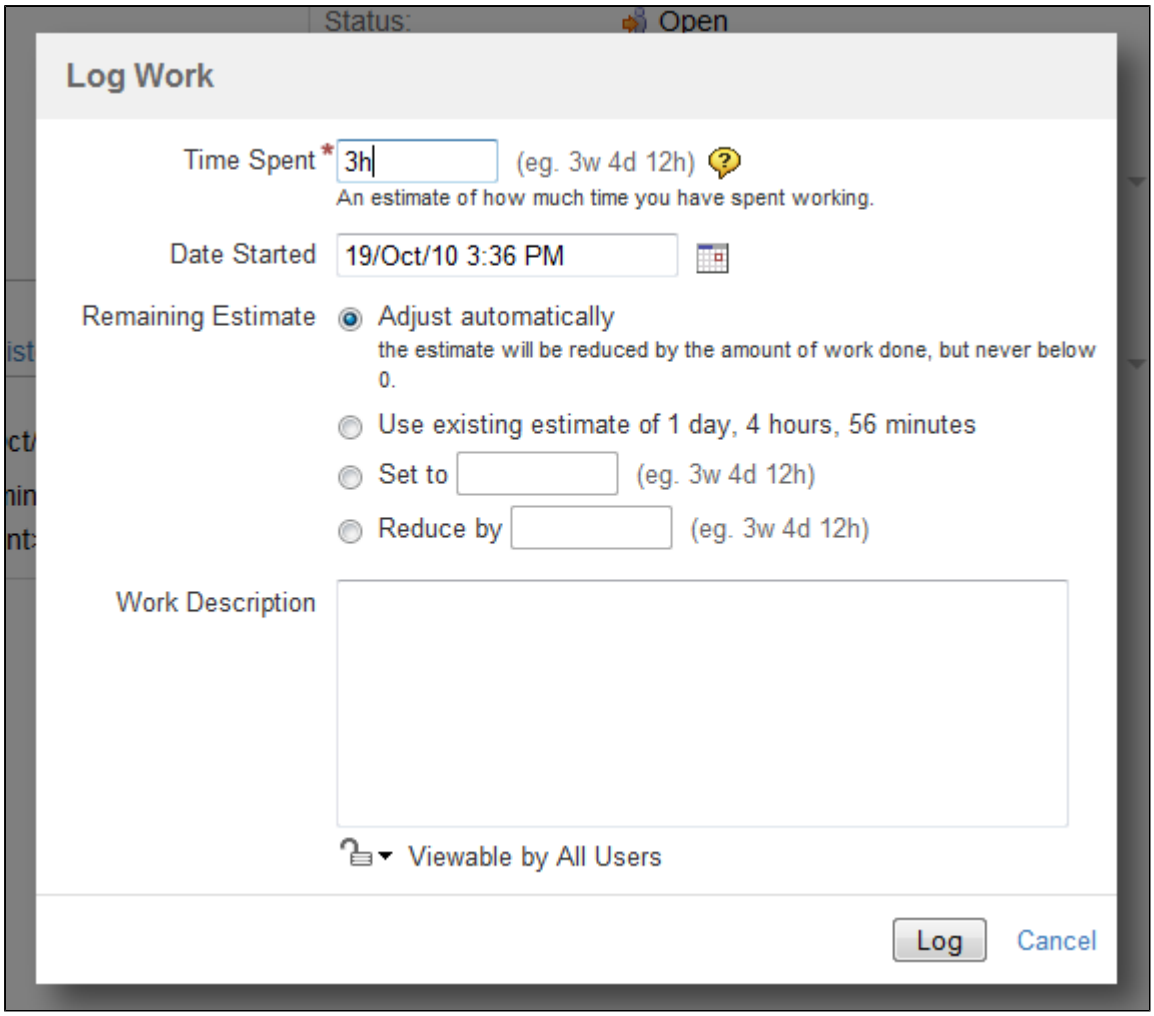
Logging work on an issue

Logging work when viewing an issue

To log work when viewing an issue:

1. Navigate to the issue and view its details.
2. Select '**Log Work**' from the **More Actions** drop-down menu.
3. The **Log Work** screen in a dialog box will be displayed.

   The Log Work Dialog Box



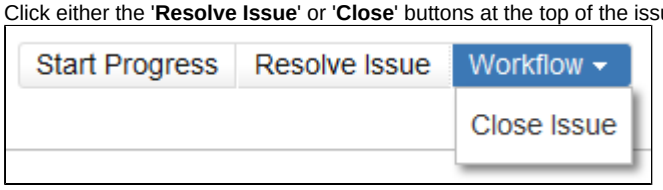
4. In the **Time Spent** field, enter the amount of time to be logged. Use 'w', 'd', 'h' and 'm' to specify weeks, days, hours or minutes. For example, to log two hours of work, type '2h'.
   If you type a number without specifying a time unit (e.g. if you type '2' instead of '2h'), the default time unit as specified by your JIRA administrator will apply.
5. In the **Date Started** field, click the calendar icon to select the date/time when you started work. The calendar popup will be displayed, where you can:
   - use the month scroll back ('<') or forward ('>') icons to choose a different month.
   - use the year scroll back ('<<') or forward ('>>') icons to choose a different year.
     If you click and hold one of these scroll icons, a dropdown menu appears, allowing you to select a month or year from the list of options provided.
   - click the hour to increase it (or <Shift> click to decrease it).
   - click the minute to increase it (or <Shift> click to decrease it).
   - click 'am' / 'pm' to toggle between them.
6. The **Remaining Estimate** field affects the value of the issue's **Remaining Estimate** value. Select one of the following:
   - '**Adjust Automatically**' — Select this if you want to automatically subtract your **Time Spent** from the issue's current **Remaining Estimate** value.
   - '**Leave Estimate Unset**' — Select this if you do not want to specify any time estimates for the issue.
     This option is only displayed if no time estimates have been specified.
   - '**Use existing estimate of ...**' — Select this if you do not want to change the issue's **Remaining Estimate** value.
     This option is displayed if a time estimate has been specified.
   - '**Set to ...**' — Select this if you want to manually set the issue's **Remaining Estimate** value to the amount specified. If you select this option, enter your new estimate into the blank field below. Use 'w', 'd', 'h' and 'm' to specify weeks, days, hours or minutes. For example, to specify 'thirty minutes', type '30m'.
   - '**Reduce by ...**' — Select this if you want to manually reduce the issue's **Remaining Estimate** value by the amount specified. If you select this option, enter your new estimate into the blank field below. Use 'w', 'd', 'h' and 'm' to specify weeks, days, hours or minutes. For example, to specify 'thirty minutes', type '30m'.
7. In the **Work Description** field, type a description or comment about the work you have done.
8. Click the padlock icon to either set this work log to be viewable only by members of a particular project role or group; or you can allow all users to view it.
   For users to view a work log, they must have the **Browse Project** permission to view the issue and be a member of the project role or group selected in this step.
9. Click the **'Log'** button to return to the issue, and verify that:
   - the **Time Spent** that you just entered has been added to the issue's total **Time Spent** field.
   - the **Remaining Estimate** value that you just entered (or chose) matches the issue's **Remaining Estimate** field.

Logging work while resolving or closing an issue

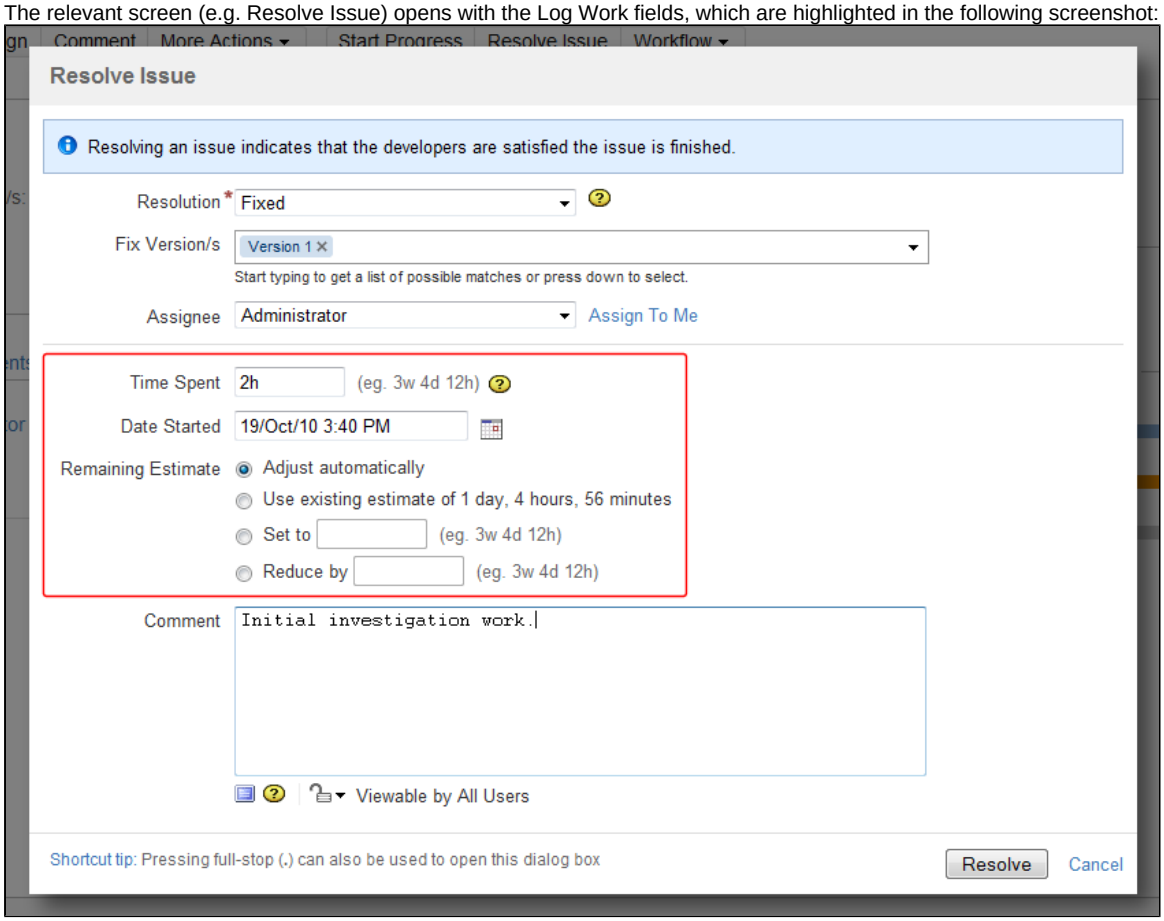
In addition to logging work when viewing an issue, you can also log work when resolving or closing an issue.

To log work on an issue while resolving or closing the issue:

1. Navigate to the issue and view its details.
2. Click either the '**Resolve Issue**' or '**Close**' buttons at the top of the issue:



The relevant screen (e.g. Resolve Issue) opens with the Log Work fields, which are highlighted in the following screenshot:



3. Edit the Log Work fields as described under the Logging work when viewing an issue section above.

> By default, JIRA should automatically copy the contents of your **Comments** to the **Workflow Description**. In doing so, the work log will be visible to members of the project role or group selected in the padlock icon dropdown.
>
> If this is not happening and you would like comments to be copied to workflow descriptions, request that your JIRA administrator enables JIRA's Copy Comments to Workflow Descriptions setting.
>
> If this setting is disabled:
>
> - The work log entry may be visible to anyone. If this is a concern, you need to edit this work log entry after creating it to modify its visibility.
> - Copying comments to workflow descriptions must be done manually after logging work.

4. Click the '**Resolve**' button (or the appropriately named button for your workflow transition or action) to return to the issue and verify that:
   - The **Time Spent** that you just entered has been added to the issue's total **Time Spent** field.
   - The **Remaining Estimate** value that you just entered (or chose) matches the issue's **Remaining Estimate** field.

If your JIRA administrator has added the ability to log work on other workflow transition sreens in JIRA's default workflow (or a customised workflow), you will be able to log work during those workflow transitions too. Additionally, if your JIRA administrator has if your JIRA administrator has added this ability to JIRA screens used by other issue types, you can log work when creating or editing these types of issues.

Editing a work log entry

1. Navigate to the issue and view its details.
2. Locate the **Activity** section and select the '**Work Log**' tab.
3. Locate the work log entry you wish to edit.
4. Click the '**Edit**' icon, located at the right of the work log entry.
5. The **Edit Work Log** screen will be displayed. Edit the fields as described under the Logging work when viewing an issue section above.
   Be aware that when you are editing a work log entry, you cannot specify an amount of time by which to reduce the Remaining Estimate. Hence, the '**Reduce By...**' field is not available on the **Edit Work Log** screen.
6. Click the '**Log**' button to return to the issue, and verify that:
   - the word **edited** is displayed to indicate that the work log entry has been edited. You can hover your mouse over this word to see who edited the work log and when this was edited.
   - the issue's total **Time Spent** field has been adjusted as per the **Time Spent** that you just edited.
   - the issue's **Remaining Estimate** value field has been adjusted as per the **Remaining Estimate** value that you just edited.

You can edit your own work log entries if you have been granted the **Edit Own Work Logs** permission. You can edit other people's work log entries if you have been granted the **Edit All Work Logs** permission.

Deleting a work log entry

1. Navigate to the issue and view its details.
2. Locate the **Activity** section and select the '**Work Log**' tab.
3. Locate the work log entry you wish to delete.
4. Click the '**Delete**' icon, located at the right of the work log entry.
5. Confirm the deletion by clicking the **'Delete'** button when prompted.
6. The **Delete Worklog** screen will be displayed. The **Adjust Estimate** field affects the value of the issue's **Remaining Estimate**. Select one of the following:
   - **'Auto Adjust'** — Select this if you want to automatically *add* the work log entry's **Time Spent** value back to the issue's current **Remaining Estimate** value.
   - **'Leave existing estimate of ... '** — Select this if do not want to change the issue's **Remaining Estimate** value.
   - **'Set estimated time remaining to ...'** — Select this if you want to manually set the issue's **Remaining Estimate** value to the amount specified. If you select this option, enter your new estimate into the blank field below. Use 'w', 'd', 'h' and 'm' to specify weeks, days, hours or minutes. For example, to specify 'thirty minutes', type '30m'.
   - **'Increase estimated time remaining by ...'** — Select this if you want to ...
7. Click the '**Delete**' button to confirm the deletion and return to the issue. Verify that:
   - the issue's **Work Log** tab no longer displays the work log entry that you just deleted.
   - the issue's **History** tab displays the **Worklog Id** (but not the description) of the deleted work log entry.
   - the issue's **Time Spent** field has been decreased by the value of the deleted work log entry's **Time Spent**.
   - the issue's **Remaining Estimate** field has been adjusted according to your choice in Step 6 (above).

You can delete your own work log entries if you have been granted the **Delete Own Work Logs** permission. You can delete other people's work log entries if you have been granted the **Delete All Work Logs** permission.

Customised JIRA installations

Logging work *and/or* specifying time estimates on the same JIRA screen

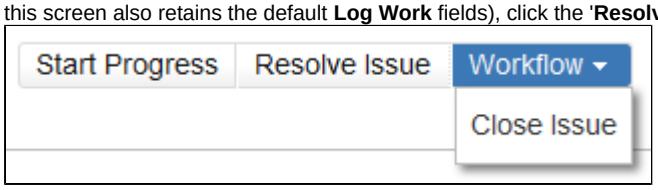
As described above, you can log work when viewing, resolving or closing an issue or specify time estimates when creating or editing an issue .

However, JIRA can be customised to allow work logging *and* specifying time estimates on the same JIRA screen when performing any JIRA operation, such as editing or creating an issue, or transitioning an issue to another status. To do this, your JIRA administrator must add both the **Log Work** *and* **Time Tracking** fields to the appropriate screen used by that operation.

To log work *and/or* specify time estimates on the same JIRA screen:

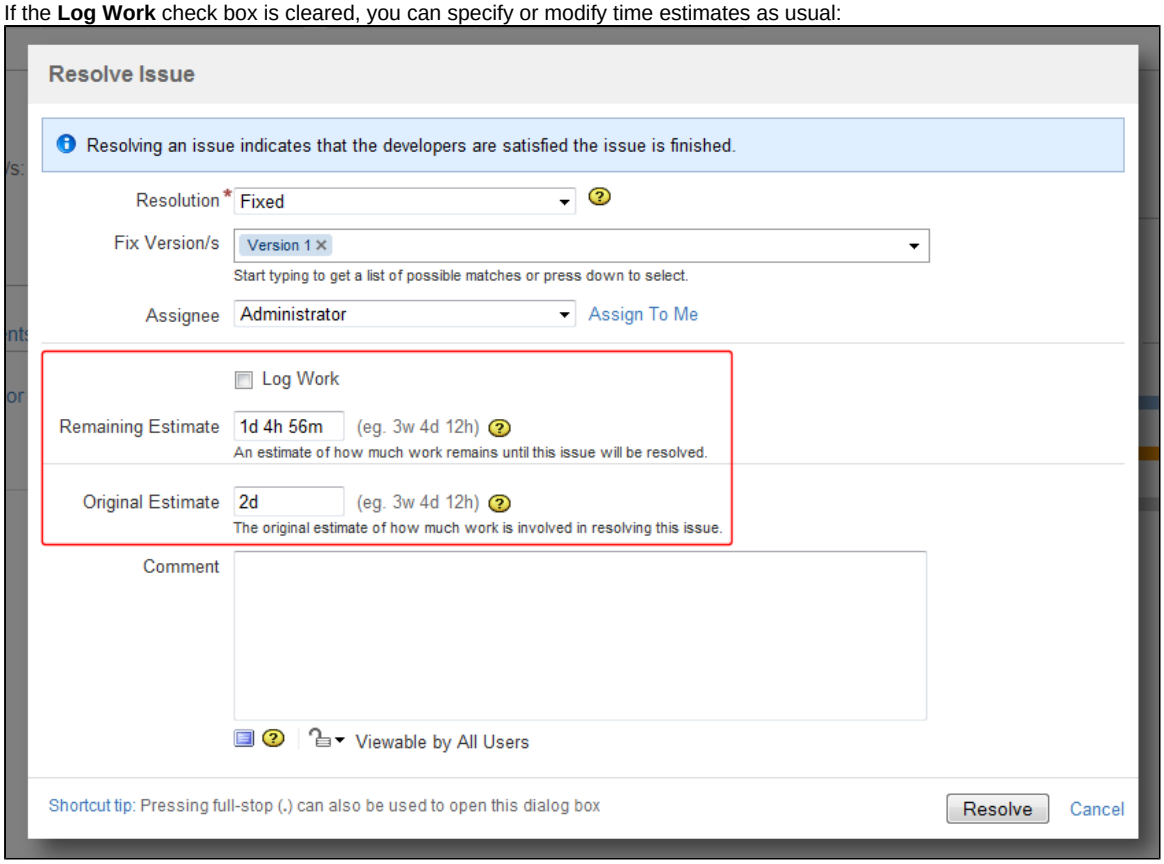
1. Navigate to the issue and view its details.
2. Perform the customised JIRA operation that allows you to log work *and* specify time estimates on the same JIRA screen. For

example, assuming that your JIRA administrator has added the **Time Tracking** fields to the **Resolve Issue Screen** (and assuming this screen also retains the default **Log Work** fields), click the '**Resolve Issue**' button at the top of the issue:

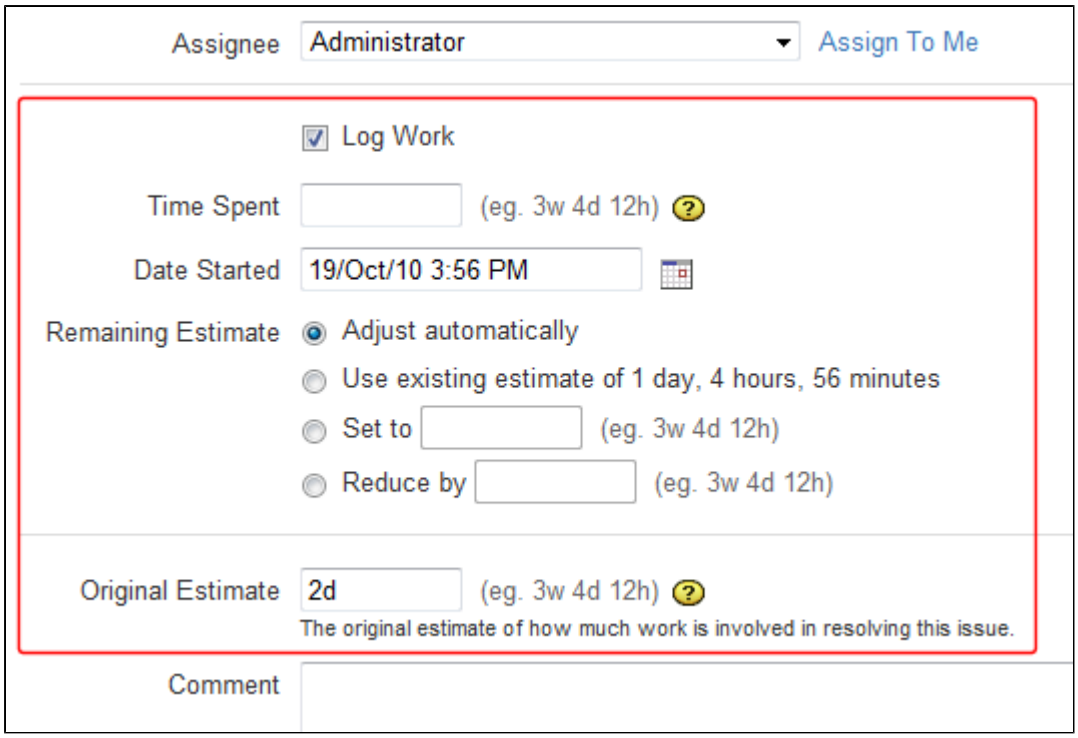


The relevant screen (e.g. Resolve Issue) opens with both the Log Work and Time Tracking fields, which are highlighted in the following three screenshots.

If logging work is optional (because your JIRA administrator has configured the Log Work fields as optional), then you can choose whether or not to log work during your JIRA operation, using the **Log Work** check box:
If the **Log Work** check box is cleared, you can specify or modify time estimates as usual:

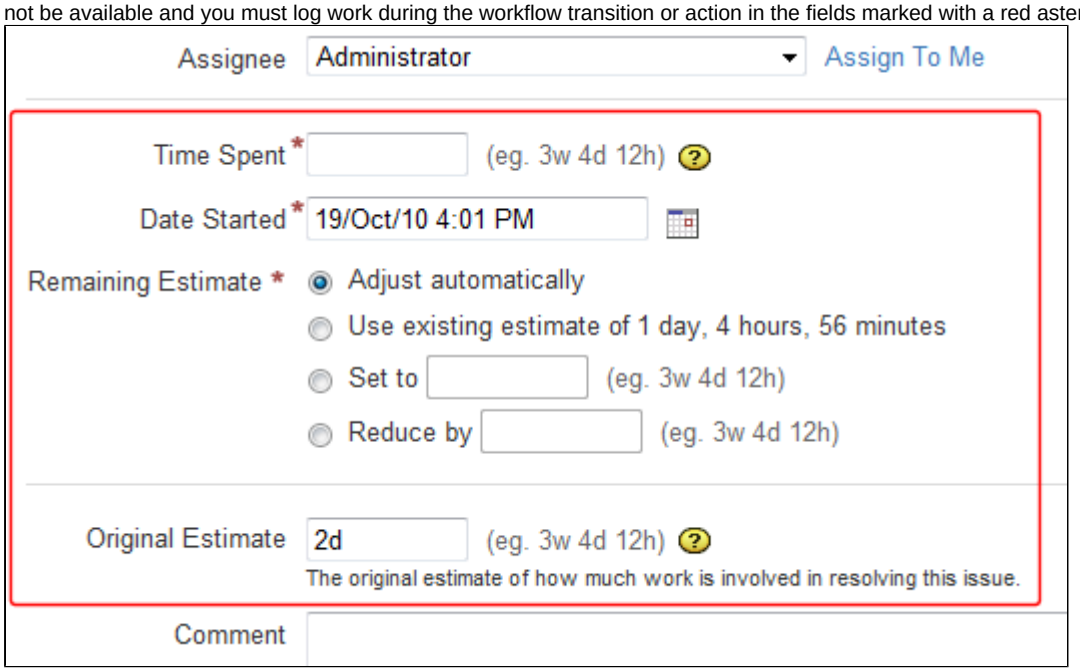


For more information about how to modify these fields, please refer to the Specifying Time Estimates section above.
If you select the **Log Work** check box, the Log Work fields become available and the **Remaining Estimate** field changes to the Remaining Estimate options for logging work.



For more information about how to modify these fields, please refer to the Logging work when viewing an issue section above.

If the Log Work fields are mandatory (because your JIRA administrator has configured them so), then the **Log Work** check box will not be available and you must log work during the workflow transition or action in the fields marked with a red asterisk:



For more information about how to modify these fields, please refer to the Logging work when viewing an issue section above.

If you are not creating an issue or Sub-Task or not explicitly using the Log Work action (above), only the **Comment** field (not the **Work Description** field) will be available for entering a description of the work activity logged.

By default, JIRA should automatically copy the contents of your **Comment** to the **Workflow Description**. In doing so, the work log will be visible to members of the project role or group selected in the padlock icon dropdown.

If this is not happening and you would like comments to be copied to workflow descriptions, request that your JIRA administrator enables JIRA's Copy Comments to Workflow Descriptions setting.

If this setting is disabled:

- The work log entry may be visible to anyone. If this is a concern, you need to edit this work log entry after creating it to modify its visibility.
- Copying comments to workflow descriptions must be done manually after logging work.

3. If you also wish to modify the time estimates, edit the relevant fields as described under Specifying time estimates (above).

See also

- Workload Pie Chart Report
- User Workload Report
- Version Workload Report
- Time Tracking Report

Modifying Multiple ('Bulk') Issues

On this page:

- About Bulk Operations
  - About the 'Bulk Change' Global Permission
  - Disabling Mail Notification for Bulk Operations
- Performing a Bulk Operation
  - Bulk Move
    - Select Issues
    - Select Projects and Issue Types
    - Select Projects and Issue Types for Sub-Tasks
    - Workflow Status Mapping
    - Field Updates
    - Retain Original Values
    - Bulk Move Confirmation
  - Bulk Edit
    - Available Operations
    - Unavailable Operations

About Bulk Operations

'*Bulk Operations*' enable multiple operations to be performed on multiple issues at once. The bulk operations are performed on the result set of a search. The following list details the available bulk operations:

- **Workflow Transition**
  This operation allows multiple issues to be transitioned through workflow at once — e.g. resolve a collection of issues.
- **Delete**
  This operation allows multiple issues to be deleted at once.
- **Move**
  This operation allows multiple issues to be moved between projects and/or issue types at once. Please see the Bulk Move section for further details.
- **Edit**
  This operation allows multiple fields in multiple issues to be edited at once. Please see the Bulk Edit section for further details.

About the 'Bulk Change' Global Permission

In order to execute a bulk operation, you will need to be granted the appropriate project-specific permission and the global Bulk Change permission by your JIRA administrator. For example, you would need to be granted both the '**Move Issue**' and '**Bulk Change**' permissions in order to execute the **Bulk Move** operation.

The project-specific permissions are still respected for the collection of issues selected for the bulk operation.

Disabling Mail Notification for Bulk Operations

It is possible to disable mail notification for a particular bulk operation by de-selecting the 'Send Notification' checkbox in the bulk operation wizard. In order for this option to be available, you must be an administrator or project administrator of all the associated projects on whose issues the bulk operation is being performed. Bulk Change fires the Issue Updated event, you can read more about this here.

Performing a Bulk Operation

1. From the Issue Navigator, perform a search with the required filters to produce an issue result set.
2. Select the **Bulk Change** option from the '**Tools**' menu of the Issue Navigator. (Note: the **Bulk Change** link is only available to people who have been granted the global Bulk Change permission.) If the result set spans a number of pages, it is possible to select all issues within the result set to be considered for the bulk operation. Alternatively, all issues on the current page can be selected for the bulk operation.
3. The next screen allows the selection of the issues that the bulk operation is to be performed on.
4. The next screen allows the bulk operation to be selected - **Workflow Transition, Delete, Move** or **Edit.**
5. If the **Delete** operation is selected, the final step is confirmation of the delete operation on the issues selected.
6. If the **Edit** operation is selected, the next screen provides a list of the available edit operations that can be performed on the issues selected. Some operations may be unavailable; please check the Bulk Editsection (see below) for further details.
   - After selecting the required **Edit** operation(s), the final step is confirmation of the edit operation(s) on the selected issues.

7. If the **Move** operation is selected, the next screens allow a target project and issue type to be selected, with the ability to migrate workflow statuses and update required fields as necessary. Further details can be found in the Bulk Move section.
8. If the **Workflow Transition** operation is selected, the next screen shows the available workflow transitions that can be performed on the issues. The transitions are grouped by workflow — along with a list of the affected issues for each workflow transition. Once an operation is selected, the appropriate field screen for that operation is displayed — allowing any necessary field edits that are required to complete the transition. It should be noted that only those issues associated with the selected transition will be updated. It is only possible to select one transition per bulk workflow transition operation.

Bulk Move

The **Bulk Move** operation allows multiple issues to be moved at once. It is possible to move a selection of issues to a new project, with the ability to select a new issue type in certain cases. The issues are selected through the Issue Navigator as discussed above.

The operation is completed as follows:

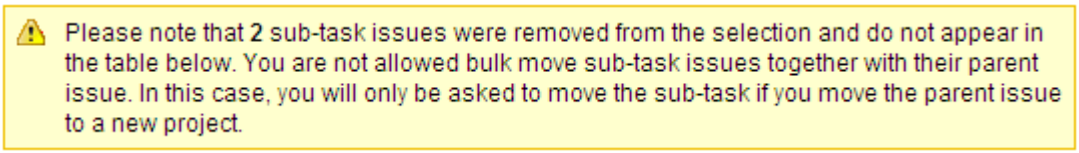
1. Select Projects and/or Issue Types
2. Select Projects and/or Issue Types for Sub-Tasks
3. Select status migration mappings for invalid statuses
4. Select values for required fields and fields with invalid values
5. Confirm changes to be made and complete the operation
   Note that steps 3 and 4 will occur once for each different target project and issue type combination.

Select Issues

The bulk move operation can be performed on both standard issues and sub-task issues. Standard issues can be moved to another project and issue type, whereas a sub-task can only have its issue type changed. (Note that it is possible to convert a sub-task to an issue, and vice versa.)

It is **not** possible to select *both* a sub-task and its parent to bulk move. This is so as to adhere to the parent/sub-task relationship (i.e. the sub-task is always located in the same project as the parent issue). Any sub-tasks of selected parent issues which were also selected will be automatically discarded from the move.

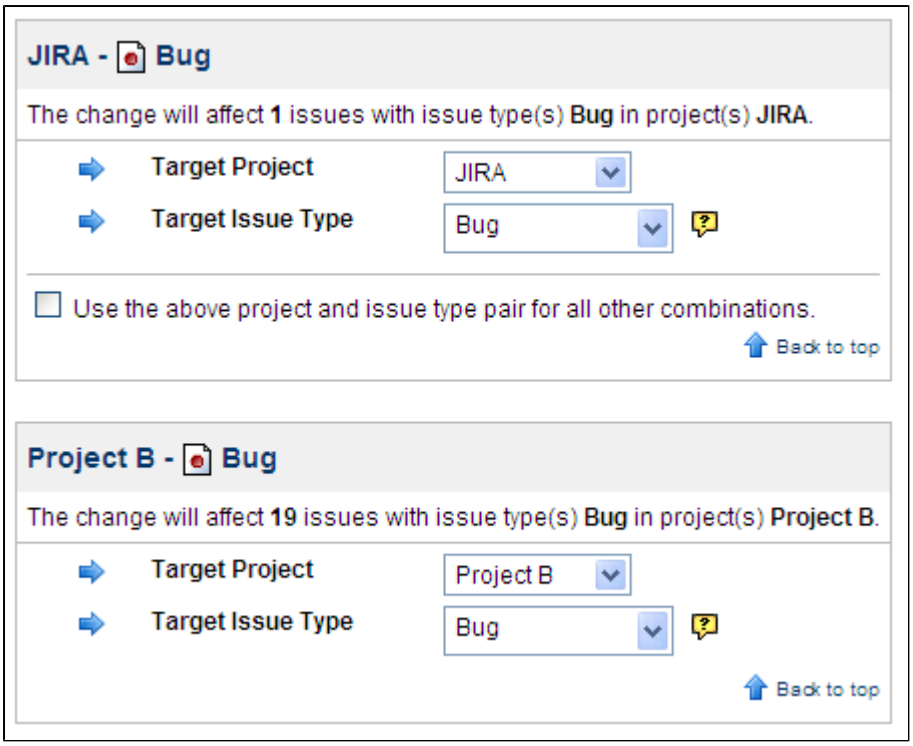
For example, you have issue B being a sub-task of issue A and you try to bulk move both A and B simultaneously. You will see a warning message (see below) and will be prompted to select a target project and issue type for issue A. If you select a new project for A, you will be prompted to move the sub-task to a new issue type based on issue A's new project. If you *don't* change the project for issue A, the sub-task will not be required to be moved.



Select Projects and Issue Types

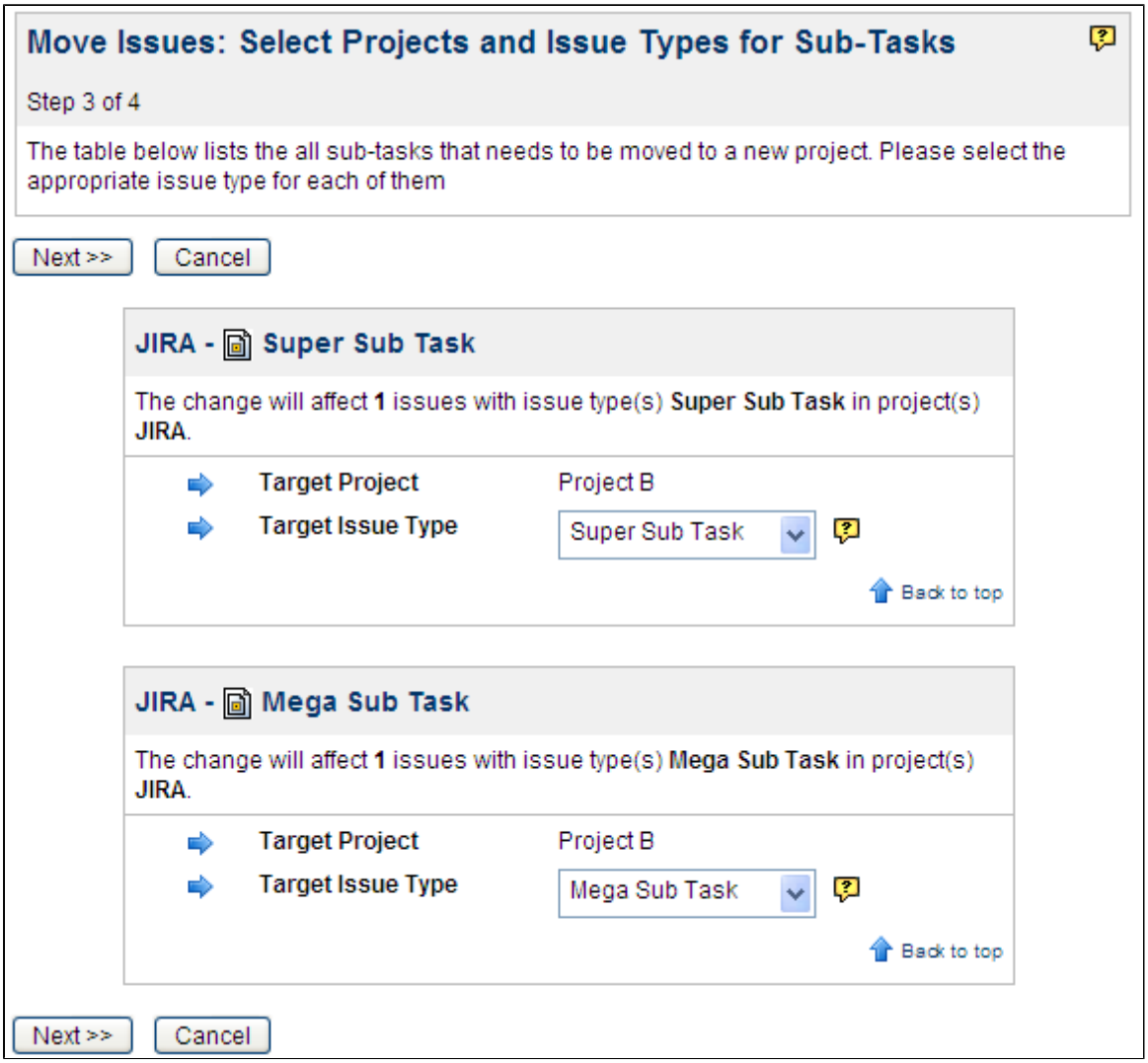
The first step of the Bulk Move wizard is to choose which projects and issue types you will move your issues to. The target project and issue type will determine whether extra steps will be required to migrate statuses and fields.

This screen shows all selected issues grouped by their current project and issue type. You can either select a new project and issue type for each one or choose to move all standard issues to a single project and issue type. To do this, select the check box with the label *Use the above project and issue type pair for all other combinations* and the selected project / issue type will apply. Note that this will not apply to sub-tasks since they cannot be moved to a standard issue type.



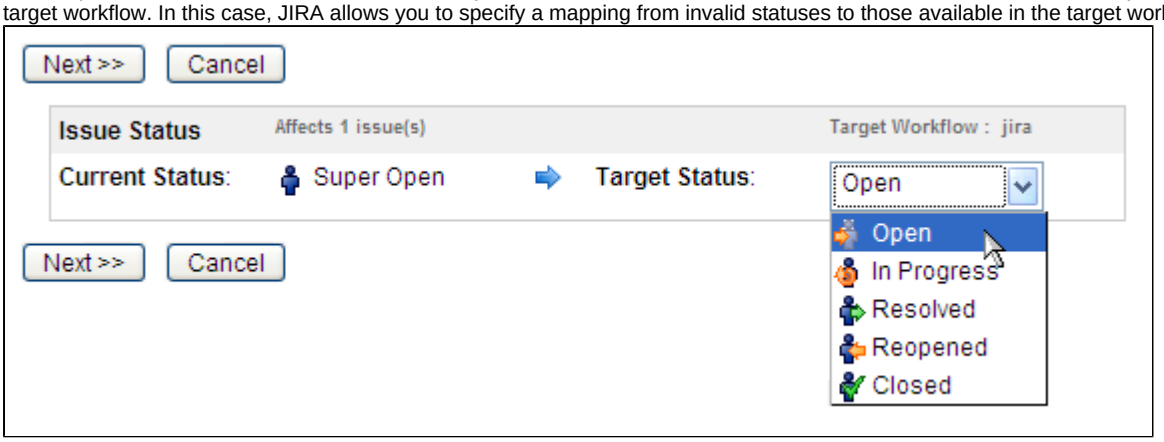
Select Projects and Issue Types for Sub-Tasks

If you are moving issues with sub-tasks to another project, you will also need to move the sub-tasks to the new project. On this screen you can elect to change the issue types of the sub-tasks being moved if you need to.

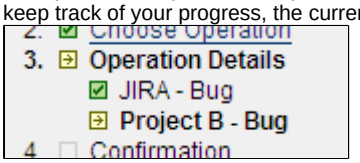


Workflow Status Mapping

As multiple workflows can be active simultaneously, some statuses associated with the collection of selected issues may not be valid in the target workflow. In this case, JIRA allows you to specify a mapping from invalid statuses to those available in the target workflow.



This step of the wizard will only appear if you have invalid statuses. If you are moving issues to different projects and issue types at the same time, you will complete this step as well as the next for each of the different target project and issue type combinations. To help you easily keep track of your progress, the current context, that is the target project and issue type, is highlighted in the left-hand 'progress' pane.



Field Updates

In order to adhere to the field configuration scheme associated with the target project and issue type, it may be necessary to update/populate required fields (e.g. fields that are required in the target project, but may not have been in the original project).

For each field that needs to be populated, you will be prompted to supply a value. This value will be applied to all issues that are being 'Bulk Moved' together (see '**Retain Original Values**' below for more details).

For the following fields, JIRA will provide a list of possible values for you to select from:

- Component
- Affects Version
- Fix Version
- Custom fields of type 'Version-Picker'

Note that versions which have been archived in the target project cannot be selected as the target when performing a bulk move. If you need to move issues into an archived version, you will need to first unarchive the version in the target project.

Retain Original Values

It is possible to retain original field values that are valid in the target destination by checking the **Retain** checkbox associated with the field. For example, some issues may already include a valid custom field value — these values can be retained, while issues that require an update will adopt the value specified on the '**Field Update**' screen.

- **Checked:** the original value is retained where possible[1]. The field will not be updated with the specified new value.
- **UnChecked:** all fields will be updated with the specified new value.

Note that the '**Retain**' checkbox is not available for the following fields, since an explicit mapping is required:

- Component
- Affects Version
- Fix Version
- Custom fields of type 'Version-Picker'

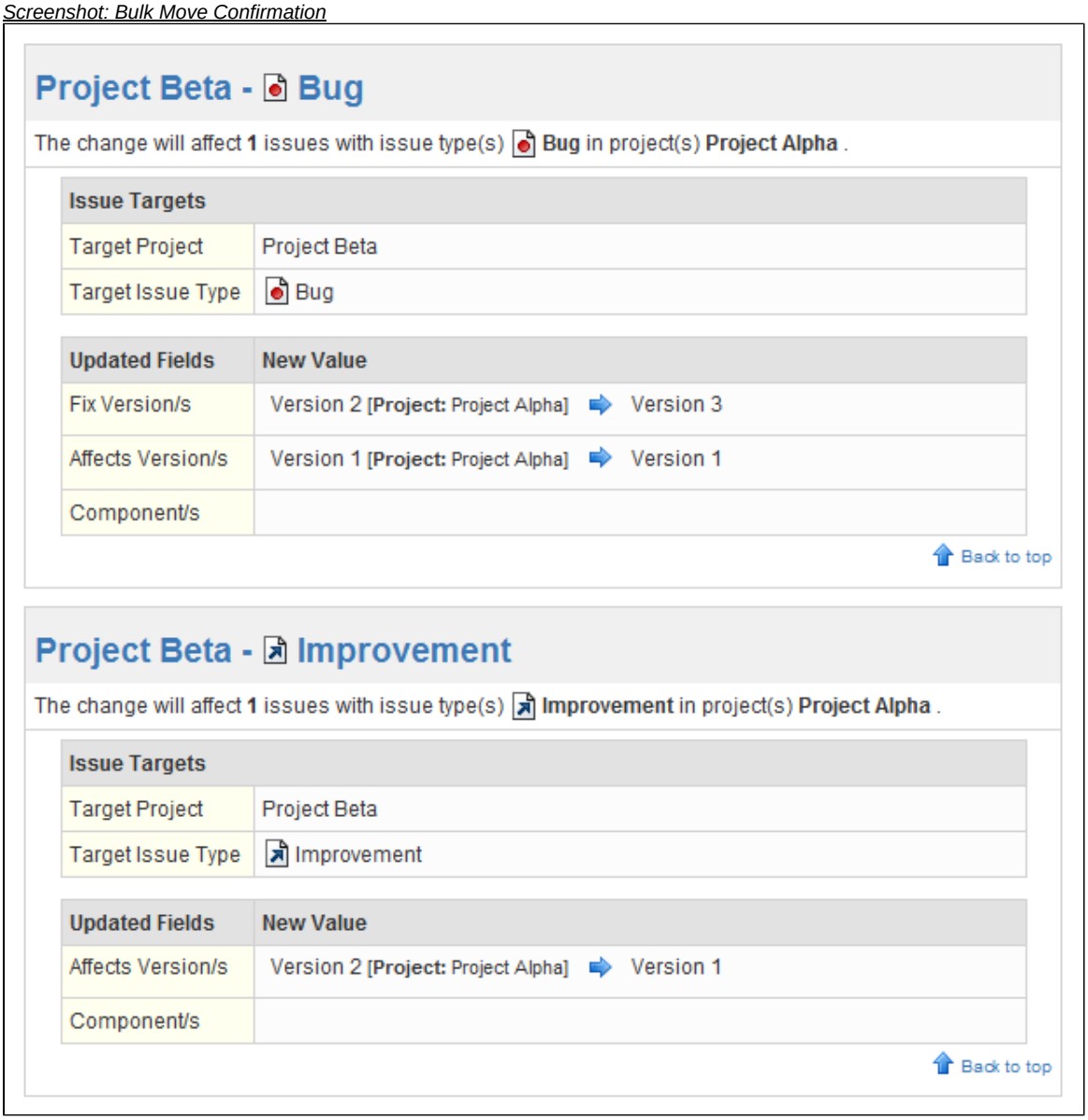
Bulk Move Confirmation

When all move parameters — e.g. target project, status mappings and field updates — have been specified for all issues, you will be presented with a confirmation screen displaying all changes that will be made to the issues being moved. The following details are displayed as applicable:

- **Issue Targets:** the target project and issue type
- **Workflow:** the target workflow and invalid status mappings
- **Updated Fields:** new values for fields that require updating
- **Removed Fields:** values to be removed in fields that are not valid in the target

The issues will only be moved once the **Confirm** button is clicked from the confirmation page. If the operation is exited anytime before this step, no changes will be made to the issues.

Screenshot: Bulk Move Confirmation



Bulk Edit

The **Bulk Edit** operations available depend on the issues selected and the nature of the field it changes.

Available Operations

The following table lists out the possible operations. Please note that *all* the conditions must be true for the corresponding operation to be available.

Available Operations	Conditions
Change Affects Version/s	• Selected issues belong to one project, and that project has version/s • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Assign To	• This field is not hidden in any field configurations the selected issues belong to • Current user has 'assign issue' permission for all the selected issues
Change Comment	• This field is not hidden in any field configurations the selected issues belong to • Current user has 'comment issue' permission for all the selected issues

Change Component/s	• Selected issues belong to one project, and that project has component/s • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Due Date	• This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues • Current user has 'schedule issue' permission for all the selected issues
Change Fix For Version/s	• Selected issues belong to one project, and that project has version/s • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Issue Type	• Current user has 'edit issue' permission for all the selected issues
Change Priority	• This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Reporter	• This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues • Current user has 'modify reporter' permission for all the selected issues
Change Security Level	• This field is not hidden in any field configurations the selected issues belong to • All the selected projects are assigned the same issue level security scheme • Current user has 'edit issue' permission for all the selected issues • Current user has 'set issue security' permission for all the selected issues
Change Custom Fields	The 'Change Custom Fields' operation is available only if: • a global custom field exists **OR** • an issue type custom field exists and the issues are all of this specific issue type **OR** • a project custom field exists and the issues are all of the same project
Edit a Closed Issue	• Your workflow must allow editing of closed issues

Unavailable Operations

The fields listed in this section have no operations for bulk editing. This is because there is an alternative method or it is not logical to perform bulk edit on them.

The following system fields are unavailable for bulk editing:

- Attachments
- Summary
- Description
- Environment
- Project — Please use 'Bulk Move' to move issues between projects
- Resolution — Please use 'Bulk Workflow Transitions' to modify the resolution of issues
- Time Tracking fields — Original Estimate, Remaining Estimate, Time Spent

The following custom fields are unavailable for bulk editing:

- Import Id
- Read Only Text

Moving an Issue

JIRA allows you to easily move an issue from one project to another by using the 'Move Issue' wizard.

Please note that you must have the appropriate project permissions to move an issue from one project to another, i.e.

- You must have the 'Move Issues' permission for the project which has the issue that you want to move.
- You must have the 'Create Issues' permission for the project that you wish to move your issue to.

If you do not have either of this permissions, please contact your JIRA administrator to have these added to your user profile.

> If you wish to move multiple issues between projects at the same time, please refer to the documentation on bulk moving issues.

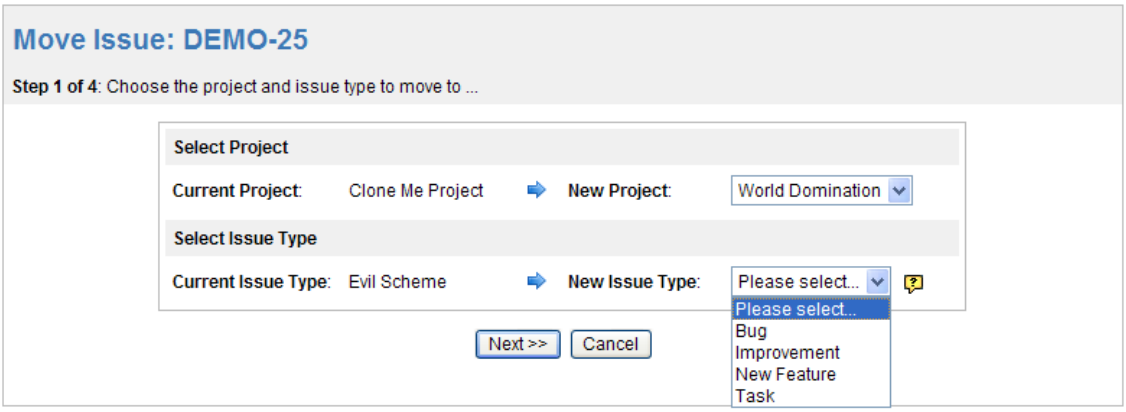
Moving an Issue

The 'Move Issue' wizard allows you to specify another project in your JIRA instance to move your selected issue to. As there may be significant differences in the configuration of your original project and target project, the 'Move Issue' wizard allows you to change certain attributes of the issue. These include:

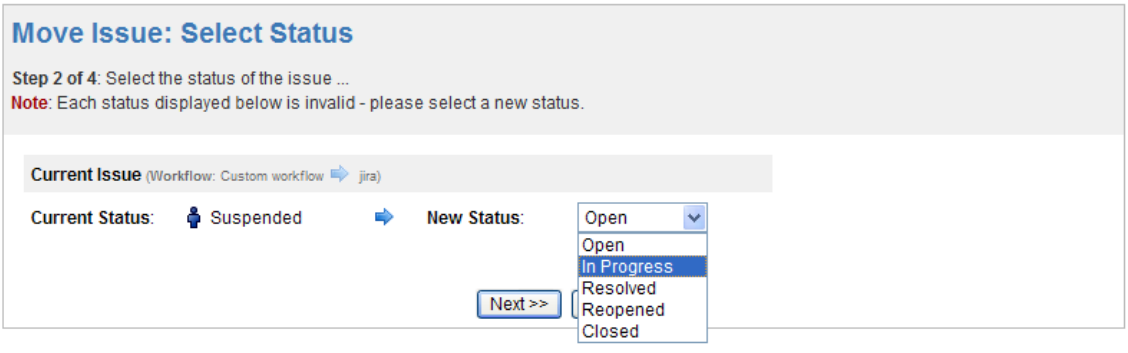
- **Issue Type** — If your issue is a custom issue type that does not exist in your target project, you must select a new issue type. You can also choose to arbitrarily change the issue type, if you wish.
- **Issue Status** — You may have set up custom issue statuses as part of a workflow. If you have assigned a custom status to your issue and it does not exist in your target project, you must select a new issue status for your issue. You cannot arbitrarily change the issue status, i.e. the option to change the issue status will only appear if you are required to change it.
- **Custom Fields** — If you have defined **required** custom fields for your issue, which do not exist in your target project, you must set values for them. You will only be prompted to change the enter values for **required custom fields** in the target project, that are missing values. If you wish to change the existing values for other fields on your issue, you can do this after the move is complete.

To move an issue:

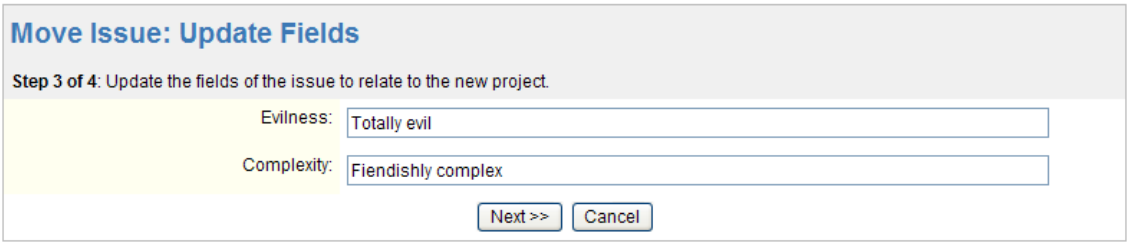
1. View the issue that you wish to move.
2. Select '**Move**' from the '**More Actions**' drop-down menu.
3. The first page of the 'Move Issue' wizard will display. Select the project that you wish to move your issue to, and if required/desired, change the issue type. Click '**Next**' to continue.



4. If you are required to change the status of your issue (see explanation above), the 'Select Status' page will display. Select the new status for your issue and click '**Next**' to continue.

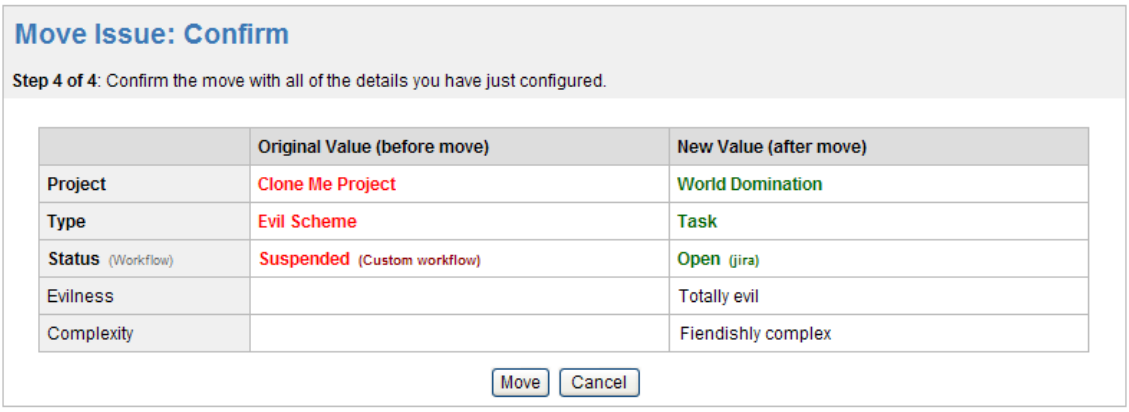


5. If you are required to specify the values for any required custom fields (see explanation above), the 'Update Fields' page will display. Specify the desired values for each field, and click '**Next**' to continue.



6. The 'Confirmation' page will display with all of your changes. If you wish to revise any of your changes, you can click the appropriate

step in the left-hand menu to return to that page of the wizard. Once you are happy with your changes, click '**Move**' to move the issue to the target project.



7. Your issue will be moved to the target project and displayed on screen. You can now edit the issue to make further changes, if you wish.

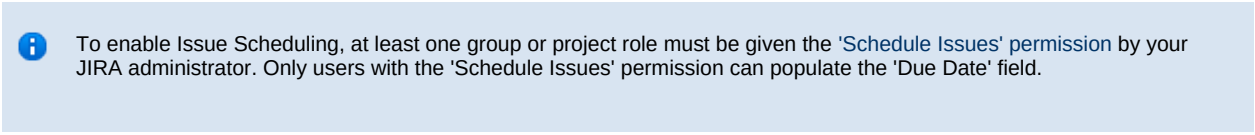
Scheduling an Issue

On this page:

- Scheduling An Issue
- Searching by 'Due Date'
  - Using Simple Search
    - Fixed Date Searches
    - Relative Period Search
      - Due Date Popup
      - Relative Period Search Syntax
  - Using Advanced Search

Scheduling An Issue

To schedule an issue, populate its 'Due Date' field. This can be done either when creating an issue, or at a later stage by editing the issue.



To enable Issue Scheduling, at least one group or project role must be given the 'Schedule Issues' permission by your JIRA administrator. Only users with the 'Schedule Issues' permission can populate the 'Due Date' field.

Searching by 'Due Date'

You can used either simple search or advanced search to search for issues by their Due Date.

Using Simple Search

You can search for issues using the search form Issue Navigator (see Searching for issues. There are two ways to search for issues based on the 'Due Date' field. The first way is using fixed date values, the second is using periods that are relative to the current date.

Fixed Date Searches

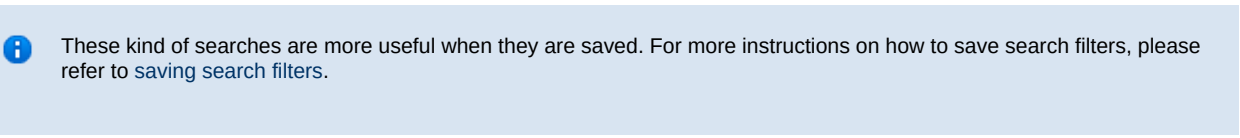
There are two text fields in the search form that allow searching based on the 'Due Date' field.

- To search for all issues that are due after a certain date, enter the date into the 'Due After' text field of the Issue Navigator. For example to find all issues that are due after 1st June 2010, enter 1-6-2010 into the 'Due After' field. You can also use the Calendar popup to select a date by clicking the calendar icon to the right of the 'Due After' field.
- To search for issues that are due before a certain date, enter the date into the 'Due Before' date. For example, to find all issues that are due before 1st July 2010, enter 1-7-2010 into the 'Due Before' field.

It is also possible to search for issues that are due between two dates by populating both the 'Due After' and the 'Due Before' fields.

Relative Period Search

It is possible to perform a search that is relative to the time when it is run. For example, it is possible to do a search for issues that are due seven days from now. To do this, enter 7d into the 'Due Date To' text field of the Issue Navigator. If the search is saved and run the next day, the issues that are due in seven days from the time that the search is run will be retrieved. Thus, this search will find all issues that are due within a week every time it is run.

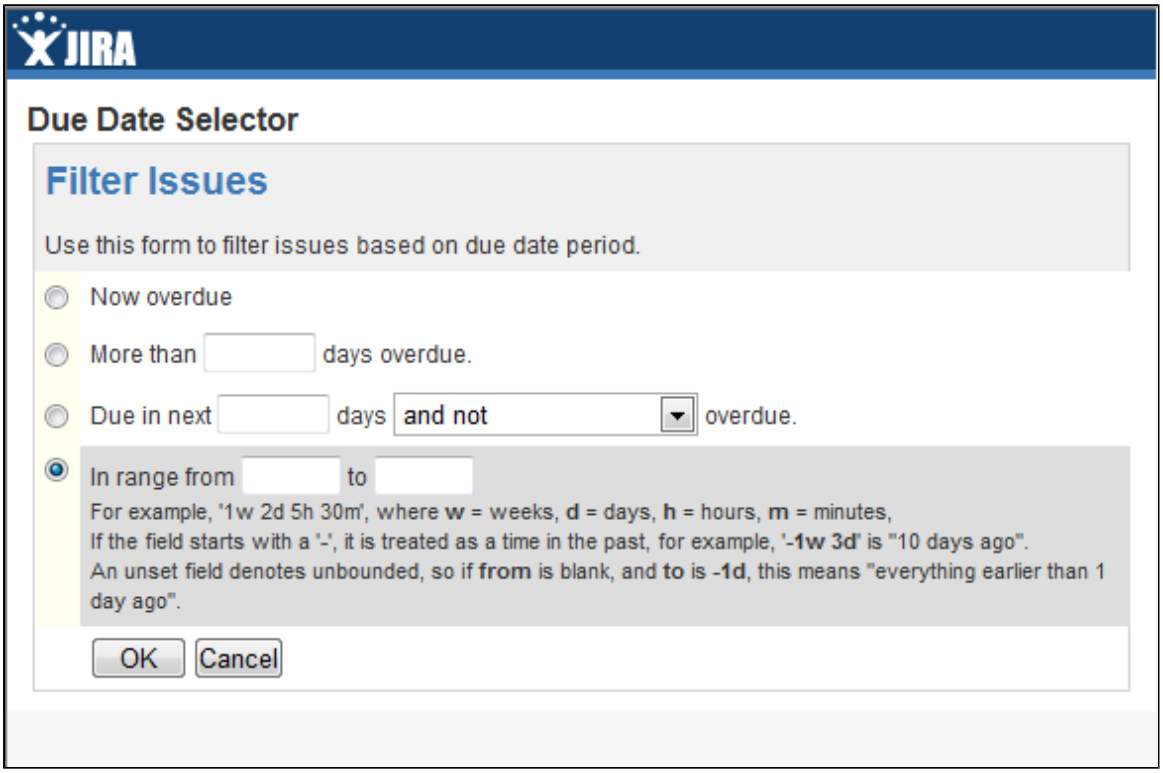


These kind of searches are more useful when they are saved. For more instructions on how to save search filters, please refer to saving search filters.

The values that are entered into the 'Due Date From' and 'Due Date To' fields have to conform to a special syntax (described below). However, it is also possible to use the Due Date popup by clicking the icon to the right of the 'Due Date To' text field to specify the search period.

Due Date Popup

The Due Date popup is shown below.



- To search for issues that are overdue at the time of the search select the first radio button and click '**OK**'.
- To search for issues that are overdue by more than a certain number of days, populate the text field in the second row, (select the second radio button, if it is not selected for you automatically) and click '**OK**'.
- To search for issues that are due in the next certain amount of days and are not overdue at the time of the search, populate the text field in the third row with the number of days, and choose '**and not**' from the select box in the third row. Select the third radio button, if one it was not selected automatically, and click '**OK**'.
- To search for issues that are due in the next certain amount of days and are overdue at the time of the search, populate the text field in the third row with the number of days, and choose '**and**' from the select box in the third row. Select the third radio button, if one it was not selected automatically, and click '**OK**'.
- The fourth row of the popup is used for arbitrary period searches. Use the '**to**' text field to specify the upper bound of the search, and the '**from**' text field to specify the lower bound of the search. A blank text field means no bound. Populating the text fields in the fourth row, actually has the same effect as populating the 'Due Date From' and 'Due Date To' text boxes in the Issue Navigator. The fields in the popup expect entries in the same syntax as the ones in the Issue Navigator. The syntax is described below

Relative Period Search Syntax

The 'Due Date From' and 'Due Date To' fields use a special syntax to denote time period bounds. The syntax uses numbers and abbreviations that follow the numbers to represent what the numbers actually mean. The abbreviations are "w" for weeks, "d" for days, "h" for hours and "m" for minutes. For example, to specify 10 days in the future use "10d" or "1w and 3d". To specify a period bound in the past prefix the value with the "-" sign. For example to specify 2 days, 4 hours and 3 minutes ago, use "-2d 4h 3m".

Using Advanced Search

You can also use JIRA Query Language (JQL) to search for issues by Due Date — see Advanced Searching, and particularly the documentation on the 'Due' field.

Setting Security on an Issue

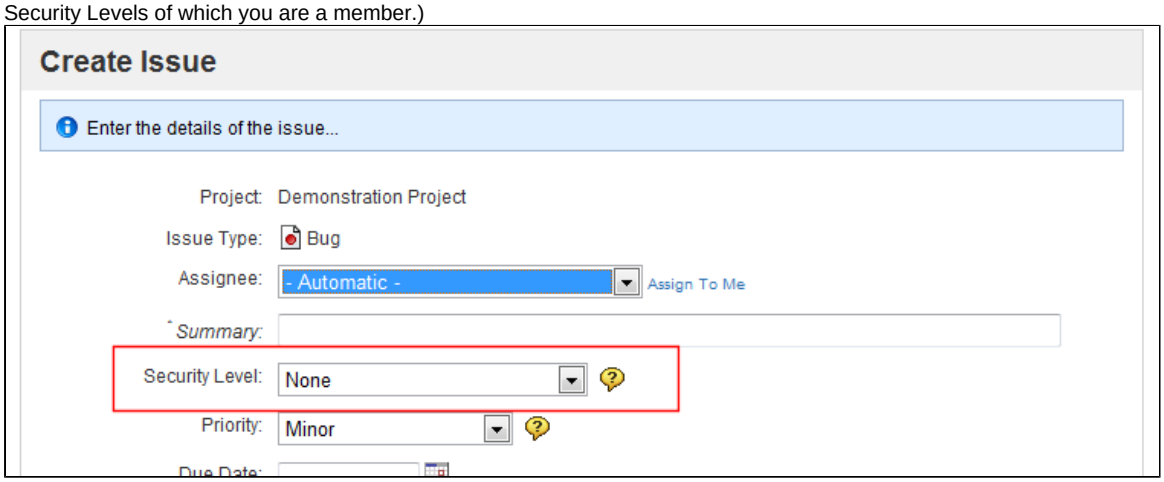
Setting the Security Level on an issue restricts the access of that issue to only people who are a member of the chosen Security Level. If you are not a member of that Security Level then you cannot access that issue and it will not appear in any filters, queries or statistics.

The Security Level of an issue can be set either when the issue is being created or afterwards when the issue is being edited.

> To be able to set the Security Level for an issue, your administator must add you to the appropriate Issue Security Level, and also grant you the 'Set Issue Security' permission for the appropriate project(s).

Setting Security on an Issue

1. Create/edit the relevant issue.
2. In the '**Security Level**' drop-down field, select the desired security level for the issue. (Note that the drop-down will only include Security Levels of which you are a member.)



3. When you save the issue, the issue will then only be accessible to members of that Security Level.

Note:

- A person can only set an issue to a Security Level of which they are a member. This prevents the issue from being set to a Security Level of which nobody is a member and effectively becoming 'lost'.
- If the person does not have the 'Set Issue Security' permission then the default Security Level will be used. This may mean that the issue created is not visible to the person that created it. (Issue Level Security should be configured by your administrator such that this does not happen.)

Viewing an Issue's Change History

An issue's *change history* is a record of changes made to an issue, including:

- changes to an issue field
- attachment of a file
- deletion of a comment
- deletion of a worklog
- creation or deletion of an issue link

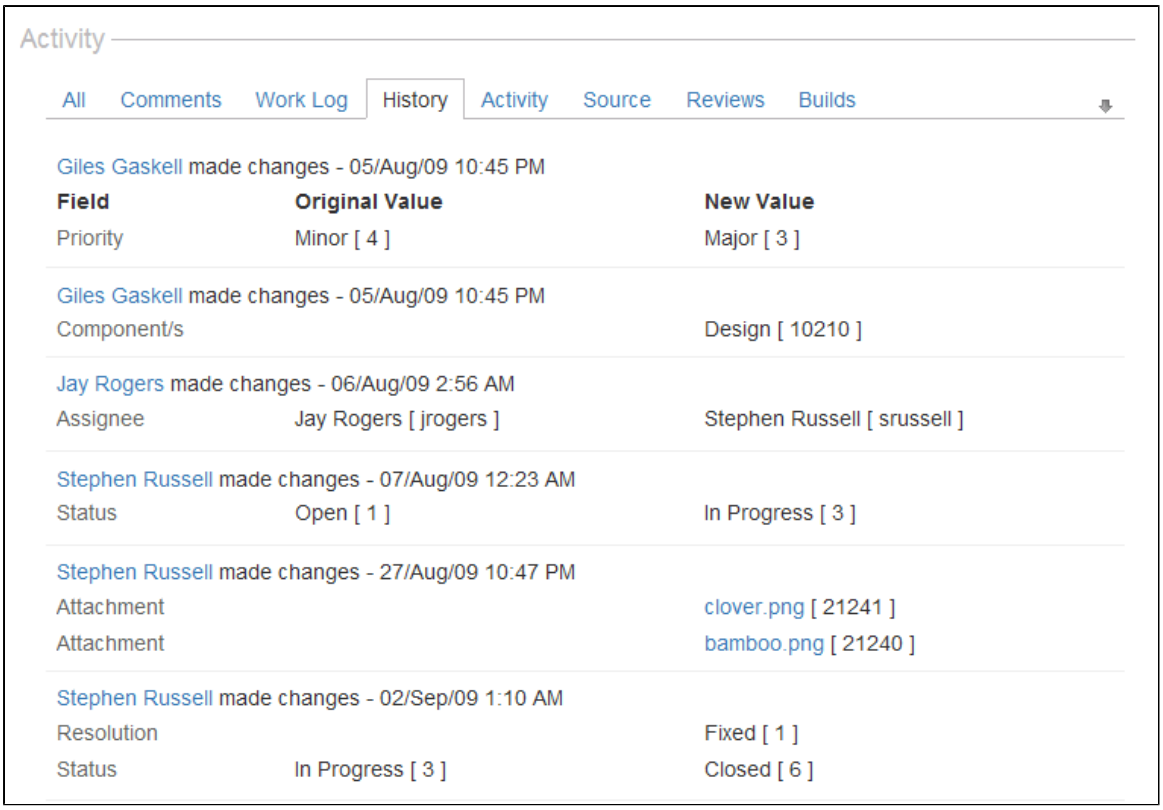
For each change, the following is recorded:

- the person who made the change
- the time at which the change was made
- if an issue field was changed, the new and old values of the field

To view an issue's change history,

1. Open the relevant issue in JIRA.
2. Click the '**History**' tab in the '**Activity**' section.
3. The list of changes to the issue will display, similar to the screenshot below.

Screenshot: An Issue's History



Viewing an Issue's Crucible Reviews

JIRA allows you to view the reviews related to an issue (that is, where the JIRA issue key was referenced in the commit message), if you are using Atlassian Crucible.

The **Reviews** tab provides you with an expandable list of code reviews related to the issue. This allows you to view the commit message and list of source-code files in each commit. You can also:

- view diffs and history for a file.
- download files.
- create a Crucible review and see the review status, if you are using Atlassian Crucible.

> To be able to view the changesets for an issue, your JIRA administrator must have configured the FishEye plugin appropriately. You will also need the 'View Version Control' permission in the appropriate projects.

Viewing an Issue's Reviews

1. Open the relevant issue in JIRA.
2. Click the '**Reviews**' tab in the '**Activity**' section.
3. The list of code reviews related to the issue will display, similar to the screenshot below.

Screenshot: The 'Reviews' Tab

See also

- Browsing a Project's Crucible Reviews

Viewing an Issue's FishEye Changesets

JIRA allows you to view the changesets related to an issue (that is, where the JIRA issue key was referenced in the commit message), if you are using a source-code repository together with Atlassian FishEye.. The regular expression used for matching the JIRA issue key in the commit message is:

```
(^|[^a-zA-Z0-9-])
```

So the key will need to be at the start of the string, or a character on each side that is not a-z, A-Z, 0-9 or the dash '-' character.

The **Source** tab provides you with an expandable list of changesets for the issue. This allows you to view the commit message and list of source-code files in each commit. You can also:

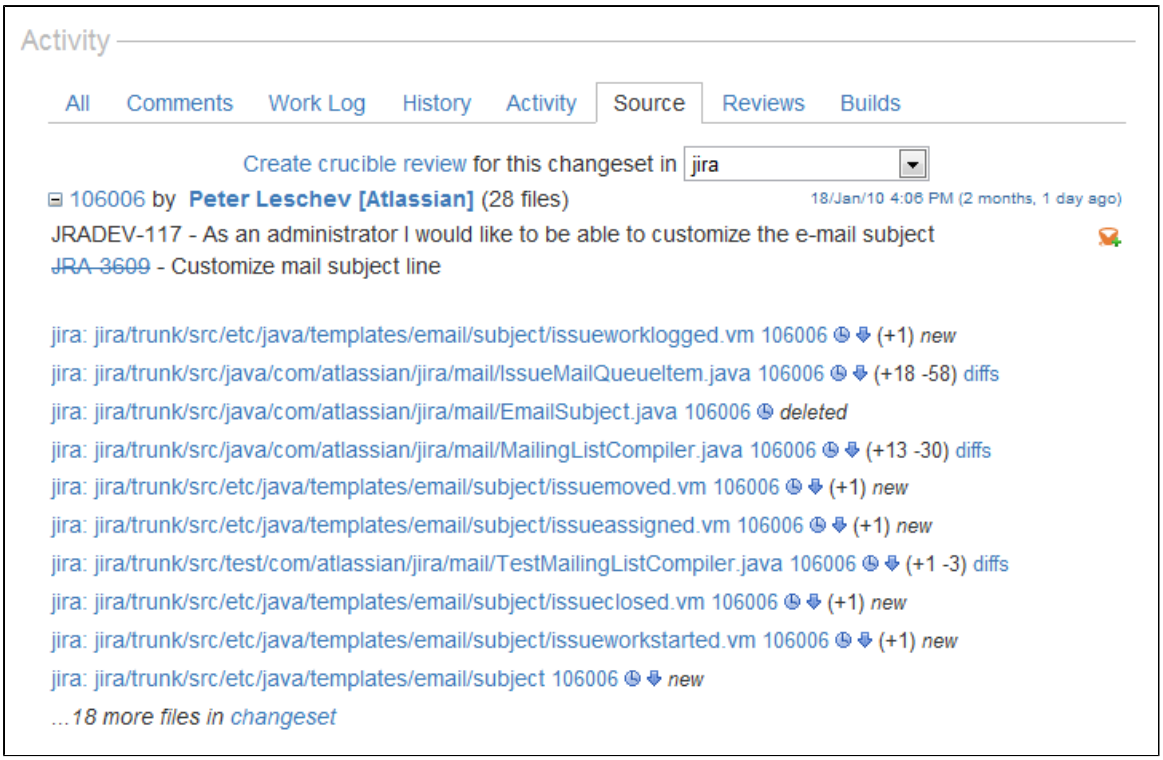
- view diffs and history for a file.
- download files.
- create a Crucible review and see the review status, if you are using Atlassian Crucible.

To be able to view the changesets for an issue, your JIRA administrator must have configured the FishEye plugin appropriately. You will also need the 'View Version Control' permission in the appropriate projects.

Viewing an Issue's Changesets

1. Open the relevant issue in JIRA.
2. Click the '**Source**' tab in the '**Activity**' section.
3. The list of changesets related to the issue will display, similar to the screenshot below.

Screenshot: The 'Source' Tab



See also

- Browsing a Project's FishEye Changesets

Viewing the Bamboo Builds related to an Issue

If your organisation uses Atlassian's Bamboo and your administrator has integrated Bamboo with JIRA, you will be able to view the Bamboo builds related to an issue.

The '**Builds**' tab provides you with a list of the builds which the issue has been linked to, either as 'Fixed' or 'Related'. (See the Bamboo documentation for instructions on linking issues to builds.)

Each entry in the list will display information about the related build, including:

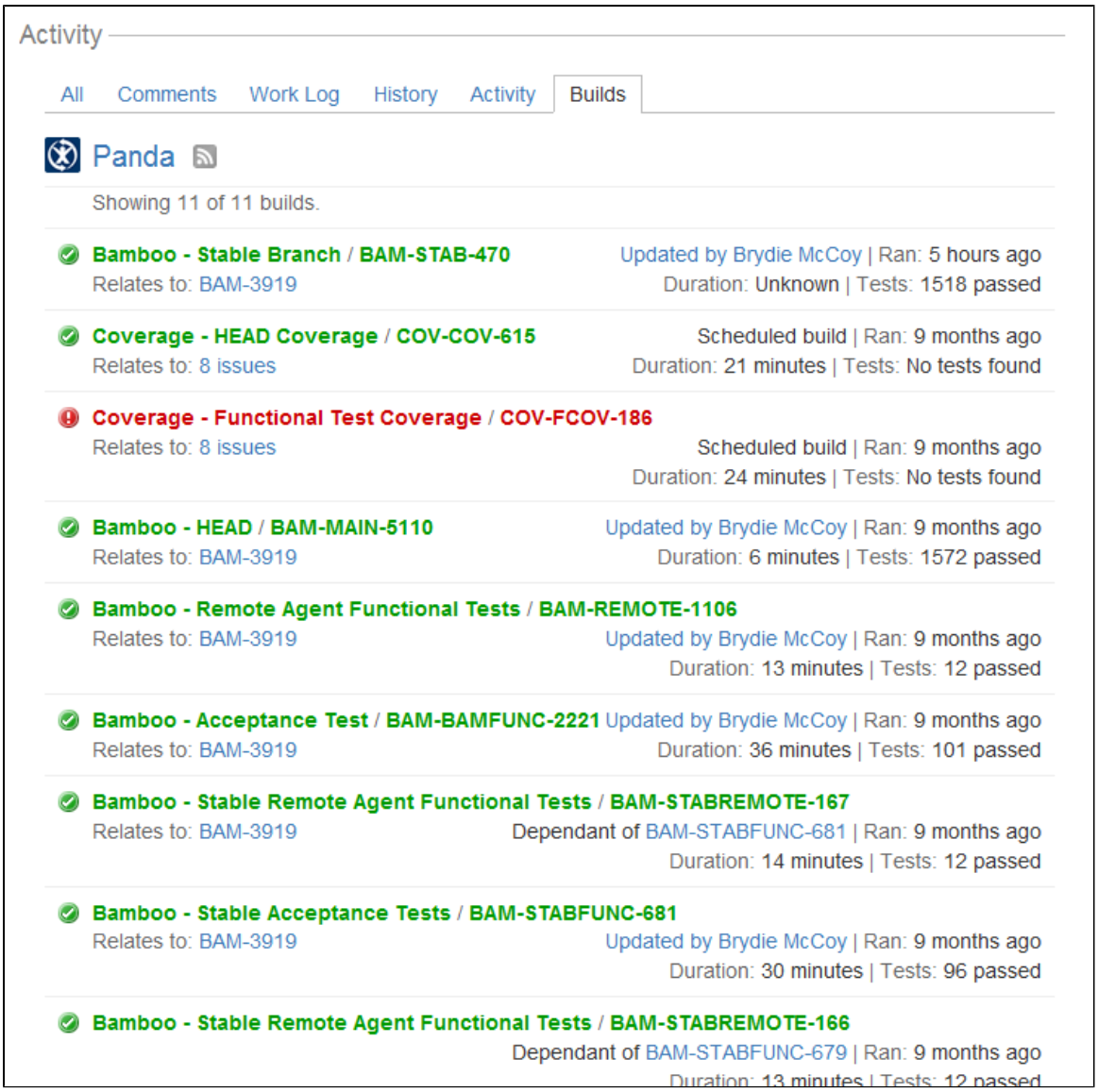
- the build name and name of the build plan
- when the build was last run
- summary information, such as related builds, duration of the build, tests passed
- build labels (if any)
- links to build artifacts (if any)

To view the Bamboo builds related to an issue,

1. Open the issue in JIRA.
2. In the '**Activity**' section, click the **'Builds'** tab.
3. The builds related to the issue will display, similar to the screenshot below.

> If you cannot see the '**Builds**' tab, your administrator may need to add the **'View Version Control'** permission to your project.

Screenshot: Viewing the Builds related to an Issue



See Also

- Browsing a Project's Bamboo Builds
- Browsing a Version's Bamboo Builds

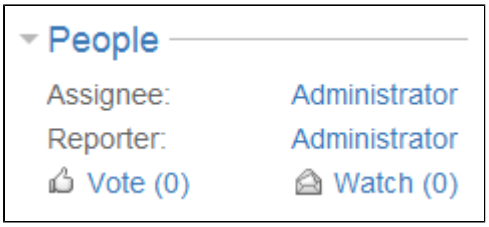
Watching and Voting on an Issue

JIRA allows you to **vote** for a particular issue — "voicing" your preference for that issue to be resolved or completed. JIRA also allows you to **watch** a particular issue, signing up for notifications of any updates relating to that issue (provided an appropriate notification scheme has been set up for the project by your JIRA administrator).

If you have the correct permissions (see below), you can also view the voter and watcher lists for an issue and, you can manage the watcher list — that is, add other people to the watcher list. This is useful if you need to draw someone's attention to a particular issue.

The voter and watcher lists are shown in at the right of the screen when viewing an issue:

Screenshot: an Issue's 'Votes and Watchers' section



You can:

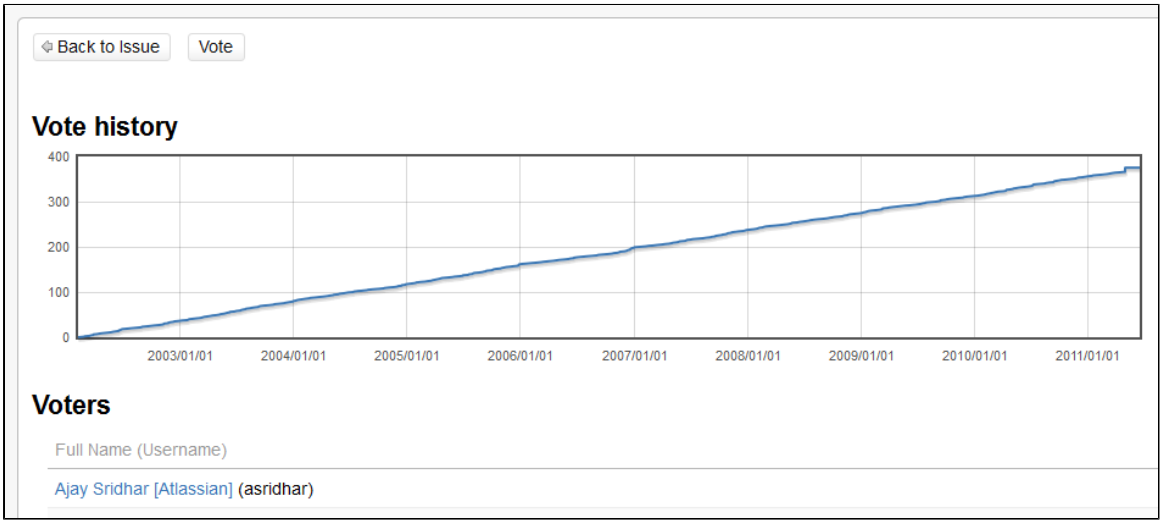
- click Vote to instantly vote for the issue.

At any subsequent time when logged in, click this again to remove your vote.
- if you have the correct permissions, click the hyperlinked number of votes to view the list of people who have voted for the issue (see screenshot below).
- click Watch to instantly become a watcher of the issue.

At any subsequent time when logged in, click this again to stop watching the issue.
- if you have the correct permissions, click the hyperlinked number of watchers to view and edit the list of people who are watching the issue. The **Watchers** form will appear (see screenshot below). You can type the required username(s) into the field provided, or click the 'user-picker' icon to select the username(s) from a list:

Screenshot: Watchers List

Screenshot: Vote History and Voters List



Permissions

JIRA incorporates two permissions to govern who may view/edit the voter and watcher lists:

- **View Voters and Watchers** — permits a user to view both the voter and watcher lists
- **Manage Watcher List** — permits a user to view/edit the watcher list

These permissions are granted by your JIRA administrator, through a Permission Scheme.

On top of this, your JIRA administrator must also ensure that JIRA's **Allow users to vote on issues** setting has not been switched off. (See Configuring JIRA Options for more information.)

Please note: It is not possible to edit the voter list.

Searching for Issues

JIRA provides a powerful issue search facility. You can search for issues across projects, versions and components using a range of search

criteria. If you need to find issues based on time-tracking details, then the standard 'simple search' interface also allows you to search agains the work logged on issues. JIRA also makes custom fields available as search criteria, allowing you to refine your searches even further.

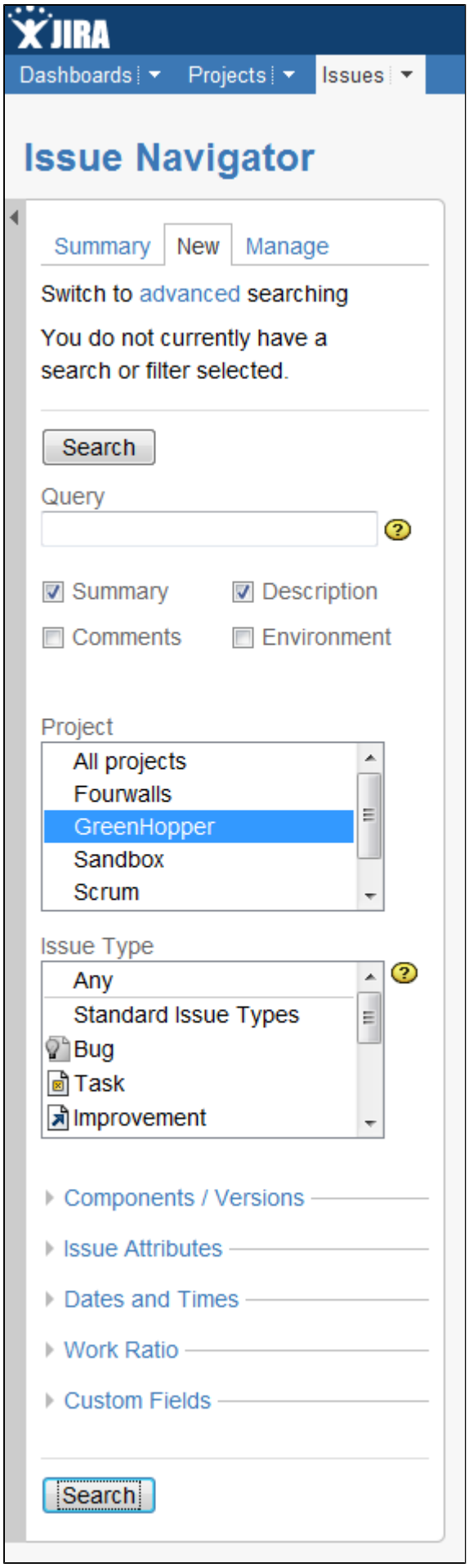
You can also save your search as an issue filter in JIRA, allowing you to recall the same search and run it again or even share it with other users. Read more about issue filters.

To perform an advanced search using JIRA query language (JQL), refer to the Advanced Searching topic.

Performing a Simple Search

1. On the top navigation bar, click the '**Issues**' tab. This will display information on the issue filter or search you currently have selected if any, on the left hand side of the page. If you have an issue filter or search currently selected, the results will be displayed in the 'Issue Navigator' on the right hand side of the page. Otherwise, no results will be shown.
2. If you currently have an issue filter or search selected and wish to run a new search, click the '**New**' link on the top left hand side of the page. The search form will display on the left-hand side of the screen.
3. Type your search term(s) into the '**Query**' box (see Specifying a Query below), and/or select other criteria from the drop-down boxes and check-boxes described below. The drop-down boxes and check-boxes allow you to narrow your search, e.g. to issues in a certain project, only issues that are marked as 'critical', only issues marked as 'enhancements', only issues reported by or assigned to a particular person.
4. Click the '**Search**' button to perform the search. Your search results will be displayed in the Issue Navigator.

Screenshot: The 'Simple Search' form



To hide the left-hand column and display your search results in full-screen width, click the icon.

To expand/collapse individual sections of the left-hand column, click the icon.
While you are interacting with JIRA, your current search will be remembered, along with whether the search you are using is new, a saved search, or a modification of a saved search.

On this page:

- Performing a Simple Search
  - Specifying a Query
  - Searching particular Projects or Issue Types
  - Using the 'Components/Versions' section
  - Using the 'Issue Attributes' section
  - Using the 'Dates and Times' section
  - Using the 'Work Ratio' section
  - Using the 'Custom Fields' section

In this chapter:

- Using Quick Search
- Performing Text Searches
- Advanced Searching
- Using the Issue Navigator
- Customising your Issue Navigator
- Saving Searches ('Issue Filters')
- Receiving Search Results as an RSS Feed
- Exporting Search Results to Microsoft Word
- Exporting Search Results to Microsoft Excel
- Displaying Search Results as a Chart
- Displaying Search Results in XML
- Receiving Search Results via Email
- Sharing a Search Result

Specifying a Query

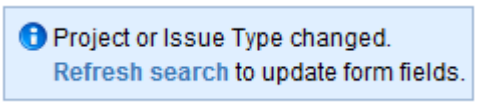
You can specify particular text to search for in any or all of the following fields:

- **Summary**
- **Description**
- **Environment**
- **Comments**

For details about how to refine your search term, please see Performing Text Searches.

Searching particular Projects or Issue Types

The **Project** and **Issue Type** fields determine which other fields are shown in the search form and what options you can see for these fields. For example, the 'Version' and 'Component' fields (see below) will only be available when searching for a single project, and will have different options for each project. When you change the project(s) or issue type(s), you may need to *refresh* the search filter in order to get the most up-to-date versions, components and custom fields. If a refresh is needed, a blue box will appear, containing a link for you to click:



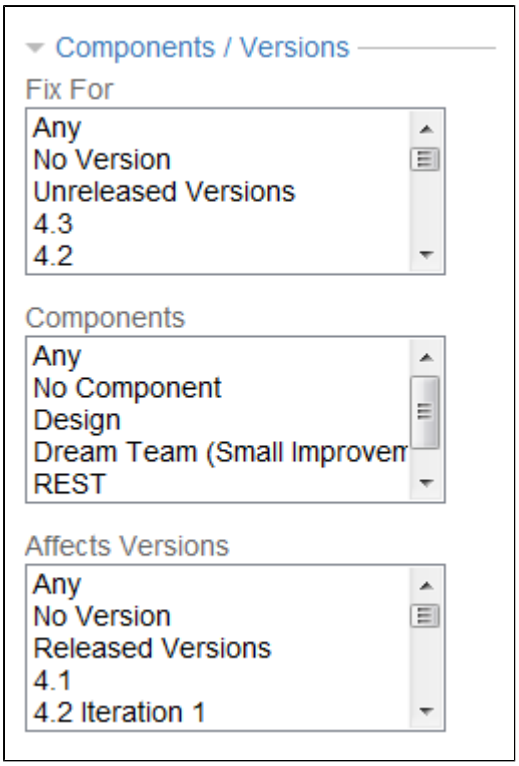
If you search on a single project, JIRA will remember that as your selected project and will default to that project on the "Browse Project" and "Create Issue" screens.

Using the 'Components/Versions' section

You can search for issues in a particular:

- **Component**
- **Affects Version**
- **Fix Version**

Screenshot: The 'Components' section



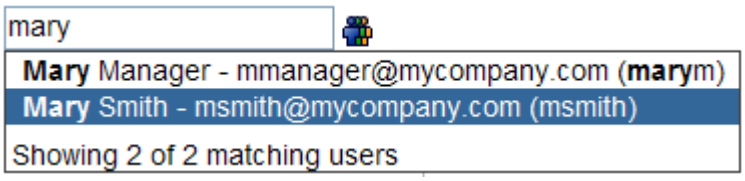
To learn more about components and versions, please see What is a Project?.

Using the 'Issue Attributes' section

You can specify particular values for the following fields:

- **Reporter**
- **Assignee**
- **Status**
- **Resolutions**
- **Priorities**
- **Labels**

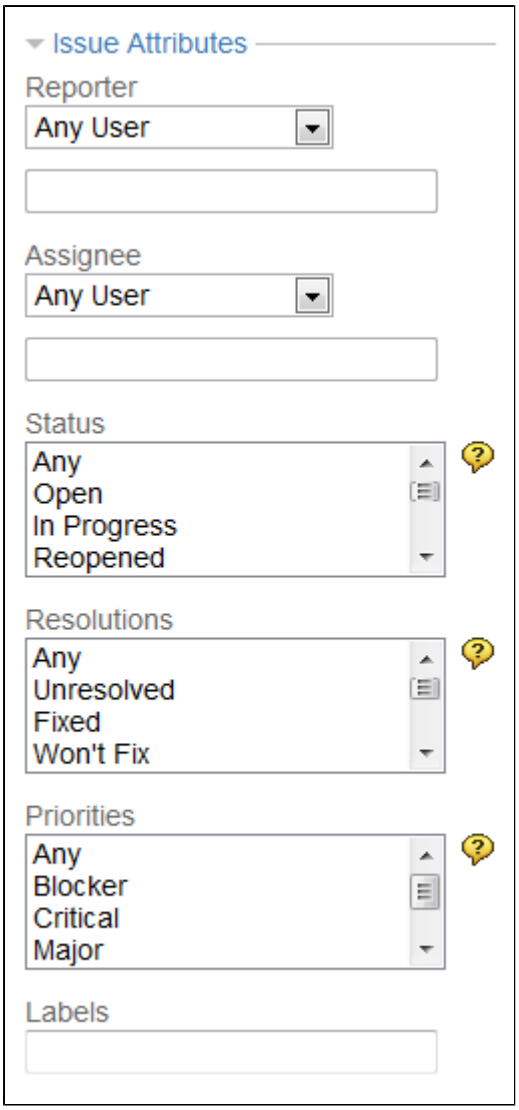
For example, to find issues reported by a particular person, select 'Specify User' in the '**Reporter**' field. In the field that appears beneath, type a few letters of the person's name (first name or surname) to display a drop-down list of matching names:



Alternatively, click the icon to display a pop-up list of all names on your JIRA system.

Please note: if the names drop-down does not display, your administrator may have disabled the 'User Picker Auto-complete' feature for your JIRA instance.

Screenshot: The 'Issue Attributes' section

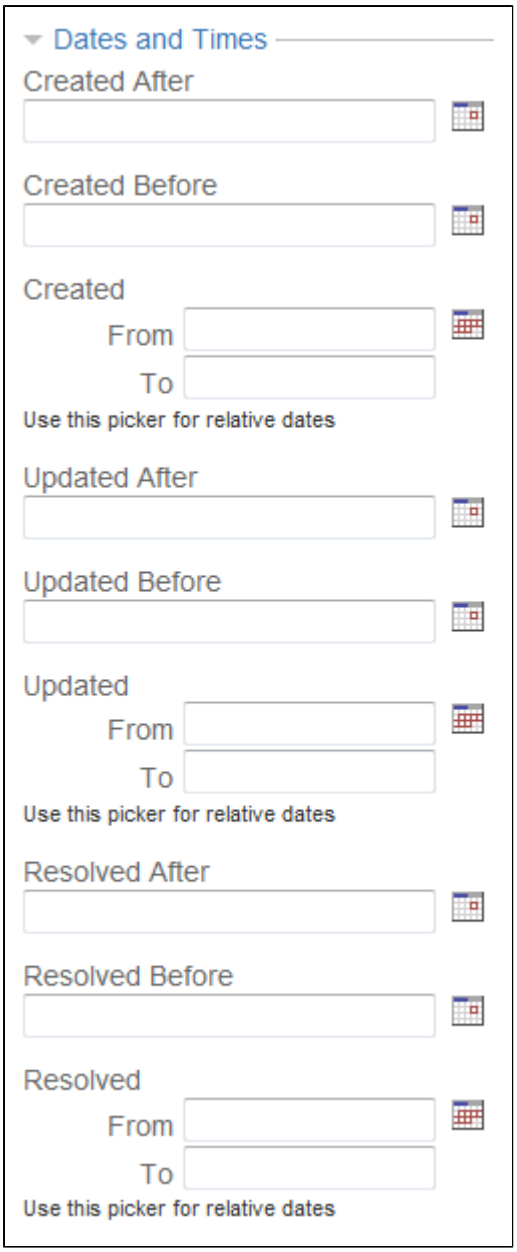


Using the 'Dates and Times' section

You can search for issues that were:

- **Created** after or before a particular date, or during a particular date range
- **Updated** after or before a particular date, or during a particular date range
- **Resolved** after or before a particular date, or during a particular date range
- **Due** after or before a particular date, or during a particular date range

Screenshot: The 'Dates and Times' section



Using the 'Work Ratio' section

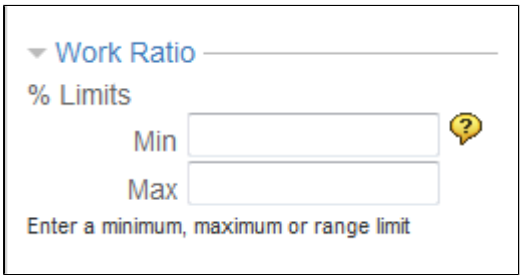
The search form contains a 'Work Ratio' section, enabling you to search JIRA issues based on time-tracking details.

The 'Work Ratio' search is based on the **Actual** work logged against an issue versus the original **Estimated** work duration.

- **Work Ratio Percentage = (Time Spent / Original Estimate) x 100**

You can enter a minimum, maximum or percentage range; the search will respectively return all issues above, below or within the specified percentage range.

Screenshot: The 'Work Ratio' section



Using the 'Custom Fields' section

Your administrator may have created custom fields for your JIRA system. Custom fields can be searched, but note that they will only appear in the search form on the left when appropriate. That is:

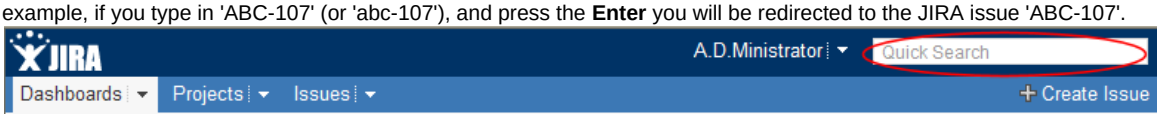
- Custom fields that relate to a particular *project* will only appear in the search form after you choose that project.
- Custom fields that relate to a particular *issue type* will only appear in the search form after you choose that issue type.

Using Quick Search

Sometimes you just want to be able to get to the particular issue that you are interested in. Other times you can't remember what the issue was, but you remember that it was an open issue, assigned to you. Quick Search can help you.

Jump to an Issue

The Quick Search box is located at the top right of your screen. If you type in the **key** of an issue, you will jump straight to that issue. For example, if you type in 'ABC-107' (or 'abc-107'), and press the **Enter** you will be redirected to the JIRA issue 'ABC-107'.



In many cases, you do not even need to type in the full key, but just the numerical part. If you are currently working on the 'ABC' project, and you type in '123', you will be redirected to 'ABC-123'.

On this page:

- Jump to an Issue
- Smart Querying
- Free-text searching
- Searching JIRA issues from your Browser's Search Box

Smart Querying

Quick Search also enables you to perform 'smart' searches with minimal typing. For example, to find all the open bugs in the 'TEST' project, you could simply type 'test open bugs', and Quick Search would locate them all for you.

Your search results will be displayed in the Issue Navigator, where you can view them in a variety of useful formats (Excel, XML, etc).

The search terms that Quick Search recognises are:

Search Term	Description	Examples
`my`	Find issues assigned to me.	`my open bugs`
`r:`	Find issues reported by you, another user or with no reporter, using the prefix *r:* followed by a specific reporter term such as *me*, a username or *none*. *Note that there can be no spaces between "r:" and the specific reporter term.*	`r:me` — finds issues reported by you. `r:samuel` — finds issues reported by the user whose username is "samuel". `r:none` — finds issues with no reporter.

`<project name>` or `<project key>`	Find issues in a particular project,	`test`
		`TST`
		`tst`

`overdue`	Find issues that were due before today.	`overdue`

`created:`
`updated:`
`due:`
	Find issues with a particular Created, Updated, or Due Date using the prefixes *created:*, *updated:*, or *due:*, respectively. For the date range, you can use *today*, *tomorrow*, *yesterday*, a single date range (e.g. '-1w'), or two date ranges (e.g. '-1w,1w'). Note that date ranges cannot have spaces in them. Valid date/time abbreviations are: 'w' (week), 'd' (day), 'h' (hour), 'm' (minute).	`created:today`
		`created:yesterday`
		`updated:-1w` — finds issues updated in the last week.
		`due:1w` — finds issues due in the next week.
		`due:-1d,1w` — finds issues due from yesterday to next week.
		`created:-1w,-30m` — finds issues created from one week ago, to 30 minutes ago.
		`created:-1d updated:-4h` — finds issues created in the last day, updated in the last 4 hours.

`<priority>`	Find issues with a particular Priority.	`blocker`
		`normal`
		`low`

`<issue type>`	Find issues with a particular Issue Type. Note that you can also use plurals.	`bug`
		`task`
		`bugs`
		`tasks`

`c:`	Find issues with a particular Component(s). You can search across multiple components. *Note that there can be no spaces between "c:" and the component name.*	`c:security` — finds issues with a component whose name contains the word "security".
`v:`	Find issues with a particular Affects Version(s). To find all issues belonging to a 'major' version, use the wildcard symbol `'*'`. *Note that there can be no spaces between "v:" and the version name.*	`v:3.0` — finds issues that match the following versions (for example): • 3.0 • 3.0 eap • 3.0 beta ...but will not match against the following versions (for example): • 3.0.1 • 3.0.0.4 That is, it will match against any version that contains the string you specify followed immediately by a space, but not against versions that do not contain a space immediately after the string you specify.
`ff:`	Find issues with a particular Fix For Version(s). Same usage as `v:` (above).	
*	Wildcard symbol `'*'`. Can be used with `v:` and `ff:`.	`v:3.2*` — finds any issue whose version number is (for example): • 3.2 • 3.2-beta • 3.2.1 • 3.2.x

In Mozilla-based browsers, try creating a bookmark with URL `http://<your-JIRA-site>/secure/QuickSearch.jspa?searchString=%s` (substituting `<your-JIRA-site>` with your JIRA site's URL) and keyword (such as *'j'*). Now, typing *'j my open bugs'* in the browser URL bar will search your JIRA site for your open bugs. Or simply type your search term in the Quick Search box, then right-click on the Quick Search box (with your search term shown) and select "Add a Keyword for this search...".

Free-text searching

You can search for any word within the issue(s) you are looking for, provided the word is in one of the following fields:

- Summary
- Description
- Comments

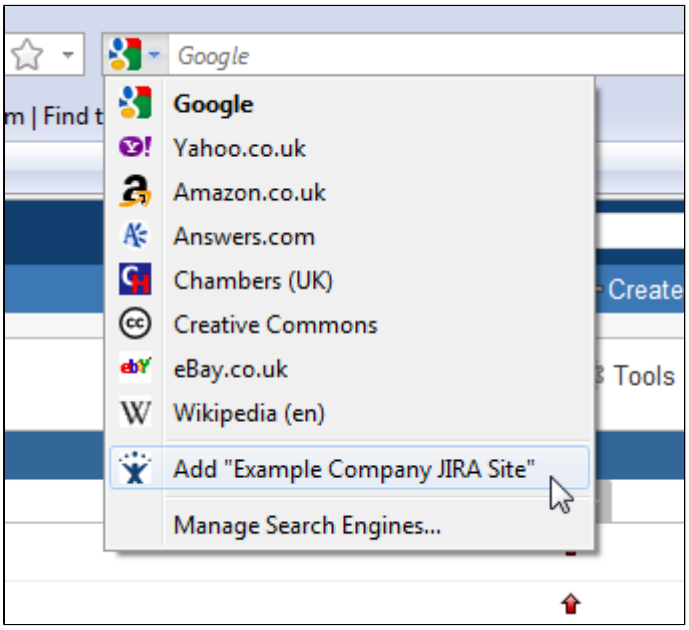
Note that, unlike the keywords listed under 'Smart Querying' above, free-text search works in both the Quick Search box *and* the Issue Filter Text Search box.

Note that you can combine free-text and keywords together. For example, *'my closed tst tasks'*, *'open test bugs pear'*, *'closed test bugs '* are all valid search queries.

Searching JIRA issues from your Browser's Search Box

If you are using Firefox or Internet Explorer 7 or later, you can add your JIRA site as a search engine/provider via the dropdown menu next to the browser's search box.

The example below shows a JIRA site called "Example Company JIRA Site", which is offered for inclusion as a search engine/provider in the browser's search box, when you visit that site.



Once you add your JIRA site as a search engine/provider in your browser, you can use it at any time to conduct a Quick Search for issues in that JIRA site.

OpenSearch

JIRA supports this browser search feature as part of the autodiscovery part of the OpenSearch standard, by supplying an OpenSearch description document. This is an XML file that describes the web interface provided by JIRA's search function. Any client applications that support OpenSearch will be able to add JIRA to their list of search engines.

Performing Text Searches

This page provides information on how to perform text searches. It applies to both simple searches and advanced searches (when used with the CONTAINS operator).

Note that this page does not apply to Quick Search.

Acknowledgements:

JIRA uses Lucene for text indexing. Lucene provides a rich query language; thanks to Jakarta and the Lucene team for such a great component. Most of the information on this page is derived from the Lucene document on Query Parser Syntax.

Query Terms

A query is broken up into **terms** and **operators.** There are two types of terms: **Single Terms** and **Phrases.**

A **Single Term** is a single word such as "`test`" or "`hello`".

A **Phrase** is a group of words surrounded by double quotes such as "`hello dolly`".

Multiple terms can be combined together with Boolean operators to form a more complex query (see below).

Note: All query terms in JIRA are case insensitive.

On this page:

- Query Terms
- Term Modifiers
  - Wildcard Searches: ? and *
  - Fuzzy Searches: ~
  - Proximity Searches
- Boosting a Term: ^
- Boolean Operators
  - OR
  - AND
  - Required term: +
  - NOT
  - Excluded term: -
- Grouping
- Escaping Special Characters: \
- Reserved Words
- Limitations
  - Whole words only

Term Modifiers

JIRA supports modifying query terms to provide a wide range of searching options.

*Wildcard Searches: ? and **

JIRA supports single and multiple character wildcard searches.

To perform a single character wildcard search use the "?" symbol.

To perform a multiple character wildcard search use the "`*`" symbol.

Wildcard characters need to be enclosed in quote-marks, as they are reserved characters in advanced search. Use quotations, e.g. `summary ~ "cha?k and che*"`

The single character wildcard search looks for terms that match that with the single character replaced. For example, to search for "`text`" or "`test`" you can use the search:

```
te?t
```

Multiple character wildcard searches looks for 0 or more characters. For example, to search for `Windows`, `Win95` or `WindowsNT` you can use the search:

```
win*
```

You can also use the wildcard searches in the middle of a term. For example, to search for `Win95` or `Windows95` you can use the search

```
wi*95
```

You cannot use a * or ? symbol as the first character of a search.

Fuzzy Searches: ~

JIRA supports fuzzy searches. To do a fuzzy search use the tilde, "~", symbol at the end of a Single word Term. For example to search for a term similar in spelling to "`roam`" use the fuzzy search:

```
roam~
```

This search will find terms like foam and roams.

Note: Terms found by the fuzzy search will automatically get a boost factor of 0.2

Proximity Searches

JIRA supports finding words are a within a specific distance away. To do a proximity search use the tilde, "~", symbol at the end of a Phrase. For example to search for a "`atlassian`" and "`jira`" within 10 words of each other in a document use the search:

```
"atlassian jira"~10
```

Boosting a Term: ^

JIRA provides the relevance level of matching documents based on the terms found. To boost a term use the caret, "^", symbol with a boost factor (a number) at the end of the term you are searching. The higher the boost factor, the more relevant the term will be.

Boosting allows you to control the relevance of a document by boosting its term. For example, if you are searching for

```
atlassian jira
```

and you want the term "`atlassian`" to be more relevant boost it using the ^ symbol along with the boost factor next to the term. You would type:

```
atlassian^4 jira
```

This will make documents with the term atlassian appear more relevant. You can also boost Phrase Terms as in the example:

```
"atlassian jira"^4 querying
```

By default, the boost factor is 1. Although, the boost factor must be positive, it can be less than 1 (i.e. .2)

Boolean Operators

Boolean operators allow terms to be combined through logic operators. JIRA supports AND, "+", OR, NOT and "-" as Boolean operators.

Boolean operators must be ALL CAPS.

OR

The OR operator is the default conjunction operator. This means that if there is no Boolean operator between two terms, the OR operator is used. The OR operator links two terms and finds a matching document if either of the terms exist in a document. This is equivalent to a union using sets. The symbol `||` can be used in place of the word OR.

To search for documents that contain either "`atlassian jira`" or just "`jira`" use the query:

```
"atlassian jira" || jira
```

or

```
"atlassian jira" OR jira
```

AND

The AND operator matches documents where both terms exist anywhere in the text of a single document. This is equivalent to an intersection using sets. The symbol && can be used in place of the word AND.

To search for documents that contain "`atlassian jira`" and "`issue tracking`" use the query:

```
"atlassian jira" AND "issue tracking"
```

Required term: +

The "+" or required operator requires that the term after the "+" symbol exist somewhere in a the field of a single document.

To search for documents that must contain "`jira`" and may contain "`atlassian`" use the query:

```
+jira atlassian
```

NOT

The NOT operator excludes documents that contain the term after NOT. This is equivalent to a difference using sets. The symbol `!` can be used in place of the word NOT.

To search for documents that contain "`atlassian jira`" but not "`japan`" use the query:

```
"atlassian jira" NOT "japan"
```

Note: The NOT operator cannot be used with just one term. For example, the following search will return no results:

```
NOT "atlassian jira"
```

Usage of the **NOT** operator over multiple fields may return results that include the specified excluded term. This is due to the fact that the search query is executed over each field in turn and the result set for each field is combined to form the final result set. Hence, an issue that matches the search query based on one field, but fails based on another field, will be included in the search result set.

Excluded term: -

The "-" or prohibit operator excludes documents that contain the term after the "-" symbol.

To search for documents that contain "atlassian jira" but not "japan" use the query:

```
"atlassian jira" -japan
```

Grouping

JIRA supports using parentheses to group clauses to form sub queries. This can be very useful if you want to control the boolean logic for a query.

To search for either "atlassian" or "jira" and "bugs" use the query:

```
bugs AND (atlassian OR jira)
```

This eliminates any confusion and makes sure you that bugs must exist and either term atlassian or jira may exist.

Do not use the grouping character '(' at the start of a search query, as this will result in an error. For example, "(atlassian OR jira) AND bugs" will not work.

Escaping Special Characters: \

JIRA supports escaping special characters that are part of the query syntax. The current list of special characters is:

```
+ - && || ! ( ) { } [ ] ^ ~ * ? \
```

To escape these characters, use the \ before the character. For example, to search for (1+1) use the query:

```
\(1\+1\)
```

Please note: the syntax for escaping special characters is different if you are using Advanced Searching — please see Reserved Characters .

Reserved Words

Certain *common words* are ignored from the search and search index.

Note that this can sometimes lead to unexpected results. For example, suppose one issue contains the words "VSX will crash" and another issue contains the words "VSX will not crash". A phrase search for "VSX will crash" will return both of the issues. This is because the words will and not are part of the reserved words list.

The full list of reserved English words is:

```
"a", "and", "are", "as", "at", "be", "but", "by", "for", "if", "in", "into", "is", "it", "no", "not",
"of", "on", "or", "s", "such", "t", "that", "the", "their", "then", "there", "these", "they", "this",
"to", "was", "will", "with"
```

Note that your JIRA Administrator can alter the behavior of JIRA in relation to these reserved words by changing the **Indexing Language** from "English" to "Other" under Administration > General Configuration.

Limitations

Please note that the following limitations of Lucene apply to JIRA:

Whole words only

You cannot search on word parts, only on whole words.

Advanced Searching

What is an Advanced Search?

An advanced search allows you to use structured queries to search for JIRA issues. Your search results will be displayed in the Issue Navigator, where you can export them to MS Excel and many other formats. You can also save and subscribe to your advanced searches if you wish.

When you perform an advanced search, you are using the JIRA Query Language (JQL).

A simple query in JQL (also known as a 'clause') consists of a *field*, followed by an *operator*, followed by one or more *values* or *functions*. Fo example, the following simple query will find all issues in the "TEST" project:

```
project = "TEST"
```

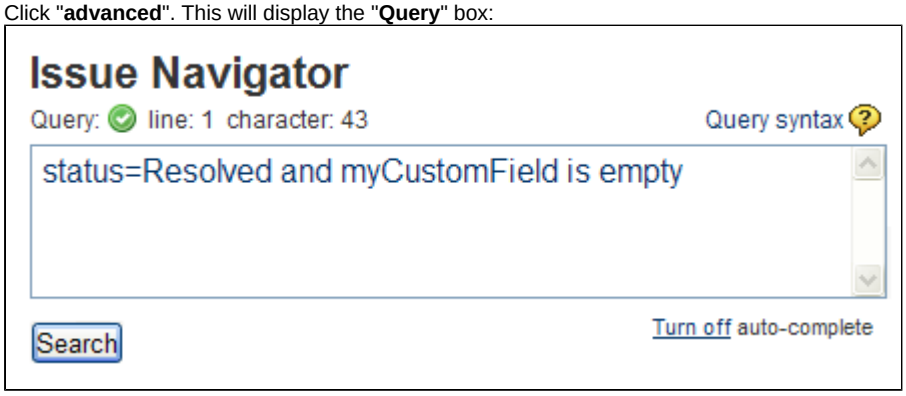
(This example uses the Project *field*, the EQUALS *operator*, and the *value* `"TEST"`.)

Be aware that it is not possible to compare two fields.

JQL gives you some SQL-like syntax, such as the ORDER BY SQL keyword and ISNULL() SQL function (i.e. the NULL keyword in JQL) However, JQL is not a database query language. For example, JQL does not have a `SELECT` statement.

How to Perform an Advanced Search

1. On the top navigation bar, click the "**Issues**" tab. This will display the Search panel.
2. Click "**advanced**". This will display the "**Query**" box:



3. Type your query using the fields, operators and field values or functions listed below.
4. Click the "**Search**" button to run your query.

On this page:

- What is an Advanced Search?
- How to Perform an Advanced Search
  - Keywords Reference
  - Operators Reference
  - Fields Reference
  - Functions Reference
  - Setting Precedence of Operators
  - Performing Text Searches
  - Using Auto-complete
  - Switching between 'Advanced' and 'Simple' Search
  - Reserved Characters
  - Reserved Words

Keywords Reference

A keyword in JQL is a word or phrase that does (or is) any of the following:

- joins two or more clauses together to form a complex JQL query
- alters the logic of one or more clauses
- alters the logic of operators
- has an explicit definition in a JQL query
- performs a specific function that alters the results of a JQL query.

^^top of topic

List of keywords:

- AND
- OR
- NOT
- EMPTY
- NULL
- ORDER BY

AND

Used to combine multiple clauses, allowing you to refine your search.

Note that you can use parentheses to control the order in which clauses are executed.

Examples

- Find all open issues in the "New office" project:

```
project = "New office" and status = "open"
```

- Find all open, urgent issues that are assigned to jsmith:

```
status = open and priority = urgent and assignee = jsmith
```

- Find all issues in a particular project that are not assigned to jsmith:

```
project = JRA and assignee != jsmith
```

- Find all issues for a specific release which consists of different version numbers across several projects:

```
project in (JRA,CONF) and fixVersion = "3.14"
```

- Find all issues where neither the Reporter nor the Assignee is Jack, Jill or John:

```
reporter not in (Jack,Jill,John) and assignee not in (Jack,Jill,John)
```

^top of keywords | ^^top of topic

OR

Used to combine multiple clauses, allowing you to expand your search.

Note that you can use parentheses to control the order in which clauses are executed.

(Note: also see IN, which can be a more convenient way to search for multiple values of a field.)

Examples

- Find all issues that were created by either jsmith or jbrown:

```
reporter = jsmith or reporter = jbrown
```

- Find all issues that are overdue or where no due date is set:

```
duedate < now() or duedate is empty
```

^top of keywords | ^^top of topic

NOT

Used to negate individual clauses or a complex JQL query (a query made up of more than one clause) using parentheses, allowing you to refine your search.

(Note: also see NOT EQUALS ("!="), DOES NOT CONTAIN ("!~"), NOT IN and IS NOT.)

Examples

- Find all issues that are assigned to any user except jsmith:

```
not assignee = jsmith
```

- Find all issues that were not created by either jsmith or jbrown:

```
not (reporter = jsmith or reporter = jbrown)
```

^top of keywords | ^^top of topic

EMPTY

Used to search for issues where a given field does not have a value. See also NULL.

Note that EMPTY can only be used with fields that support the IS and IS NOT operators. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues without a DueDate:

```
duedate = empty
```

or

```
duedate is empty
```

^top of keywords | ^^top of topic

NULL

Used to search for issues where a given field does not have a value. See also EMPTY.

Note that NULL can only be used with fields that support the IS and IS NOT operators. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues without a DueDate:

```
duedate = null
```

or

```
duedate is null
```

^top of keywords | ^^top of topic

ORDER BY

Used to specify the fields by whose values the search results will be sorted.

By default, the field's own sorting order will be used. You can override this by specifying ascending order ("`asc`") or descending order ("`desc`").

Examples

- Find all issues without a DueDate, sorted by CreationDate:

```
duedate = empty order by created
```

- Find all issues without a DueDate, sorted by CreationDate, then by Priority (highest to lowest):

```
duedate = empty order by created, priority desc
```

- Find all issues without a DueDate, sorted by CreationDate, then by Priority (lowest to highest):

```
duedate = empty order by created, priority asc
```

^top of keywords | ^^top of topic

Operators Reference

An operator in JQL is one or more symbols or words which compares the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved by the clause. Some operators may use the NOT keyword.

^^top of topic

List of Operators:

- EQUALS: =
- NOT EQUALS: !=
- GREATER THAN: >
- GREATER THAN EQUALS: >=
- LESS THAN: <
- LESS THAN EQUALS: <=
- IN
- NOT IN
- CONTAINS: ~
- DOES NOT CONTAIN: !~
- IS
- IS NOT
- WAS
- WAS IN
- WAS NOT IN
- WAS NOT
- CHANGED

EQUALS: =

The "=" operator is used to search for issues where the value of the specified field exactly matches the specified value. (Note: cannot be used with text fields; see the CONTAINS operator instead.)

To find issues where the value of a specified field exactly matches *multiple* values, use multiple "=" statements with the AND operator.

Examples

- Find all issues that were created by jsmith:

```
reporter = jsmith
```

- Find all issues that were created by John Smith:

```
reporter = "John Smith"
```

^top of operators | ^^top of topic

NOT EQUALS: !=

The "!=" operator is used to search for issues where the value of the specified field does not match the specified value. (Note: cannot be used with text fields; see the DOES NOT MATCH ("!~") operator instead.)

Note that typing `field != value` is the same as typing `NOT field = value`, and that `field != EMPTY` is the same as `field IS_NOT EMPTY`.

The "!=" operator will not match a field that has no value (i.e. a field that is empty). For example, `component != fred` will only match issues that have a component **and** the component is not "fred". To find issues that have a component other than "fred" **or have no component**, you would need to type: `component != fred or component is empty`.

Examples

- Find all issues that are assigned to any user except jsmith:

```
not assignee = jsmith
```

  or:

```
assignee != jsmith
```

- Find all issues that are not assigned to jsmith:

```
assignee != jsmith or assignee is empty
```

- Find all issues that were reported by me but are not assigned to me:

```
reporter = currentUser() and assignee != currentUser()
```

- Find all issues where the Reporter or Assignee is anyone except John Smith:

```
assignee != "John Smith" or reporter != "John Smith"
```

- Find all issues that are not unassigned:

```
assignee is not empty
```

  or

```
assignee != null
```

^top of operators | ^^top of topic

GREATER THAN: >

The ">" operator is used to search for issues where the value of the specified field is greater than the specified value. Cannot be used with text fields.

Note that the ">" operator can only be used with fields which support ordering (e.g. date fields and version fields). To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with more than 4 votes:

```
votes > 4
```

- Find all overdue issues:

```
duedate < now() and resolution is empty
```

- Find all issues where priority is higher than "Normal":

```
priority > normal
```

^top of operators | ^^top of topic

GREATER THAN EQUALS: >=

The ">=" operator is used to search for issues where the value of the specified field is greater than or equal to the specified value. Cannot be used with text fields.

Note that the ">=" operator can only be used with fields which support ordering (e.g. date fields and version fields). To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with 4 or more votes:

```
votes >= 4
```

- Find all issues due on or after 31/12/2008:

```
duedate >= "2008/12/31"
```

- Find all issues created in the last five days:

```
created >= "-5d"
```

^top of operators | ^^top of topic

LESS THAN: <

The "<" operator is used to search for issues where the value of the specified field is less than the specified value. Cannot be used with text fields.

Note that the "<" operator can only be used with fields which support ordering (e.g. date fields and version fields). To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with less than 4 votes:

```
votes < 4
```

^top of operators | ^^top of topic

LESS THAN EQUALS: <=

The "<=" operator is used to search for issues where the value of the specified field is less than or equal to than the specified value. Cannot be used with text fields.

Note that the "<=" operator can only be used with fields which support ordering (e.g. date fields and version fields). To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with 4 or fewer votes:

```
votes <= 4
```

- Find all issues that have not been updated in the past month (30 days):

```
updated <= "-4w 2d"
```

^top of operators | ^^top of topic

IN

The "`IN`" operator is used to search for issues where the value of the specified field is one of multiple specified values. The values are specified as a comma-delimited list, surrounded by parentheses.

Using "`IN`" is equivalent to using multiple `EQUALS (=)` statements, but is shorter and more convenient. That is, typing `reporter IN (tom, jane, harry)` is the same as typing `reporter = "tom" OR reporter = "jane" OR reporter = "harry"`.

Examples

- Find all issues that were created by either jsmith or jbrown or jjones:

```
reporter in (jsmith,jbrown,jjones)
```

- Find all issues where the Reporter or Assignee is either Jack or Jill:

```
reporter in (Jack,Jill) or assignee in (Jack,Jill)
```

- Find all issues in version 3.14 or version 4.2:

```
affectedVersion in ("3.14", "4.2")
```

^top of operators | ^^top of topic

NOT IN

The "`NOT IN`" operator is used to search for issues where the value of the specified field is not one of multiple specified values.

Using "`NOT IN`" is equivalent to using multiple `NOT_EQUALS (!=)` statements, but is shorter and more convenient. That is, typing `reporter NOT IN (tom, jane, harry)` is the same as typing `reporter != "tom" AND reporter != "jane" AND reporter != "harry"`.

The "`NOT IN`" operator will not match a field that has no value (i.e. a field that is empty). For example, `assignee not in (jack,jill)` will only match issues that have an assignee **and** the assignee is not "jack" or "jill". To find issues that are assigned to someone other than "jack" or "jill" **or are unassigned**, you would need to type: `assignee not in (jack,jill) or assignee is empty`.

Examples

- Find all issues where the Assignee is someone other than Jack, Jill or John:

```
assignee not in (Jack,Jill,John)
```

- Find all issues where the Assignee is not Jack, Jill or John:

```
assignee not in (Jack,Jill,John) or assignee is empty
```

- Find all issues where the FixVersion is not 'A', 'B', 'C' or 'D':

```
FixVersion not in (A, B, C, D)
```

- Find all issues where the FixVersion is not 'A', 'B', 'C' or 'D', or has not been specified:

```
FixVersion not in (A, B, C, D) or FixVersion is empty
```

^top of operators | ^^top of topic

CONTAINS: ~

The "~" operator is used to search for issues where the value of the specified field matches the specified value (either an exact match or a "fuzzy" match — see examples below). For use with text fields only, i.e.:

- Summary
- Description
- Environment
- Comments
- custom fields which use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
  - Free Text Field (unlimited text)
  - Text Field (< 255 characters)
  - Read-only Text Field

Note: when using the "~" operator, the value on the right-hand side of the operator can be specified using JIRA text-search syntax.

Examples

- Find all issues where the Summary contains the word "win" (or simple derivatives of that word, such as "wins"):

```
summary ~ win
```

- Find all issues where the Summary contains a wild-card match for the word "win":

```
summary ~ "win*"
```

- Find all issues where the Summary contains the exact phrase "full screen" (see Reserved Characters for details on how to escape quote-marks and other special characters):

```
summary ~ "\"full screen\""
```

^top of operators | ^^top of topic

DOES NOT CONTAIN: !~

The "!~" operator is used to search for issues where the value of the specified field is not a "fuzzy" match for the specified value. For use with text fields only, i.e.:

- Summary
- Description

- Environment
- Comments
- custom fields which use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
  - Free Text Field (unlimited text)
  - Text Field (< 255 characters)
  - Read-only Text Field

Note: when using the "`!~`" operator, the value on the right-hand side of the operator can be specified using JIRA text-search syntax.

Examples

- Find all issues where the Summary does not contain the word "run" (or derivatives of that word, such as "running" or "ran"):

```
summary !~ run
```

^top of operators | ^^top of topic

IS

The "`IS`" operator can only be used with EMPTY or NULL. That is, it is used to search for issues where the specified field has no value.

Note that not all fields are compatible with this operator; see the individual field reference for details.

Examples

- Find all issues that have no Fix Version:

```
fixVersion is empty
```

or

```
fixVersion is null
```

^top of operators | ^^top of topic

IS NOT

The "`IS NOT`" operator can only be used with EMPTY or NULL. That is, it is used to search for issues where the specified field has a value.

Note that not all fields are compatible with this operator; see the individual field reference for details.

Examples

- Find all issues that have one or more votes:

```
votes is not empty
```

or

```
votes is not null
```

^top of operators | ^^top of topic

WAS

The "`WAS`" operator is used to find issues that currently have, or previously had, the specified value for the specified field.

This operator has the following optional predicates:

- `AFTER "date"`
- `BEFORE "date"`
- `BY "username"`

- `DURING ("date1","date2")`
- `ON "date"`

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution and Status fields only.)

Examples

- Find issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

- Find issues that were resolved by Joe Smith before 2nd February:

```
status WAS "Resolved" BY jsmith BEFORE "02/02/2011"
```

- Find issues that were resolved by Joe Smith during 2010:

```
status WAS "Resolved" BY jsmith DURING ("01/01/2010","01/01/2011")
```

^top of operators | ^^top of topic

WAS IN

The "`WAS IN`" operator is used to find issues that currently have, or previously had, any of multiple specified values for the specified field. The values are specified as a comma-delimited list, surrounded by parentheses.

Using "`WAS IN`" is equivalent to using multiple `WAS` statements, but is shorter and more convenient. That is, typing `status WAS IN ('Resolved', 'Closed')` is the same as typing `status WAS "Resolved" OR status WAS "Closed"`.

This operator has the following optional predicates:

- `AFTER "date"`
- `BEFORE "date"`
- `BY "username"`
- `DURING ("date1","date2")`
- `ON "date"`

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution and Status fields only.)

Examples

- Find all issues that currently have, or previously had, a status of 'Resolved' or 'In Progress':

```
status WAS IN ("Resolved","In Progress")
```

^top of operators | ^^top of topic

WAS NOT IN

The "`WAS NOT IN`" operator is used to search for issues where the value of the specified field has never been one of multiple specified values.

Using "`WAS NOT IN`" is equivalent to using multiple `WAS_NOT` statements, but is shorter and more convenient. That is, typing `status WAS NOT IN ("Resolved","In Progress")` is the same as typing `status WAS NOT "Resolved" AND status WAS NOT "In Progress"`.

This operator has the following optional predicates:

- `AFTER "date"`
- `BEFORE "date"`

- BY "username"
- DURING ("date1","date2")
- ON "date"

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution and Status fields only.)

Examples

- Find issues that have never had a status of 'Resolved' or 'In Progress':

```
status WAS NOT IN ("Resolved","In Progress")
```

- Find issues that did not have a status of 'Resolved' or 'In Progress' before 2nd February:

```
status WAS NOT IN ("Resolved","In Progress") BEFORE "02/02/2011"
```

^top of operators | ^^top of topic

WAS NOT

The "WAS NOT" operator is used to find issues that have never had the specified value for the specified field.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"
- BY "username"
- DURING ("date1","date2")
- ON "date"

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution and Status fields only.)

Examples

- Find issues that do not have, and has never had, a status of 'In Progress':

```
status WAS NOT "In Progress"
```

- Find issues that did not have a status of 'In Progress' before 2nd February:

```
status WAS NOT "In Progress" BEFORE "02/02/2011"
```

^top of operators | ^^top of topic

CHANGED

The "CHANGED" operator is used to find issues that have a value which had changed for the specified field.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"
- BY "username"
- DURING ("date1","date2")
- ON "date"
- FROM "oldvalue"

- `TO "newvalue"`

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution and Status fields only.)

Examples

- Find issues whose assignee had changed:

```
assignee CHANGED
```

- Find issues whose status had changed from 'In Progress' back to 'Open':

```
status CHANGED FROM "In Progress" TO "Open"
```

- Find issues whose priority was changed by user 'freddo' after the start and before the end of the current week.

```
priority CHANGED BY freddo BEFORE endOfWeek() AFTER startOfWeek()
```

^top of operators | ^^top of topic

Fields Reference

A field in JQL is a word that represents a JIRA field (or a custom field that has already been defined in JIRA). In a clause, a field is followed by an operator, which in turn is followed by one or more values (or functions). The operator compares the value of the field with one or more values or functions on the right, such that only true results are retrieved by the clause.

^^top of topic

List of Fields:

- Affected Version
- Assignee
- Category
- Comment
- Component
- Created
- Custom Field
- Description
- Due
- Environment
- Filter
- Fix Version
- Issue Key
- Level
- Original Estimate
- Parent
- Priority
- Project
- Remaining Estimate
- Reporter
- Resolution
- Resolved
- Status
- Summary
- Text
- Type
- Time Spent
- Updated
- Voter
- Votes
- Watcher
- Watchers
- Work Ratio

Affected Version

Search for issues that are assigned to a particular Affects Version(s). You can search by version name or version ID (i.e. the number that JIRA automatically allocates to a version).

It is safer to search by version ID than by version name

Different projects may have versions with the same name, so searching by version name may return issues from multiple projects. It is also possible for your JIRA administrator to change the name of a version, which could break any saved filters which rely on that name. Version IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
affectedVersion
```

Field Type

VERSION

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Note that the comparison operators (e.g. ">") use the version order that has been set up by your project administrator, not a numeric or alphabetic order.

Supported FunctionsWhen used with the IN and NOT IN operators, this field supports:

- releasedVersions()
- latestReleasedVersion()
- unreleasedVersions()
- earliestUnreleasedVersion()

Examples

- Find issues with an AffectedVersion of 3.14:

  ```
  affectedVersion = "3.14"
  ```

  (Note that full-stops are reserved characters, so they need to be surrounded by quote marks.)

- Find issues with an AffectedVersion of "Big Ted":

  ```
  affectedVersion = "Big Ted"
  ```

- Find issues with an AffectedVersion ID of 10350:

  ```
  affectedVersion = 10350
  ```

^top of fields | ^^top of topic

Assignee

Search for issues that are assigned to a particular user. You can search by the user's Full Name, ID or Email Address.

Note: this field supports auto-complete.

Syntax

```
assignee
```

Field Type

USER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	✅

Supported FunctionsWhen used with the IN and NOT IN operators, this field supports:

- membersOf()

When used with the EQUALS and NOT EQUALS operators, this field supports:

- currentUser()

Examples

- Find issues that are assigned to John Smith:

```
assignee = "John Smith"
```

or

```
assignee = jsmith
```

- Find issues that are currently assigned, or were previously assigned, to John Smith:

```
assignee WAS "John Smith"
```

or

```
assignee WAS jsmith
```

- Find issues that are assigned by the user with email address "bob@mycompany.com":

```
assignee = "bob@mycompany.com"
```

(Note that full-stops and "@" symbols are reserved characters, so the email address needs to be surrounded by quote-marks.)

^top of fields | ^^top of topic

Category

Search for issues that belong to projects in a particular Category.

Note: this field supports auto-complete.

Syntax

```
category
```

Field Type

CATEGORY

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues that belong to projects in the "Alphabet Projects" Category:

```
category = "Alphabet Projects"
```

^top of fields | ^^top of topic

Comment

Search for issues that have a Comment which contains particular text.

JIRA text-search syntax can be used.

Note: this field does not support auto-complete.

Syntax

```
comment
```

Field Type

TEXT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	✅	✅	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues where a Comment contains text that matches "My PC is quite old" (i.e. a "fuzzy" match:

```
comment ~ "My PC is quite old"
```

- Find issues where a Comment contains the exact phrase "My PC is quite old":

```
comment ~ "\"My PC is quite old\""
```

^top of fields | ^^top of topic

Component

Search for issues that belong to a particular component(s) of a project. You can search by component name or component ID (i.e. the number that JIRA automatically allocates to a component).

> **It is safer to search by component ID than by component name**
>
> Different projects may have components with the same name, so searching by component name may return issues from multiple projects. It is also possible for your JIRA administrator to change the name of a component, which could break any saved filters which rely on that name. Component IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
component
```

Field Type

COMPONENT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

When used with the IN and NOT IN operators, **`component`** supports:

- componentsLeadByUser()

Examples

- Find issues in the "Comp1" or "Comp2" component:

```
component in (Comp1, Comp2)
```

- Find issues in the "Comp1" *and* "Comp2" components:

```
component in (Comp1) and component in (Comp2)
```

  or

```
component = Comp1 and component = Comp2
```

- Find issues in the component with ID 20500:

```
component = 20500
```

^top of fields | ^^top of topic

Created

Search for issues that were created on, before or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the JIRA server's time zone).

Use one of the following formats:

```
"yyyy/MM/dd HH:mm"
"yyyy-MM-dd HH:mm"
"yyyy/MM/dd"
"yyyy-MM-dd"
```

Or use `"w"` (weeks), `"d"` (days), `"h"` (hours) or `"m"` (minutes) to specify a date relative to the current time. The default is `"m"` (minutes). Be sure to use quote-marks ("); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Note: this field does not support auto-complete.

Syntax

```
created
```

Alias:

```
createdDate
```

Field Type

DATE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported FunctionsWhen used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:

- currentLogin()
- lastLogin()
- now()
- startOfDay()
- startOfWeek()
- startOfMonth()
- startOfYear()
- endOfDay()
- endOfWeek()
- endOfMonth()
- endOfYear()

Examples

- Find all issues created before 12th December 2010:

```
created < "2010/12/12"
```

- Find all issues created on or before 12th December 2010:

```
created <= "2010/12/13"
```

- Find all issues created on 12th December 2010 before 2:00pm:

```
created > "2010/12/12" and created < "2010/12/12 14:00"
```

- Find issues created less than one day ago:

```
created > "-1d"
```

- Find issues created in January 2011:

```
created > "2011/01/01" and created < "2011/02/01"
```

- Find issues created on 15 January 2011:

```
created > "2011/01/15" and created < "2011/01/16"
```

^top of fields | ^^top of topic

Custom Field

Only applicable if your JIRA administrator has created one or more Custom Fields.

Search for issues where a particular Custom Field has a particular value.

You can search by Custom Field name or Custom Field ID (i.e. the number that JIRA automatically allocates to an Custom Field).

> **It is safer to search by Custom Field ID than by Custom Field name**
>
> It is possible for a Custom Field to have the same name as a built-in JIRA system field, in which case JIRA will search on the system field (not your custom field). It is also possible for your JIRA administrator to change the name of a Custom Field, which could break any saved filters which rely on that name. Custom Field IDs, however, are unique and cannot be changed.

Note:

- JIRA text-search syntax can be used with Custom Fields of type 'Text'.
- auto-complete is supported for Custom Fields of type picker, group picker, select (except 'Cascading Select'), check-box and radio button fields.

Syntax

```
CustomFieldName
```

Alias:

```
cf[CustomFieldID]
```

Field Type

Depends on the Custom Field's configuration

Supported Operators

Different types of Custom Fields support different operators. For the default Custom Field Types, the following operators are supported:

- Number and date/time fields:

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

- Picker, select, check-box and radio button fields:

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

- Text fields:

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	✅	✅	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌	❌	❌

Supported Functions

Different types of Custom Fields support different functions. For the default Custom Field Types, the following functions are supported:

- Date/time fields: When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:
  - currentLogin()
  - lastLogin()
  - now()
  - startOfDay()
  - startOfWeek()
  - startOfMonth()
  - startOfYear()
  - endOfDay()
  - endOfWeek()
  - endOfMonth()
  - endOfYear()
- Version picker fields: When used with the IN and NOT IN operators, this field supports:
  - releasedVersions()
  - latestReleasedVersion()
  - unreleasedVersions()
  - earliestUnreleasedVersion()

Examples

- Find issues where the value of the "Location" Custom Field is "New York":

```
location = "New York"
```

- Find issues where the value of the Custom Field with ID 10003 is "New York":

```
cf[10003] = "New York"
```

- Find issues where the value of the "Location" Custom Field is "London" or "Milan" or "Paris":

```
cf[10003] in ("London", "Milan", "Paris")
```

- Find issues where the "Location" Custom Field has no value:

```
location != empty
```

^top of fields | ^^top of topic

Description

Search for issues where the Description contains particular text.

JIRA text-search syntax can be used.

Note: this field does not support auto-complete.

Syntax

```
description
```

Field Type

TEXT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	✅	✅	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues where the Description contains text that matches "Please see screenshot" (i.e. a "fuzzy" match):

```
description ~ "Please see screenshot"
```

- Find issues where the Description contains the exact phrase "Please see screenshot":

```
description ~ "\"Please see screenshot\""
```

^top of fields | ^^top of topic

Due

Search for issues that were due on, before or after a particular date (or date range). Note that Due Date relates to the *date* only (not to the time).

Use one of the following formats:

`"yyyy/MM/dd"`
`"yyyy-MM-dd"`

Or use `"w"` (weeks) or `"d"` (days) to specify a date relative to the current date. Be sure to use quote-marks (").

Note: this field does not support auto-complete.

Syntax

```
due
```

Alias:

```
dueDate
```

Field Type

DATE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported FunctionsWhen used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:

- currentLogin()
- lastLogin()
- now()
- startOfDay()
- startOfWeek()
- startOfMonth()
- startOfYear()
- endOfDay()
- endOfWeek()
- endOfMonth()
- endOfYear()

Examples

- Find all issues due before 31st December 2010:

```
due < "2010/12/31"
```

- Find all issues due on or before 31st December 2010:

```
due <= "2011/01/01"
```

- Find all issues due tomorrow:

```
due = "1d"
```

- Find all issues due in January 2011:

```
due >= "2011/01/01" and due <= "2011/01/31"
```

- Find all issues due on 15 January 2011:

```
due = "2011/01/15"
```

^top of fields | ^^top of topic

Environment

Search for issues where the Environment contains particular text.

JIRA text-search syntax can be used.

Note: this field does not support auto-complete.

Syntax

```
environment
```

Field Type

TEXT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✗	✗	✓	✓	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗

Supported Functions

n/a

Examples

- Find issues where the Environment contains text that matches "Third floor" (i.e. a "fuzzy" match):

```
environment ~ "Third floor"
```

- Find issues where the Environment contains the exact phrase "Third floor":

```
environment ~ "\"Third floor\""
```

^top of fields | ^^top of topic

Filter

You can use a saved filter to narrow your search. You can search by filter name or filter ID (i.e. the number that JIRA automatically allocates to a saved filter).

> **It is safer to search by filter ID than by filter name**
>
> It is possible for a filter name to be changed, which could break a saved filter that invokes another filter by name. Filter IDs, however, are unique and cannot be changed.

Note:

- An Advanced Searching statement in your typed query will override an ORDER BY statement in the saved filter.
- You cannot run or save a filter that would cause an infinite loop (i.e. you cannot reference a saved filter if it eventually references your current filter).
- This field supports auto-complete.Syntax

```
filter
```

Aliases:

```
request
```

```
savedFilter
```

```
searchRequest
```

Field Type

FILTER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Search the results of the filter "My Saved Filter" (which has an ID of 12000) for issues assigned to the user jsmith:

```
filter = "My Saved Filter" and assignee = jsmith
```

or

```
filter = 12000 and assignee = jsmith
```

^top of fields | ^^top of topic

Fix Version

Search for issues that are assigned to a particular Fix Version. You can search by version name or version ID (i.e. the number that JIRA automatically allocates to a version).

> **It is safer to search by version ID than by version name**
>
> Different projects may have versions with the same name, so searching by version name may return issues from multiple projects. It is also possible for your JIRA administrator to change the name of a version, which could break any saved filters that rely on that name. Version IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
fixVersion
```

Field Type

VERSION

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅

Note that the comparison operators (e.g. ">") use the version order that has been set up by your project administrator, not a numeric or alphabetic order.

Supported FunctionsWhen used with the IN and NOT IN operators, this field supports:

- releasedVersions()
- latestReleasedVersion()
- unreleasedVersions()
- earliestUnreleasedVersion()

Examples

- Find issues with a Fix Version of 3.14 or 4.2:

```
fixVersion in ("3.14", "4.2")
```

  (Note that full-stops are reserved characters, so they need to be surrounded by quote marks.)

- Find issues with a Fix Version of "Little Ted":

```
fixVersion = "Little Ted"
```

- Find issues with a Fix Version ID of 10001:

```
fixVersion = 10001
```

^top of fields | ^^top of topic

Issue Key

Search for issues with a particular Issue Key or Issue ID (i.e. the number that JIRA automatically allocates to an Issue).

Note: this field does not support auto-complete.

Syntax

```
issueKey
```

Aliases:

```
id
```

```
issue
```

```
key
```

Field Type

ISSUE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

When used with the IN or NOT IN operators, **issueKey** supports:

- issueHistory()
- linkedIssues()
- votedIssues()
- watchedIssues()

Examples

- Find the issue with key "ABC-123":

```
issueKey = ABC-123
```

^top of fields | ^^top of topic

Level

Only available if Issue Level Security has been enabled by your JIRA administrator.

Search for issues with a particular Security Level. You can search by Issue Security Level name or Issue Security Level ID (i.e. the number that JIRA automatically allocates to an Issue Security Level).

> **It is safer to search by Security Level ID than by Security Level name**
>
> It is possible for your JIRA administrator to change the name of a Security Level, which could break any saved filter which rely on that name. Security Level IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
level
```

Field Type

SECURITY LEVEL

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Search for issues with a Security Level of "Really High" or "level1":

```
level in ("Really High", level1)
```

- Search for issues with a Security Level ID of 123:

```
level = 123
```

^top of fields | ^^top of topic

Original Estimate

Only available if time-tracking has been enabled by your JIRA administrator.

Search for issues where the Original Estimate is set to a particular value (i.e. a number, not a date or date range).

Use "w", "d", "h" and "m" to specify weeks, days, hours or minutes.

Note: this field does not support auto-complete.

Syntax

```
originalEstimate
```

Alias:

```
timeOriginalEstimate
```

Field Type

DURATION

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues with an Original Estimate of 1 hour:

```
originalEstimate = 1h
```

- Find issues with an Original Estimate of more than 2 days:

```
originalEstimate > 2d
```

^top of fields | ^^top of topic

Parent

Only available if sub-tasks have been enabled by your JIRA administrator.

Search for all sub-tasks of a particular issue. You can search by Issue Key or by Issue ID (i.e. the number that JIRA automatically allocates to an Issue).

Note: this field does not support auto-complete.

Syntax

```
parent
```

Field Type

ISSUE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues that are sub-tasks of issue TEST-1234:

```
parent = TEST-1234
```

^top of fields | ^^top of topic

Priority

Search for issues with a particular Priority. You can search by Priority name or Priority ID (i.e. the number that JIRA automatically allocates to a Priority).

> **It is safer to search by Priorty ID than by Priority name**
>
> It is possible for your JIRA administrator to change the name of a Priority, which could break any saved filter which rely on that name. Priority IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
priority
```

Field Type

PRIORITY

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅

Supported Functions

n/a

Examples

- Find issues with a Priority of "High":

```
priority = High
```

- Find issues with a Priority ID of 10000:

```
priority = 10000
```

^top of fields | ^^top of topic

Project

Search for issues that belong to a particular Project.

You can search by Project Name, by Project Key or by Project ID (i.e. the number that JIRA automatically allocates to a project).

Note: this field supports auto-complete.

Syntax

```
project
```

Field Type

PROJECT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

When used with the IN and NOT IN operators, **`project`** supports:

- projectsLeadByUser()
- projectsWhereUserHasPermission()
- projectsWhereUserHasRole()

Examples

- Find issues that belong to the Project that has the name "ABC Project":

```
project = "ABC Project"
```

- Find issues that belong to the Project that has the key "ABC":

```
project = "ABC"
```

- Find issues that belong to the Project that has the ID "1234":

```
project = 1234
```

^top of fields | ^^top of topic

Remaining Estimate

Only available if time-tracking has been enabled by your JIRA administrator.

Search for issues where the Remaining Estimate is set to a particular value (i.e. a number, not a date or date range).

Use "w", "d", "h" and "m" to specify weeks, days, hours or minutes.

Note: this field does not support auto-complete.

Syntax

```
remainingEstimate
```

Alias:

```
timeEstimate
```

Field Type

DURATION

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues with a Remaining Estimate of more than 4 hours:

```
remainingEstimate > 4h
```

^top of fields | ^^top of topic

Reporter

Search for issues that were reported by (i.e. created by) a particular user.

You can search by the user's Full Name, ID or Email Address.

Note: this field supports auto-complete.

Syntax

```
reporter
```

Field Type

USER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	✅

Supported FunctionsWhen used with the IN and NOT IN operators, this field supports:

- membersOf()

When used with the EQUALS and NOT EQUALS operators, this field supports:

- currentUser()

Examples

- Search for issues that were created by Jill Jones:

```
reporter = "Jill Jones"
```

or

```
reporter = jjones
```

- Search for issues that were created by the user with email address "bob@mycompany.com":

```
assignee = "bob@mycompany.com"
```

(Note that full-stops and "@" symbols are reserved characters, so the email address needs to be surrounded by quote-marks.)

^top of fields | ^^top of topic

Resolution

Search for issues that have a particular Resolution

You can search by Resolution name or Resolution ID (i.e. the number that JIRA automatically allocates to a Resolution).

> **It is safer to search by Resolution ID than Resolution name**
>
> It is possible for your JIRA administrator to change the name of a Resolution, which could break any saved filter which rely on that name. Resolution IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
resolution
```

Field Type

RESOLUTION

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅

Supported Functions

n/a

Examples

- Find issues with a Resolution of "Cannot Reproduce" or "Won't Fix":

```
resolution in ("Cannot Reproduce", "Won't Fix")
```

- Find issues with a Resolution ID of 5:

```
resolution = 5
```

- Find issues that do not have a Resolution:

```
resolution = unresolved
```

^top of fields | ^^top of topic

Resolved

Search for issues that were resolved on, before or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the JIRA server's time zone).

Use one of the following formats:

```
"yyyy/MM/dd HH:mm"
"yyyy-MM-dd HH:mm"
"yyyy/MM/dd"
"yyyy-MM-dd"
```

Or use `"w"` (weeks), `"d"` (days), `"h"` (hours) or `"m"` (minutes) to specify a date relative to the current time. The default is `"m"` (minutes). Be sure to use quote-marks (`"`); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Note: this field does not support auto-complete.

Syntax

```
resolved
```

Alias:

```
resolutionDate
```

Field Type

DATE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:

- currentLogin()
- lastLogin()
- now()
- startOfDay()
- startOfWeek()
- startOfMonth()
- startOfYear()

- endOfDay()
- endOfWeek()
- endOfMonth()
- endOfYear()

Examples

- Find all issues that were resolved before 31st December 2010:

```
resolved <= "2010/12/31"
```

- Find all issues that were resolved before 2.00pm on 31st December 2010:

```
resolved < "2010/12/31 14:00"
```

- Find all issues that were resolved on or before 31st December 2010:

```
resolved <= "2011/01/01"
```

- Find issues that were resolved in January 2011:

```
resolved > "2011/01/01" and resolved < "2011/02/01"
```

- Find issues that were resolved on 15 January 2011:

```
resolved > "2011/01/15" and resolved < "2011/01/16"
```

- Find issues that were resolved in the last hour:

```
resolved > -1h
```

^top of fields | ^^top of topic

Status

Search for issues that have a particular Status.

You can search by Status name or Status ID (i.e. the number that JIRA automatically allocates to a Status).

> **It is safer to search by Status ID than Status name**
>
> It is possible for your JIRA administrator to change the name of a Status, which could break any saved filter which rely on that name. Status IDs, however, are unique and cannot be changed.
>
> Please note, though, that the `WAS`, `WAS_NOT`, `WAS_IN` and `WAS_NOT_IN` operators can only be used with the name (not the ID).

Note: this field supports auto-complete.

Syntax

```
status
```

Field Type

STATUS

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	✅

Supported Functions

n/a

Examples

- Find issues with a Status of "Open":

```
status = Open
```

- Find issues with a Status ID of 1:

```
status = 1
```

- Find issues that currently have, or previously had, a Status of "Open":

```
status WAS Open
```

^top of fields | ^^top of topic

Summary

Search for issues where the Summary contains particular text.

JIRA text-search syntax can be used.

Note: this field does not support auto-complete.

Syntax

```
summary
```

Field Type

TEXT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	✅	✅	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues where the Summary contains text that matches "Error saving file" (i.e. a "fuzzy" match):

```
summary ~ "Error saving file"
```

- Find issues where the Summary contains the exact phrase "Error saving file":

```
summary ~ "\"Error saving file\""
```

^top of fields | ^^top of topic

Text

This is a "master-field" that allows you to search all text fields, i.e.:

- Summary
- Description
- Environment
- Comments
- custom fields which use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
  - Free Text Field (unlimited text)
  - Text Field (< 255 characters)
  - Read-only Text Field

Note: The **`text`** master-field can only be used with the CONTAINS operator ("~" and "!~").

Syntax

```
text
```

Field Type

TEXT

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

Supported Functions

n/a

Examples

- Find issues where a text field matches the word "Fred":

```
text ~ "Fred"
```

  or

```
text ~ Fred
```

- Find all issues where a text field contains the exact phrase "full screen":

```
text ~ "\"full screen\""
```

^top of fields | ^^top of topic

Type

Search for issues that have a particular Issue Type.

You can search by Issue Type name or Issue Type ID (i.e. the number that JIRA automatically allocates to an Issue Type).

> **It is safer to search by Type ID than Type name**
>
> It is possible for your JIRA administrator to change the name of a Type, which could break any saved filter which rely on that name. Type IDs, however, are unique and cannot be changed.

Note: this field supports auto-complete.

Syntax

```
type
```

Alias:

```
issueType
```

Field Type

ISSUE_TYPE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues with an Issue Type of "Bug":

```
type = Bug
```

- Find issues with an Issue Type of "Bug" or "Improvement":

```
issueType in (Bug,Improvement)
```

- Find issues with an Issue Type ID of 2:

```
issueType = 2
```

^top of fields | ^^top of topic

Time Spent

Only available if time-tracking has been enabled by your JIRA administrator.

Search for issues where the Time Spent is set to a particular value (i.e. a number, not a date or date range).

Use "w", "d", "h" and "m" to specify weeks, days, hours or minutes.

Note: this field does not support auto-complete.

Syntax

```
timeSpent
```

Field Type

DURATION

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues where the Time Spent is more than 5 days:

```
timeSpent > 5d
```

^top of fields | ^^top of topic

Updated

Search for issues that were last updated on, before or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the JIRA server's time zone).

Use one of the following formats:

```
"yyyy/MM/dd HH:mm"
"yyyy-MM-dd HH:mm"
"yyyy/MM/dd"
"yyyy-MM-dd"
```

Or use `"w"` (weeks), `"d"` (days), `"h"` (hours) or `"m"` (minutes) to specify a date relative to the current time. The default is `"m"` (minutes). Be sure to use quote-marks (`"`); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Note: this field does not support auto-complete.

Syntax

```
updated
```

Alias:

```
updatedDate
```

Field Type

DATE

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported FunctionsWhen used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:

- currentLogin()
- lastLogin()
- now()
- startOfDay()
- startOfWeek()
- startOfMonth()
- startOfYear()
- endOfDay()
- endOfWeek()
- endOfMonth()
- endOfYear()

Examples

- Find issues that were last updated before 12th December 2010:

```
updated < "2010/12/12"
```

- Find issues that were last updated on or before 12th December 2010:

```
updated < "2010/12/13"
```

- Find all issues that were last updated before 2.00pm on 31st December 2010:

```
updated < "2010/12/31 14:00"
```

- Find issues that were last updated more than two weeks ago:

```
updated < "-2w"
```

- Find issues that were last updated on 15 January 2011:

```
updated > "2011/01/15" and updated < "2011/01/16"
```

- Find issues that were last updated in January 2011:

```
updated > "20011/01/01" and updated < "2011/02/01"
```

^top of fields | ^^top of topic

Voter

Search for issues for which a particular user has voted. You can search by the user's Full Name, ID or Email Address. Note that you can only find issues for which you have the "View Voters and Watchers" permission, unless you are searching for your own votes. See also votedIssues.

Note: this field supports auto-complete.

Syntax

```
voter
```

Field Type

USER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported FunctionsWhen used with the IN and NOT IN operators, this field supports:

- membersOf()

When used with the EQUALS and NOT EQUALS operators, this field supports:

- currentUser()

Examples

- Search for issues for which you have voted:

```
voter = currentUser()
```

- Search for issues for which the user "jsmith" has voted:

```
voter = "jsmith"
```

- Search for issues for which a member of the group "jira-developers" has voted:

```
voter in membersOf("jira-developers")
```

^top of fields | ^^top of topic

Votes

Search for issues with a specified number of votes.

Note: this field does not support auto-complete.

Syntax

```
votes
```

Field Type

NUMBER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find all issues that have 12 or more votes:

```
votes >= 12
```

^top of fields | ^^top of topic

Watcher

Search for issues that a particular user is watching. You can search by the user's Full Name, ID or Email Address. Note that you can only find issues for which you have the "View Voters and Watchers" permission, unless you are searching for issues where you are the watcher. See also watchedIssues.

Note: this field supports auto-complete.

Syntax

```
watcher
```

Field Type

USER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	❌	❌	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported FunctionsWhen used with the IN and NOT IN operators, this field supports:

- membersOf()

When used with the EQUALS and NOT EQUALS operators, this field supports:

- currentUser()

Examples

- Search for issues that you are watching:

```
watcher = currentUser()
```

- Search for issues that the user "jsmith" is watching:

```
watcher = "jsmith"
```

- Search for issues that are being watched by a member of the group "jira-developers":

```
watcher in membersOf("jira-developers")
```

^top of fields | ^^top of topic

Watchers

Search for issues with a specified number of watchers.

Note: this field does not support auto-complete.

Syntax

```
watchers
```

Field Type

NUMBER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find all issues that are being watched by more than 3 people:

```
watchers > 3
```

^top of fields | ^^top of topic

Work Ratio

Only available if time-tracking has been enabled by your JIRA administrator.

Search for issues where the Work Ratio has a particular value.

Work Ratio is calculated as follows: **workRatio = timeSpent / originalEstimate) x 100**

Note: this field does not support auto-complete.

Syntax

```
workRatio
```

Field Type

NUMBER

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
✅	✅	❌	❌	✅	✅	✅	✅	✅	✅	✅	✅	❌	❌	❌	❌	❌

Supported Functions

n/a

Examples

- Find issues on which more than 75% of the Original Estimate has been spent:

```
workRatio > 75
```

^top of fields | ^^top of topic

Functions Reference

A function in JQL appears as a word followed by parentheses which may contain one or more explicit values or JIRA fields. In a clause, a function is preceded by an operator, which in turn is preceded by a field. A function performs a calculation on either specific JIRA data or the function's content in parentheses, such that only true results are retrieved by the function and then again by the clause in which the function is used.

^top of topic

List of Functions:

- cascadeOption()
- componentsLeadByUser()
- currentLogin()
- currentUser()
- earliestUnreleasedVersion()
- endOfDay()
- endOfMonth()
- endOfWeek()
- endOfYear()
- issueHistory()
- lastLogin()
- latestReleasedVersion()
- linkedIssues()
- membersOf()
- now()
- projectsLeadByUser()
- projectsWhereUserHasPermission()
- projectsWhereUserHasRole()
- releasedVersions()
- standardIssueTypes()
- startOfDay()
- startOfMonth()
- startOfWeek()
- startOfYear()
- subtaskIssueTypes()
- unreleasedVersions()
- votedIssues()
- watchedIssues()

cascadeOption()

Search for issues that match the selected values of a 'cascading select' custom field.

The *parentOption* parameter matches against the first tier of options in the cascading select field. The *childOption* parameter matches against the second tier of options in the cascading select field, and is optional.

The keyword `"none"` can be used to search for issues where either or both of the options have no value.

Syntax

```
cascadeOption(parentOption)
```

or

```
cascadeOption(parentOption,childOption)
```

Supported Fields

- custom fields of type 'Cascading Select'

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues where a custom field ("Location") has the value "USA" for the first tier and "New York" for the second tier:

```
location in cascadeOption("USA","New York")
```

- Find issues where a custom field ("Location") has the value "USA" for the first tier and any value (or no value) for the second tier:

```
location in cascadeOption("USA")
```

- Find issues where a custom field ("Location") has the value "USA" for the first tier and no value for the second tier:

```
location in cascadeOption("USA",none)
```

- Find issues where a custom field ("Location") has no value for the first tier and no value for the second tier:

```
location in cascadeOption(none)
```

- Find issues where a custom field ("Referrer") has the value "none" for the first tier and "none" for the second tier:

```
referrer in cascadeOption("\"none\"","\"none\"")
```

- Find issues where a custom field ("Referrer") has the value "none" for the first tier and no value for the second tier:

```
referrer in cascadeOption("\"none\"",none)
```

^top of functions | ^^top of topic

componentsLeadByUser()

Find issues in components that are lead by a specific user.

You can optionally specify a user, or if the user is omitted the current user (i.e. you) will be used.

Note that if you are not logged in to JIRA, a user must be specified.

Syntax

```
componentsLeadByUser()
```

or

```
componentsLeadByUser(username)
```

Supported Fields

- Component

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find open issues in components that are lead by you:

```
component in componentsLeadByUser() AND status = Open
```

- Find open issues in components that are lead by Bill:

```
component in componentsLeadByUser(bill) AND status = Open
```

^top of functions | ^^top of topic

currentLogin()

Perform searches based on the time at which the current user's session began. See also lastLogin.

Syntax

```
currentLogin()
```

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues that have been created during my current session:

```
created > currentLogin()
```

^top of functions | ^^top of topic

currentUser()

Perform searches based on the currently logged-in user.

Note that this function can only be used by logged-in users. So if you are creating a saved filter that you expect to be used by anonymous users, do not use this function.

Syntax

```
currentUser()
```

Supported Fields

- Assignee
- Reporter
- Voter
- Watcher
- custom fields of type User

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED

✅	✅	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	❌

Examples

- Find issues that are assigned to me:

```
assignee = currentUser()
```

- Find issues that were reported to me but are not assigned to me:

```
reporter = currentUser() and assignee != currentUser()
```

^top of functions | ^^top of topic

earliestUnreleasedVersion()

Perform searches based on the earliest unreleased version (i.e. next version that is due to be released) of a specified project. See also unreleasedVersions.

Note that the "earliest" is determined by the ordering assigned to the versions, not by actual Version Due Dates.

Syntax

```
earliestUnreleasedVersion(project)
```

Supported Fields

- AffectedVersion
- FixVersion
- custom fields of type Version

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues whose FixVersion is the earliest unreleased version of the ABC project:

```
fixVersion = earliestUnreleasedVersion(ABC)
```

- Find issues that relate to the earlist unreleased version of the ABC project:

```
affectedVersion = earliestUnreleasedVersion(ABC) or fixVersion =
earliestUnreleasedVersion(ABC)
```

^top of functions | ^^top of topic

endOfDay()

Perform searches based on the end of the current day. See also endOfWeek, endOfMonth and endOfYear; and startOfDay, startOfWeek, startOfMonth and startOfYear.

Syntax

```
endOfDay()
```

or

```
endOfDay("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. `endOfDay("+1")` is the same as `endOfDay("+1d")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues due by the end of today:

```
due < endOfDay()
```

- Find issues due by the end of tomorrow:

```
due < endOfDay("+1")
```

^top of functions | ^^top of topic

endOfMonth()

Perform searches based on the end of the current month. See also endOfDay, endOfWeek and endOfYear; and startOfDay, startOfWeek, startOfMonth and startOfYear.

Syntax

```
endOfMonth()
```

or

```
endOfMonth("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. `endOfMonth("+1")` is the same as `endOfMonth("+1M")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues due by the end of this month:

```
due < endOfMonth()
```

- Find issues due by the end of next month:

```
due endOfMonth("+1")
```

- Find issues due by the 15th of next month:

```
due endOfMonth("+15d")
```

^top of functions | ^^top of topic

endOfWeek()

Perform searches based on the end of the current week. See also endOfDay, endOfMonth and endOfYear; and startOfDay, startOfWeek, startOfMonth and startOfYear.

For the endOfWeek() function the result depends upon your locale. For example, in Europe the first day of the week is generally considered to be Monday, while in the USA it is considered to be Sunday.

Syntax

```
endOfWeek()
```

or

```
endOfWeek("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. endOfWeek("+1") is the same as endofWeek("+1w").
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues due by the end of this week:

```
due < endOfWeek()
```

- Find issues due by the end of next week:

```
due < endOfWeek("+1")
```

^top of functions | ^^top of topic

endOfYear()

Perform searches based on the end of the current year. See also startOfDay, startOfWeek and startOfMonth; and endOfDay, endOfWeek, endOfMonth and endOfYear.

```
startOfYear()
```

or

```
startOfYear("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. `endOfYear("+1")` is the same as `endofYear("+1y")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues due by the end of this year:

```
due < endOfYear()
```

- Find issues due by the end of March next year:

```
due < endOfYear("+3M")
```

^top of functions | ^^top of topic

issueHistory()

Find issues that you have recently viewed, i.e. issues that are in the 'Recent Issues' section of the 'Issues' drop-down menu.

Note:

- `issueHistory()` returns up to 50 issues, whereas the 'Recent Issues' drop-down returns only 5.
- if you are not logged in to JIRA, only issues from your current browser session will be included.

Syntax

```
issueHistory()
```

Supported Fields

- Issue

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues which I have recently viewed, that are assigned to me:

```
issue in issueHistory() AND assignee = currentUser()
```

^top of functions | ^^top of topic

lastLogin()

Perform searches based on the time at which the current user's previous session began. See also currentLogin.

Syntax

```
currentLogin()
```

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues that have been created during my last session:

```
created > lastLogin()
```

^top of functions | ^^top of topic

latestReleasedVersion()

Perform searches based on the latest released version (i.e. the most recent version that has been released) of a specified project. See also releasedVersions().

Note that the "latest" is determined by the ordering assigned to the versions, not by actual Version Due Dates.

Syntax

```
latestReleasedVersion(project)
```

Supported Fields

- AffectedVersion
- FixVersion
- custom fields of type Version

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues whose FixVersionis the latest released version of the ABC project:

```
fixVersion = latestReleasedVersion(ABC)
```

- Find issues that relate to the latest released version of the ABC project:

```
affectedVersion = latestReleasedVersion(ABC) or fixVersion = latestReleasedVersion(ABC)
```

^top of functions | ^^top of topic

linkedIssues()

Perform searches based on issues which are linked to a specified issue.

You can optionally restrict the search to links of a particular type. Note that LinkType is case-sensitive.

Syntax

```
linkedIssues(issueKey)
```

or

```
linkedIssues(issueKey,linkType)
```

Supported Fields

- Issue

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues that are linked to a particular issue:

```
issue in linkedIssues(ABC-123)
```

- Find issues that are linked to a particular issue via a particular type of link:

```
issue in linkedIssues(ABC-123,"is duplicated by")
```

^top of functions | ^^top of topic

membersOf()

Perform searches based on the members of a particular group.

Syntax

```
membersOf(Group)
```

Supported Fields

- Assignee
- Reporter
- Voter
- Watcher
- custom fields of type User

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues where the Assignee is a member of the group "jira-developers":

```
assignee in membersOf("jira-developers")
```

- Search through multiple groups and a specific user, e.g:

```
reporter in membersOf("jira-developers") or reporter in membersOf("jira-administrators")
or reporter=jsmith
```

- Search for a particular group, but exclude a particular member or members, e.g.:

```
assignee in membersOf(QA) and assignee not in ("John Smith","Jill Jones")
```

- Exclude members of a particular group:

```
assignee not in membersOf(QA)
```

^top of functions | ^^top of topic

now()

Perform searches based on the current time.

Syntax

```
now()
```

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find issues that are overdue:

```
duedate < now() and status not in (closed, resolved)
```

^top of functions | ^^top of topic

projectsLeadByUser()

Find issues in projects that are lead by a specific user.

You can optionally specify a user, or if the user is omitted the current user will be used.

Note that if you are not logged in to JIRA, a user must be specified.

Syntax

```
projectsLeadByUser()
```

or

```
projectsLeadByUser(username)
```

Supported Fields

- Project

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find open issues in projects that are lead by you:

```
project in projectsLeadByUser() AND status = Open
```

- Find open issues in projects that are lead by Bill:

```
project in projectsLeadByUser(bill) AND status = Open
```

^top of functions | ^^top of topic

projectsWhereUserHasPermission()

Find issues in projects where you have a specific permission.

Note: This function operates at the project level. This means that if a permission (e.g. "Edit Issues") is granted to the reporter of issues in a project, then you may see some issues returned where you are
not the reporter and therefore don't have the permission specified.

Also note that this function is only available if you are logged in to JIRA.

Syntax

```
projectsWhereUserHasPermission(permission)
```

For the `permission` parameter you can specify any of the following:

Project Permissions	Explanation
Administer Projects	Permission to administer a project in JIRA. This includes the ability to edit project role membership, project components, project versions and some project details ('Project Name', 'URL', 'Project Lead', 'Project Description').
Browse Projects	Permission to browse projects, use the Issue Navigator and view individual issues (except issues that have been restricted via Issue Security). **Many other permissions are dependent on this permission**, e.g. the 'Work On Issues' permission is only effective for users who also have the 'Browse Projects' permission.
View Version Control	Permission to view the version control information (e.g. CVS, Subversion, FishEye, etc) for an issue. Note that for CVS, to view the Version Control information the project needs to be associated with at least one Repository.
View (Read-Only) Workflow	Permission to view the project's 'read-only' workflow when viewing an issue. This permission provides the 'View Workflow' link against the 'Status' field of the 'View Issue' page.
Issue Permissions	**Explanation**
Assign Issues	Permission to assign issues to users. (See also Assignable User permission below)

Assignable User	Permission to be assigned issues. (Note that this does not include the ability to assign issues; see Assign Issue permission above).
Close Issues	Permission to close issues. (This permission is useful where, for example, developers resolve issues and testers close them). Also see the Resolve Issues permission.
Create Issues	Permission to create issues in the project. (Note that the Create Attachments permission is required in order to create attachments.) Includes the ability to create sub-tasks (if sub-tasks are enabled).
Delete Issues	Permission to delete issues. Think carefully about which groups or project roles you assign this permission to; usually it will only be given to administrators. Note that deleting an issue will delete all of its comments and attachments, even if the user does not have the Delete Comments or Delete Attachments permissions. However, the Delete Issues permission does not include the ability to delete individual comments or attachments.
Edit Issues	Permission to edit issues (excluding the 'Due Date' field — see the Schedule Issues permission). Includes the ability to convert issues to sub-tasks and vice versa (if sub-tasks are enabled). Note that the Delete Issue permission is required in order to delete issues. The Edit Issue permission is usually given to any groups or project roles who have the Create Issue permission (perhaps the only exception to this is if you give **everyone** the ability to create issues — it may not be appropriate to give everyone the ability to edit too). Note that all edits are recorded in the Issue Change History for audit purposes.
Link Issues	Permission to link issues together. (Only relevant if Issue Linking is enabled).
Modify Reporter	Permission to modify the 'Reporter' of an issue. This allows a user to create issues 'on behalf of' someone else. This permission should generally only be granted to administrators.
Move Issues	Permission to move issues from one project to another, or from one workflow to another workflow within the same project. Note that a user can only move issues to a project for which they have Create Issue permission.
Resolve Issues	Permission to resolve and reopen issues. This also includes the ability to set the 'Fix For version' field for issues. Also see the Close Issues permission.
Schedule Issues	Permission to schedule an issue — that is, set and edit the 'Due Date' of an issue.
Set Issue Security	Permission to set the security level on an issue to control who can access the issue. Only relevant if issue security has been enabled.
Voters & Watchers Permissions	**Explanation**
Manage Watcher List	Permission to manage (i.e. view/add/remove users to/from) the watcher list of an issue.
View Voters and Watchers	Permission to view the voter list and watcher list of an issue. Also see the Manage Watcher List permission.
Comments Permissions	**Explanation**
Add Comments	Permission to add comments to issues. Note that this does not include the ability to edit or delete comments.
Delete All Comments	Permission to delete any comments, regardless of who added them.
Delete Own Comments	Permission to delete comments that were added by the user.
Edit All Comments	Permission to edit any comments, regardless of who added them.

Edit Own Comments	Permission to edit comments that were added by the user.
Attachments Permissions	**Explanation**
Create Attachments	Permission to attach files to an issue. (Only relevant if attachments are enabled). Note that this does not include the ability to delete attachments.
Delete All Attachments	Permission to delete any attachments, regardless of who added them.
Delete Own Attachments	Permission to delete attachments that were added by the user.
Time Tracking Permissions	**Explanation**
Work On Issues	Permission to log work against an issue, i.e. create a worklog entry. (Only relevant if Time Tracking is enabled).
Delete All Worklogs	Permission to delete any worklog entries, regardless of who added them. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.
Delete Own Worklogs	Permission to delete worklog entries that were added by the user. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.
Edit All Worklogs	Permission to edit any worklog entries, regardless of who added them. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.
Edit Own Worklogs	Permission to edit worklog entries that were added by the user. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.

Supported Fields

- Project

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find open issues in projects where you have the "Resolve Issues" permission:

```
project in projectsWhereUserHasPermission("Resolve Issues") AND status = Open
```

^top of functions | ^^top of topic

projectsWhereUserHasRole()

Find issues in projects where you have a specific role.

Note that this function is only available if you are logged in to JIRA.

Syntax

```
projectsWhereUserHasRole(rolename)
```

Supported Fields

- Project

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find open issues in projects where you have the "Developers" role:

```
project in projectsWhereUserHasRole("Developers") AND status = Open
```

^top of functions | ^^top of topic

releasedVersions()

Perform searches based on the released versions (i.e. versions that your JIRA administrator has released) of a specified project.

You can also search on the released versions of all projects, by omitting the *project* parameter.

See also latestReleasedVersion().

Syntax

```
releasedVersions()
```

or

```
releasedVersions(project)
```

Supported Fields

- AffectedVersion
- FixVersion
- custom fields of type Version

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues whose FixVersionis a released version of the ABC project:

```
fixVersion in releasedVersions(ABC)
```

- Find issues that relate to released versions of the ABC project:

```
affectedVersion in releasedVersions(ABC)
```

or

```
fixVersion in releasedVersions(ABC)
```

^top of functions | ^^top of topic

standardIssueTypes()

Perform searches based on "standard" Issue Types, that is, search for issues which are not sub-tasks.

See also subtaskIssueTypes().

Syntax

```
standardIssueTypes()
```

Supported Fields

- Type

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues that are not subtasks (i.e. issues whose Issue Type is a standard issue type, not a subtask issue type):

```
issuetype in standardIssueTypes()
```

^top of functions | ^^top of topic

startOfDay()

Perform searches based on the start of the current day. See also startOfWeek, startOfMonth and startOfYear; and endOfDay, endOfWeek, endOfMonth and endOfYear.

Syntax

```
startOfDay()
```

or

```
startOfDay("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. `startOfDay("+1")` is the same as `startofDay("+1d")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find new issues created since the start of today:

```
created > startOfDay()
```

- Find new issues created since the start of yesterday:

```
created > startOfDay("-1")
```

- Find new issues created in the last three days:

```
created > startOfDay("-3d")
```

^top of functions | ^^top of topic

startOfMonth()

Perform searches based on the start of the current month. See also startOfDay, startOfWeek and startOfYear; and endOfDay, endOfWeek, endOfMonth and endOfYear.

Syntax

```
startOfMonth()
```

or

```
startOfMonth("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. `startOfMonth("+1")` is the same as `startofMonth("+1M")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find new issues since the start of this month:

```
created > startOfMonth()
```

- Find new issues since the start of last month:

```
created > startOfMonth("-1")
```

- Find new issues since the 15th of this month:

```
created > startOfMonth("+14d")
```

^top of functions | ^^top of topic

startOfWeek()

Perform searches based on the start of the current week. See also startOfDay, startOfMonth and startOfYear; and endOfDay, endOfWeek, endOfMonth and endOfYear.

For the startOfWeek() function the result depends upon your locale. For example, in Europe the first day of the week is generally considered to be Monday, while in the USA it is considered to be Sunday.

Syntax

```
startOfWeek()
```

or

```
startOfWeek("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. startOfWeek("+1") is the same as startofWeek("+1w").
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find new issues since the start of this week:

```
created > startOfWeek()
```

- Find new issues since the start of last week:

```
created > startOfWeek("-1")
```

^top of functions | ^^top of topic

startOfYear()

Perform searches based on the start of the current year. See also startOfDay, startOfWeek and startOfMonth; and endOfDay, endOfWeek, endOfMonth and endOfYear.

```
startOfYear()
```

or

```
startOfYear("inc")
```

where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`

- If the time unit qualifier is omitted it defaults to the natural period of the function, e.g. `startOfYear("+1")` is the same as `startofYear("+1y")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHA
✅	✅	❌	❌	✅	✅	✅	✅	❌	❌	❌	❌	(only in predicate)	(only in predicate)	(only in predicate)	(only in predicate)	(only predi

Examples

- Find new issues since the start of this year:

```
created > startOfYear()
```

- Find new issues since the start of last year:

```
created > startOfYear("-1")
```

^top of functions | ^^top of topic

subtaskIssueTypes()

Perform searches based on issues which are sub-tasks.

See also standardIssueTypes().

Syntax

```
subtaskIssueTypes()
```

Supported Fields

- Type

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues that are subtasks (i.e. issues whose Issue Type is a subtask issue type):

```
issuetype in subtaskIssueTypes()
```

^top of functions | ^^top of topic

unreleasedVersions()

Perform searches based on the unreleased versions (i.e. versions that your JIRA administrator has not yet released) of a specified project.

You can also search on the unreleased versions of all projects, by omitting the *project* parameter.

See also earliestUnreleasedVersion().

Syntax

```
unreleasedVersions()
```

or

```
unreleasedVersions(project)
```

Supported Fields

- AffectedVersion
- FixVersion
- custom fields of type Version

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues whose FixVersionis an unreleased version of the ABC project:

```
fixVersion in unreleasedVersions(ABC)
```

- Find issues that relate to unreleased versions of the ABC project:

```
affectedVersion in unreleasedVersions(ABC)
```

or

```
fixVersion in unreleasedVersions(ABC)
```

^top of functions | ^^top of topic

votedIssues()

Perform searches based on issues for which you have voted. Also see the Voter field.

Note that this function can only be used by logged-in users.

Syntax

```
votedIssues()
```

Supported Fields

- Issue

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues that you have voted for:

```
issue in votedIssues()
```

^top of functions | ^^top of topic

watchedIssues()

Perform searches based on issues which you are watching. Also see the Watcher field.

Note that this function can only be used by logged-in users.

Syntax

```
watchedIssues()
```

Supported Fields

- Issue

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues that you are watching:

```
issue in watchedIssues()
```

Supported Fields

- Issue

Supported Operators

=	!=	~	!~	>	>=	<	<=	IS	IS NOT	IN	NOT IN	WAS	WAS IN	WAS NOT	WAS NOT IN	CHANGED
❌	❌	❌	❌	❌	❌	❌	❌	❌	❌	✅	✅	❌	❌	❌	❌	❌

Examples

- Find issues that you have recently viewed:

```
issue in issueHistory()
```

^top of functions | ^^top of topic

Setting Precedence of Operators

You can use parentheses in complex JQL statements to enforce the precedence of operators.

For example, if you want to find all resolved issues in the SysAdmin project as well as all issues (any status, any project) currently assigned to the system administrator (bobsmith), you can use parentheses to enforce the precedence of the boolean operators in your query, i.e.:

```
(status=resolved AND project=SysAdmin) OR assignee=bobsmith
```

Note that if you do not use parentheses, the statement will be evaluated left-to-right.

You can also use parentheses to group clauses, so that you can apply the NOT operator to the group.

Performing Text Searches

You can use Lucene's text-searching features when performing searches on the following fields, using the CONTAINS operator:

- Summary
- Description
- Environment
- Comments
- custom fields which use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
  - Free Text Field (unlimited text)
  - Text Field (< 255 characters)
  - Read-only Text Field

For details, please see the page on Performing Text Searches.

Using Auto-complete

As you type your query, JIRA will recognise the context and offer a list of "auto-complete" suggestions as follows:

- If you type a space at the start of your query...
- If you type one or more characters...
- If you type a field then a space...
- If you type a field, then an operator, then a space...
- If you type a field, then an operator, then one or more characters...

The list of auto-complete suggestions is displayed alphabetically and includes the first 15 matches. Note that auto-complete suggestions are not offered for function parameters.

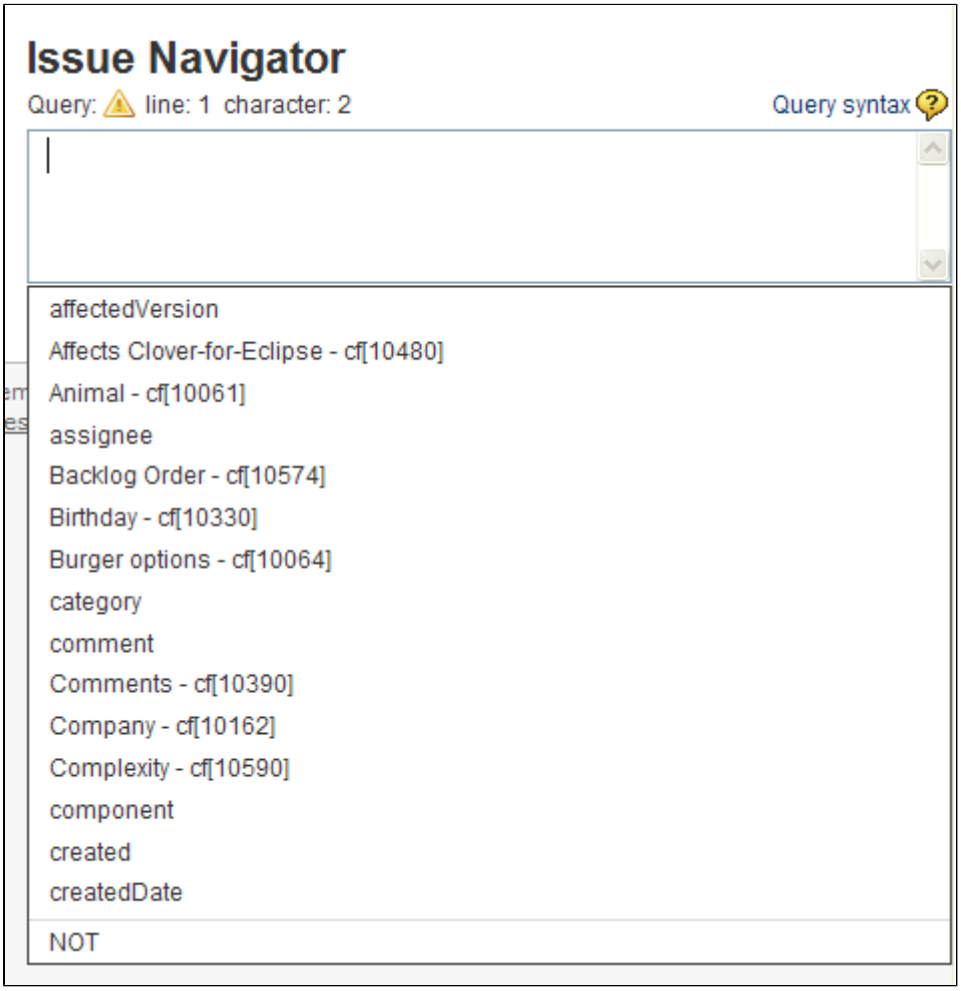
Please note:

- If no auto-complete suggestions are offered, your administrator may have disabled the "JQL Auto-complete" feature for your JIRA instance.
- If you prefer not to be offered auto-complete suggestions, click the "**Turn off auto-complete**" link below the "Query" box.

Auto-complete suggestions are not offered for all fields. Check the fields reference to see which fields support auto-complete.

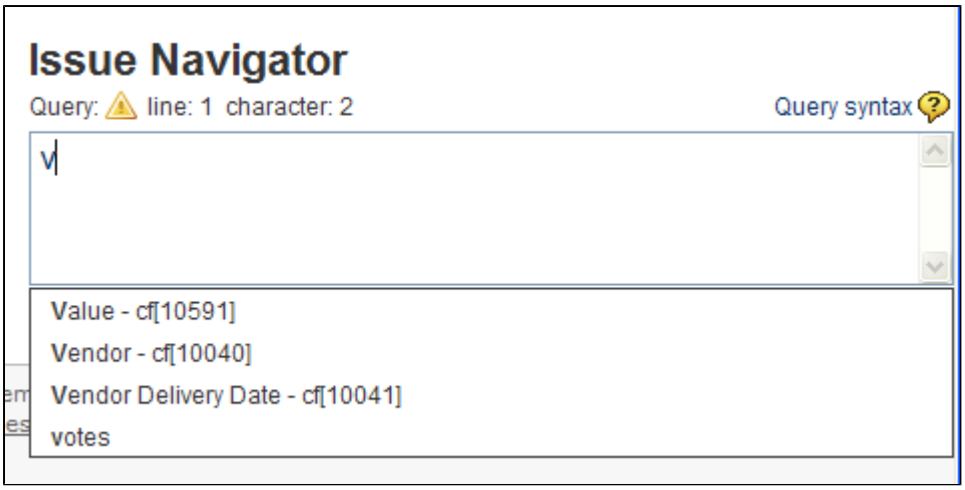
If you type a space at the start of your query...

...JIRA will offer a list of all available fields, e.g.:



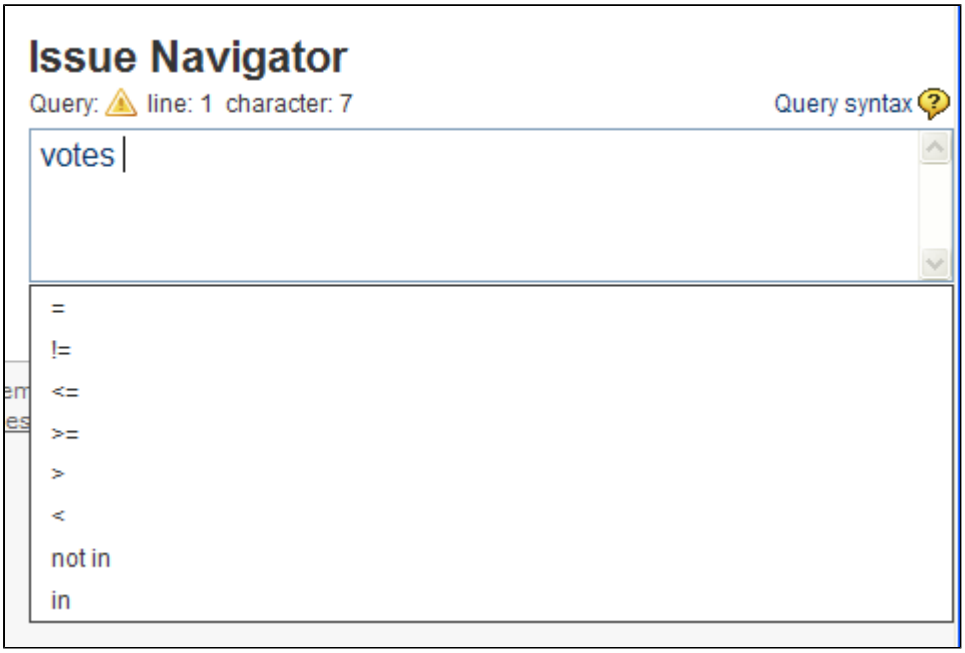
If you type one or more characters...

...JIRA will offer a list of matching fields, e.g.:



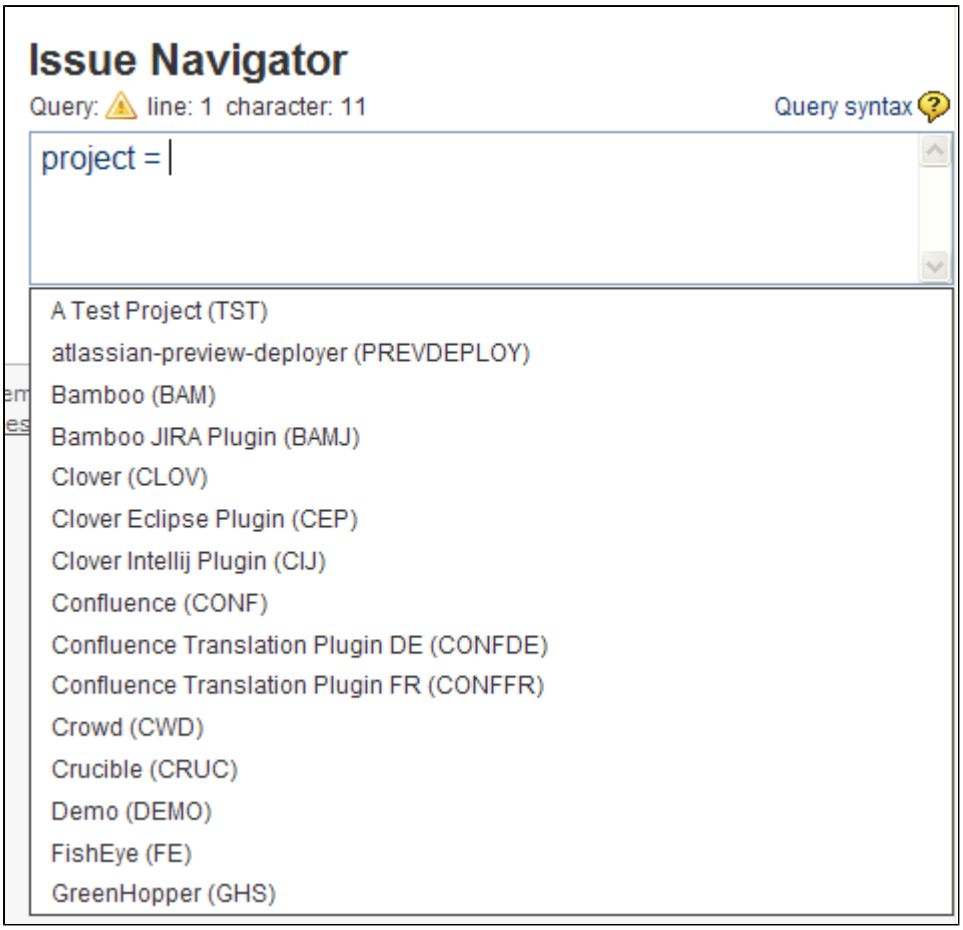
If you type a field then a space...

...JIRA will offer a list of valid operators, e.g.:



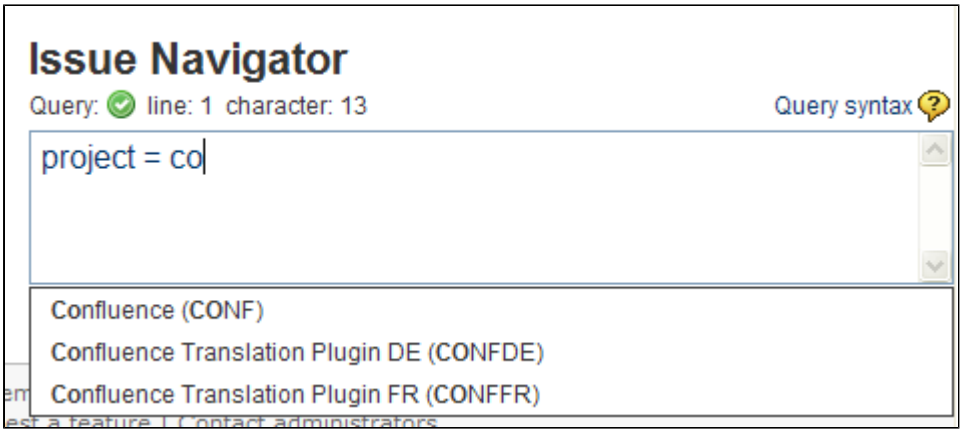
If you type a field, then an operator, then a space...

...JIRA will offer a list of valid values, e.g.:



If you type a field, then an operator, then one or more characters...

...JIRA will offer a list of valid values (if your field supports this) and valid functions for the field/operator combination, e.g.:



Switching between 'Advanced' and 'Simple' Search

In general, a query created using 'Simple Search' will be able to be translated to 'Advanced Search' (i.e. JQL), and back again.

However, a query created using 'Advanced Search' may not be able to be translated to 'Simple Search', particular if:

- the query contains an OR operator (note you can have an IN operator and it will be translated, e.g. `project in (A, B)`)
  - i.e. even though this query: `(project = JRA OR project = CONF)` is equivalent to this query: `(project in (JRA, CONF))`, only the second query will be translated.
- the query contains a NOT operator
- the query contains an EMPTY operator
- the query contains any of the comparison operators: !=, IS, IS NOT, >, >=, <, <=
- the query specifies a field and value that is related to a project (e.g. version, component, custom fields) and the project is not explicitly included in the query (e.g. `fixVersion = "4.0"`, without the `AND project=JRA`). This is especially tricky with custom fields since they can be configured on a Project/Issue Type basis. The general rule of thumb is that if the query cannot be created in the 'Simple Search' form, then if it is created using 'Advanced Search' it will not be able to be translated to 'Simple Search'.

Reserved Characters

JQL has a list of reserved characters:

- **space** (" ")
- "+"
- "."
- ","
- ":"
- "?"
- "|"
- "*"
- "/"
- "%"
- "^"
- "$"
- "#"
- "@"
- "["
- "]"

If you wish to use these characters in queries, you need to:

- surround them with quote-marks (you can use either single quote-marks (') or double quote-marks ("));
  and, if you are searching a text field and the character is on the list of reserved characters for Text Searches,
- precede them with two backslashes.

The text fields are:

- Summary
- Description
- Environment
- Comments
- custom fields which use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
  - Free Text Field (unlimited text)
  - Text Field (< 255 characters)
  - Read-only Text Field

.

For example:

```
version = "[example]"
```

```
version = "4.2"
```

```
summary ~ "\\[example\\]"
```

```
summary ~ "4.2"
```

Reserved Words

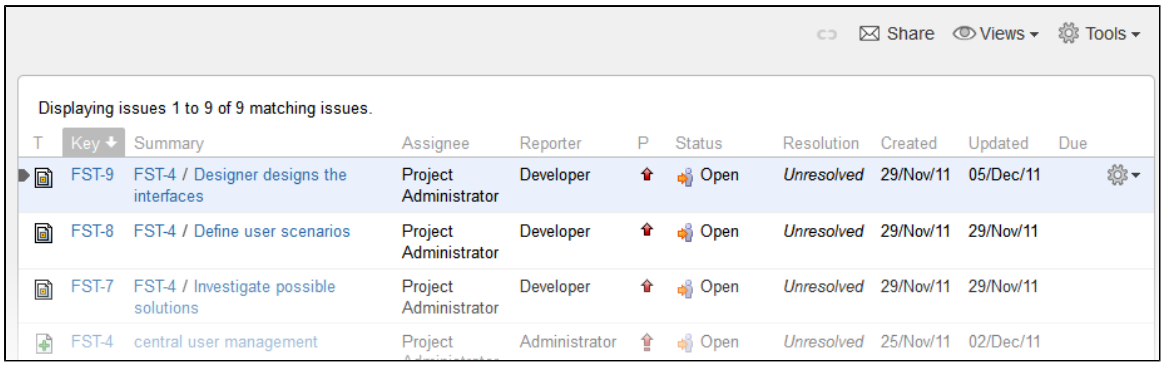
JQL has a list of reserved words. These words need to be surrounded by quote-marks if you wish to use them in queries:

"abort", "access", "add", "after", "alias", "all", "alter", "and", "any", "as", "asc",
"audit", "avg", "before", "begin", "between", "boolean", "break", "by", "byte", "catch", "cf",
"char", "character

^^top of topic

Using the Issue Navigator

The Issue Navigator displays the search results from an issue filter, a Quick Search or an Advanced Search, e.g:



On this page:

- Re-ordering the search results
- Viewing individual issues
- Accessing protected data

The Issue Navigator provides convenient ways to perform many of JIRA's most useful functions. You can:

- Click the **permalink** icon to create a permanent URL link to your search results.
- Click the **Share** button to send a link to your search results to other JIRA users as well as *any* email addresses. For details, see Sharing a Search Result.
- Use the '**j**' and '**k**' keys on your keyboard to move between issues on the Issue Navigator. The currently selected issue is depicted by an arrow pointer on the left of the list and any 'Issue Navigator' keyboard shortcuts will be performed on this issue.
- Use the actions icon for a particular issue to perform the following actions:
  - **Start Progress** — Set the issue's Status to **In Progress**.
  - **Resolve issue** — Set the issue's Status to **Resolved** and select the appropriate Resolution.
  - **Close issue** — Set the issue's Status to **Closed** and if the issue has not already been **Resolved**, select the appropriate Resolution.
  - **Reopen issue** — Set a **Resolved** or **Closed** issue's Status to **Reopened**.
  - **Edit** — Edit the issue's details (**Summary, Description,** etc).
  - **Assign** — Select an asignee for the issue.
  - **Assign To Me** — Assign the issue to yourself.
  - **Comment** — Add a comment to the issue.
  - **Log Work** — Record the work done and time spent on the issue. This option is only available if Time Tracking has been activated on your JIRA site.
  - **Attach Files** — Select a file, upload it and attach it to the issue.
  - **Attach Screenshot** — Select a file, upload it and attach it to the issue.
  - **Voters** — Opens the Voters list of the issue, where you can manage your vote and see others who have voted on the issue too.
  - **Add Vote** — Adds your vote to the issue. (This option is only available if you did not create the issue.)
  - **Watch Issue** — Become a watcher of the issue.
  - **Stop Watching** — Stop watching the issue. (This option is only available on issues you are currently watching.)
  - **Watchers** — Opens the Watchers List, where you can manage watchers of the issue.
  - **Create Sub-Task** — Create a new issue which is a sub-task of the issue.
  - **Convert to Issue** — If the issue is a sub-task, convert it to a standalone issue.
  - **Convert to Sub-Task** — If the issue is a standalone issue, convert it to a sub-task.
  - **Move** — Move the issue to a different project.
  - **Link** — Create a link between the issue and another issue. This option is only available if Issue Linking has been enabled on your JIRA site.
  - **Clone** — Create a new issue which is an identical copy of the issue.
  - **Labels** — Edit the issue's labels.
  - **Delete** — Permanently remove the issue.

  Note that some options in the **Actions** menu will only be available if you have the necessary permissions, or if certain features have been enabled by your JIRA administrator. Options which change an issue's Status (**Resolve issue, Close issue, Reopen issue**) may differ from this list, depending on your organisation's workflow.

- Use the **Views** menu to view/export your search results in various formats:
  - **Printable** — All search results on one page, with one row of data per issue. Includes the issue fields that are currently configured in your Issue Navigator.
  - **Full Content** — All search results on one page. Includes **Description, Comments** and all other issue data, not just the issue fields that are currently configured in your Issue Navigator.
  - **XML** — An XML view of issue data, suitable for use with the Confluence JIRA Issues Macro. (Also suitable for use as an RSS 0.9.2 feed). For details, see 'Displaying Search Results in XML'.
  - **RSS (Issues)** — An RSS 2.0 feed of issue data, suitable for displaying in an RSS reader. For details, see 'Receiving Search

Results via RSS'.
- **RSS (Comments)** — An RSS 2.0 feed of comments on the issues, suitable for displaying in an RSS reader. For details, see 'Receiving Search Results via RSS'.
- **Word** — An MS Word document. Includes **Description, Comments** and all other issue data, not just the issue fields that are currently configured in your Issue Navigator. Can be saved and/or edited. For details, see 'Exporting Search Results to Microsoft Word'.
- **Excel (All fields)** — An MS Excel spreadsheet. Includes **Description** and all other issue data (excluding comments). Can be saved, or edited to produce graphs and charts; also useful for basic reporting and statistics. For details, see 'Exporting Search Results to Microsoft Excel'.
- **Excel (Current fields)** — An MS Excel spreadsheet containing the issue fields that are currently configured in your Issue Navigator. Can be saved, or edited to produce graphs and charts; also useful for basic reporting and statistics. For details, see 'Exporting Search Results to Microsoft Excel'.
- **Charts** — A graphical summary of the search results, which can be saved as a dashboard gadget. For details, see ' Displaying Search Results as a Chart'.
- **On Dashboard** — A gadget on your dashboard. You will be prompted to:
  - select which gadget to use: Filter Results; Average Age Chart; Created vs Resolved Chart; Heat Map; Pie Chart; Recently Created Chart; Resolution Time; or Time Since Chart.
  - name the filter which the search results will be saved as.
  - choose which dashboard the gadget will be saved to, if you have created your own dashboards (if you have not, the gadget will be saved to a new dashboard which will be a copy of the default dashboard).

- Use the **Tools** menu to:
  - **Bulk Change** — You can move, edit or delete multiple issues at once as described in 'Modifying Multiple ('Bulk') Issues'. (Note that this option will only be available if you have been given the necessary permission.)
  - **Configure Columns** — You can add, remove and re-order columns as described in 'Customising your Issue Navigator'.

The Issue Navigator only displays one set of search results at any one time, even if you have multiple browser windows open. However, you can easily save your searches (see 'issue filters'), then display them as needed.

Re-ordering the search results

When viewing search results in the Issue Navigator, you can re-order the issues by clicking on the column header*. For example, if you click the **'Reporter'** column header, the Issue Navigator will re-display the issues in ascending order of reporter's name. If you click the **'Reporter'** column header a second time, the Issue Navigator will re-display the issues in *descending* order of reporter's name.

* With some exceptions, e.g. the 'Images' column and the sub-task aggregate columns (i.e. all columns beginning with '') are non-orderable.

To choose different fields to display in your Issue Navigator, see Customising your Issue Navigator.

If you are performing an advanced search, re-ordering issues in the Issue Navigator adds an 'ORDER BY' keyword and associated content to your JQL query (or modifies any existing 'ORDER BY' content), to reflect the order of issues in your search results. This comes in useful if you need to save searches for other purposes.

Viewing individual issues

When an issue from a search result set is selected and displayed, a mini-navigator is shown at the right of the issue's title bar:

This mini-navigator indicates the current issue's position within the result set. It also provides linked arrow icons to the previous and next issues in the result set (as shown in the image above), along with a 'Return to search' link that leads you back to the search results. You can also navigate through the search results by using the shortcut keys: 'p' (previous) and 'n' (next).

Accessing protected data

When accessing data generated from JIRA, you may find that access to some resources requires user authentication (i.e. requires you to login). There are three options for this:

1. To enable access to data without logging in, your JIRA administrator may specify the 'Browse' permission for **Anyone.**
2. You can provide the parameters **os_username** and **os_password** in the request URL (e.g. `http://jira.atlassian.com/browse/TST-1?os_username=tester&os_password=tstpassword`). The problem with this method is that it transmits your username and password across the wire in clear text, which may not be an option for some users.
3. You can provide the request parameter **os_authType=basic** (e.g. `http://mycompany.com/anypage?os_authType=basic`). This will force the server to issue a challenge for user credentials (i.e. a login prompt) via the basic http authentication protocol. If you are running over SSL, you still need to specify the **os_authType=basic** parameter if you require the user to authenticate.

Customising your Issue Navigator

The Issue Navigator displays the search results from an issue filter, a Quick Search or an Advanced Search.

You can customise your Issue Navigator by choosing:

- which **columns** (i.e. issue fields) to display
- how many **rows** (i.e. issues) to display

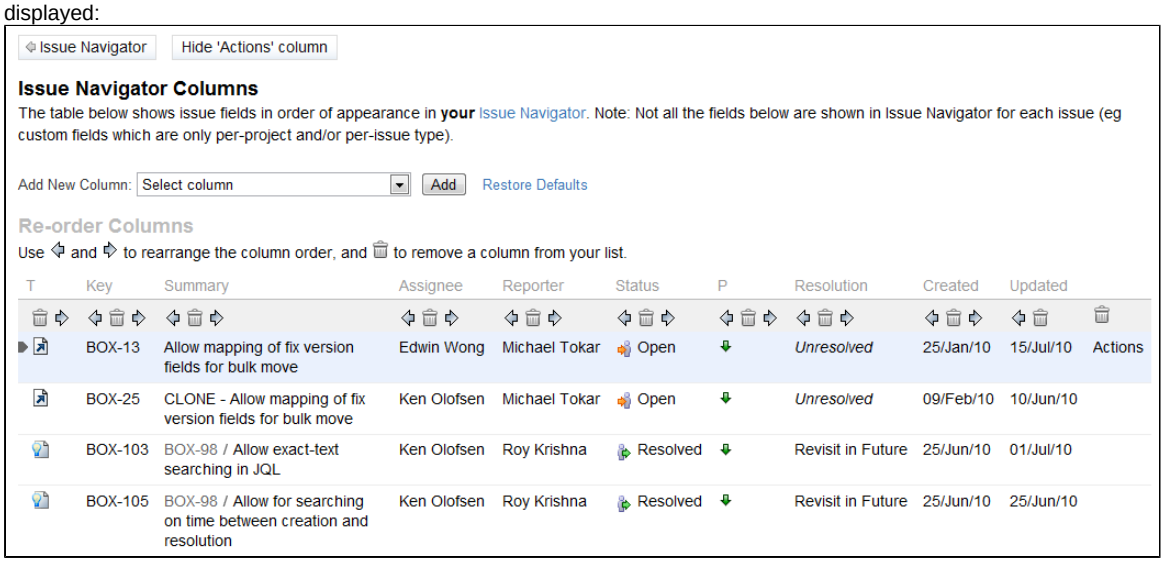
On this page:

- Customising your Issue Navigator columns
- Customising your Issue Navigator rows
- Related Topics

Customising your Issue Navigator columns

To choose which columns (i.e. issue fields) to display in your Issue Navigator,

1. Click the **'Issues'** link in the navigation bar at the top of the screen. The Issue Navigator will be displayed.
2. From the **'Tools'** menu at the right of the screen (above the search results), select **'Configure Columns'.** The following will be displayed:



   - **To move a column left or right,** click on the left-arrow or right-arrow icon that appears under the column's heading.
   - **To remove a column from the list,** click the bin icon which appears under the column's heading.
   - **To add a column to the list,** select the issue field name from the drop-down box titled **'Add New Column'** and click the **'Add'** button. The column will appear as the right-most column in the list. You can then position the column where desired by using the arrow icons.
   - **To hide the 'Actions' column,** click the **'Hide Actions Column'** button at the top of the screen.
   - **To restore the default configuration,** click the '**Restore Defaults**' link.

Customising your Issue Navigator rows

To choose how many rows (i.e. issues) to display on each page of your Issue Navigator:

1. Click your user name at the top-right of the page to open your User Profile.
2. Click **'Edit Preferences',** under **'Operations'** in the left-hand column of the screen.
3. Enter your preferred **'Number of Issues displayed per Issue Navigator page'**, (The default is 50).
4. Click the **'Update'** button.

Related Topics

- Using the Issue Navigator

Saving Searches ('Issue Filters')

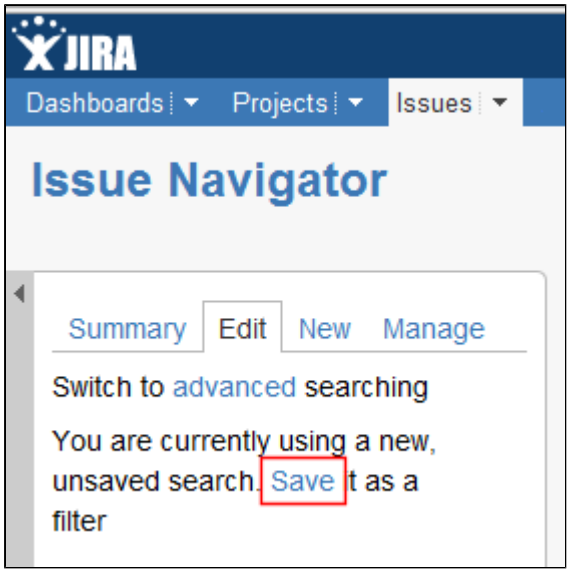
JIRA's powerful issue search functionality is further enhanced by the ability to *save* a search for use at a later time. A saved search is called an 'issue filter' or simply **'filter'.**

With a filter, you can:

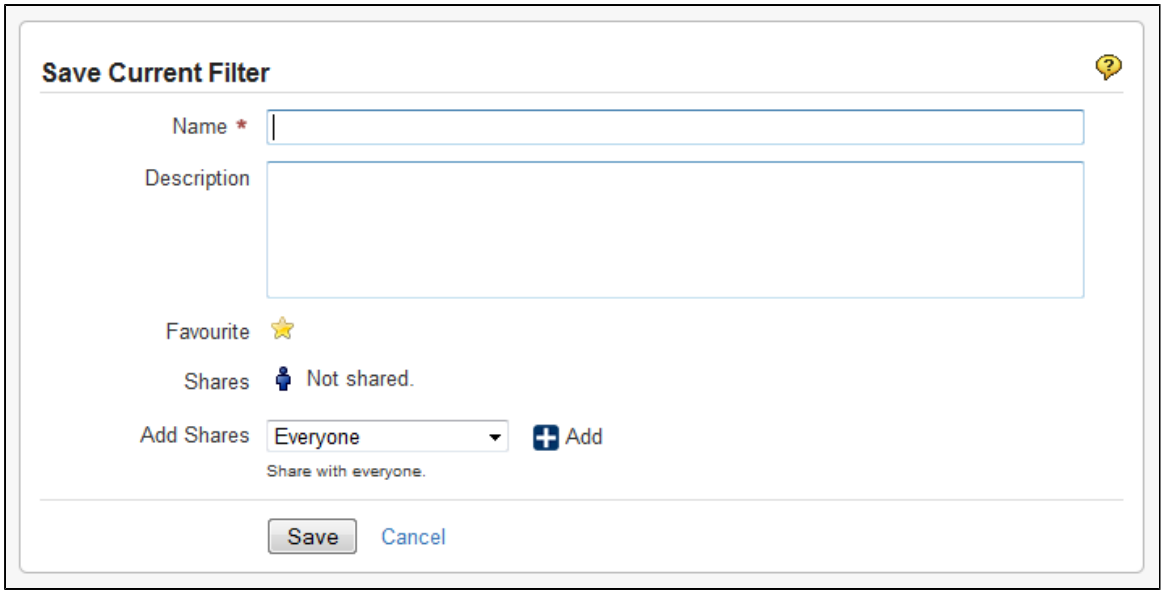
- display the search results in the Issue Navigator, where you can view and export them in various formats (RSS, Excel, etc)
- display the search results in a report format
- display the search results in a dashboard Gadget
- share the search with colleagues (see below)
- add another user's shared filter as a favourite (see below)
- have the search results emailed to you according to your preferred schedule

Saving a search as a Filter

1. Refine and run your search as described in 'Searching for Issues'.
2. Click the '**Save** it as a filter' link in the left-hand column of the Issue Navigator.



3. The 'Save Current Filter' page will display. Provide a name for the new filter and optionally enter a short description.



4. Your new filter will be added as a favourite filter by default upon creation. If you do not wish this filter to be added as a favourite, deselect the star icon. You can add the filter as a favourite after it has been created. Read more about adding an existing filter as a Favourite.
5. The sharing of your new filter is defaulted, depending on your sharing preference in your user profile. If you have not specified a personal preference, then the global default for sharing will apply (i.e. 'Private', unless changed by your JIRA Administrator under 'User Defaults' in the Administration menu). If you wish to change the sharing of your filter, refer to the instructions on sharing filters below.

Please note, you need the 'Create Shared Object' global permission to be able to share your filter. If you cannot see the 'Shares' fields, contact your JIRA Administrator to have this permission added to your profile.

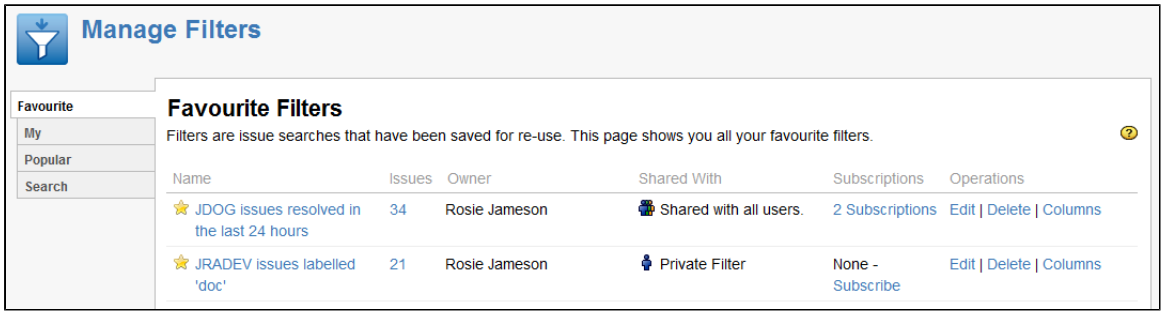
On this page:

- Saving a search as a Filter
- Managing your Filters
  - Adding a Filter as a Favourite
  - Sharing a Filter
  - Searching for a Filter
  - Updating a Filter's Details
  - Editing a Filter's Search Criteria
  - Cloning an Issue Filter
  - Defining filter-specific Issue Navigator Column Order
    - Adding a Column Order
    - Removing Column Order
    - Overriding Column Order
    - Exporting Column Ordered Issues
  - Subscribing to a Filter
- Managing Other User's Shared Filters

Managing your Filters

The '**Manage Filters**' page allows you to view and configure filters that you have created, as well as work with filters that other users have shared with you.

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. The 'Manage Filters' page will display. From this page, you can perform the functions listed below:
   - Create a new search to be saved as a filter.
   - Add a filter as a favourite.
   - Share a filter that you have created with other users.
   - Search for filters that has been created by you or shared with you by other users.
   - Update an existing filter's details or edit a filter's search criteria for a filter that you have created.

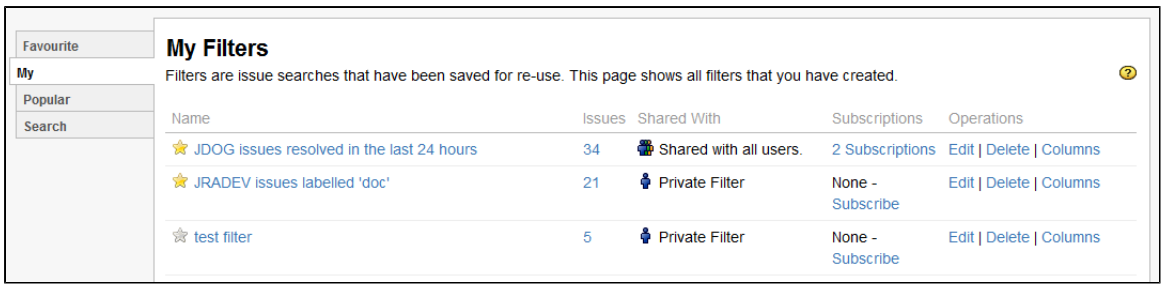


Adding a Filter as a Favourite

Issue filters that you created or that have been shared by other people can be added as a favourite filter. This means that the filter will appear in the 'Filters' dropdown in the top menu, as well as, display in the 'Favourite Filters' gadget on your dashboard (if you have this gadget added to your dashboard).

To add an existing shared filter as a favourite:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Locate the filter you wish to add as a favourite. If you created the filter, it will be listed under the 'My' tab, otherwise you can search for filters shared by other users via the 'Search' tab.
   - Filters that are already favourites are shown with a yellow star.
   - Filters that are not currently your favourites are shown with a grey star.
3. Click the star icon next to the filter name to select it as a favourite.

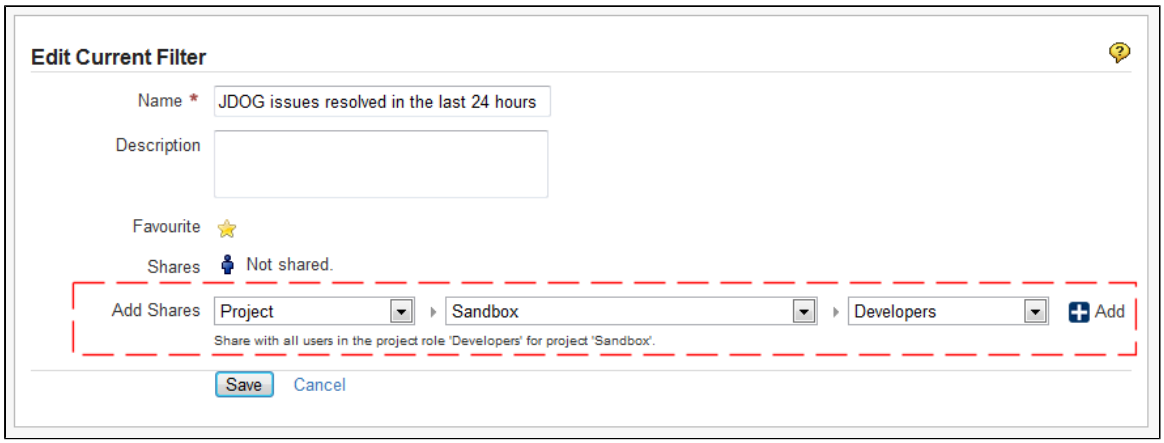


Sharing a Filter

Issue filters that you have created can be shared with other users via user groups, projects and project roles. Issue filters can also be shared globally. Note that if a filter is shared, it will also be visible to users who have the **'JIRA Administrators'** global permission via the 'Shared Filters' feature available in JIRA's Administration mode. See Managing Other User's Shared Filters below for details.

To share an existing filter:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Locate the filter you wish to share under the '**My**' tab, and click the '**Edit**' link in the '**Operations**' column.
3. Select the group, project or project role that you want to share the filter with, or share it with all users, if you wish. Click the '**Add**' link to add the share. You can add further share permissions if you wish.
   Note that you can only share filters with groups/roles of which you are a member.



4. Click '**Save**' to save your changes.

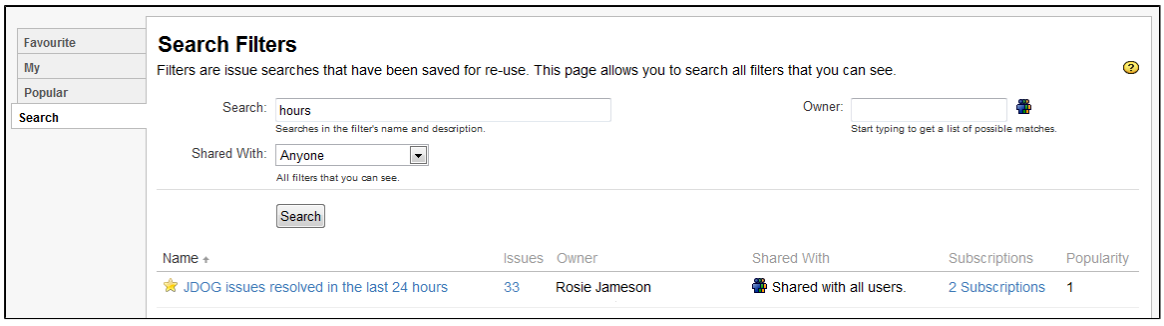
If the filter sharing functions described above are not available to you, you probably do not have the 'Create Shared Object' global permission assigned to you. Please contact your JIRA administrator to obtain this permission.

Searching for a Filter

Issue filters that you have created or have been shared by other users can be found via the issue filter search function of the 'Manage Filters' page. If the filter has been added as a favourite by many users, you also may be able locate it on the 'Popular' tab of the 'Manage Filters' page. This tab lists the top twenty most popular filters, counted by the number of users that have selected the filter as a favourite.

To search for an existing filter:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Click the '**Search**' tab. The issue filter Search will display. Enter your search criteria and click '**Search**' to run the search.



3. Your search results will be displayed on the same page. Click the name of any issue filter to run it and select it as your current filter. You can also sort the search results by any of the columns, by clicking the column headers.

Updating a Filter's Details

You can always update the details, i.e. Name, Description, Sharing, Favourite, of an existing Issue Filter after its creation. Please note that you can only update the details of Issue Filters which you have created.

To update the details of one of your existing filters:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Click the '**My**' tab. This tab displays all the filters that have been created by you.
3. Locate the filter you wish to update, and click the '**Edit**' link in the 'Operations' column.

4. The 'Save Current Filter' page will display. Update the filter details as required. If you wish to change the sharing or favourite settings for the filter, refer to the relevant instructions above.

5. Click the '**Save**' button to save your changes.

If the filter sharing functions described above are not available to you, you probably do not have the 'Create Shared Object' global permission assigned to you. Please contact your JIRA administrator to obtain this permission.

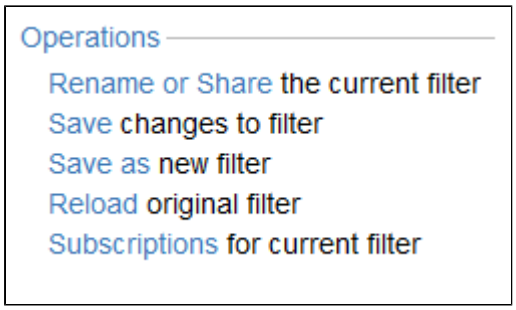
Editing a Filter's Search Criteria

The search criteria of an existing issue filter can always be changed after creation by editing the issue filter.

You can only edit filters that you have created. If you want to edit a filter that was shared with you by someone else, you will need to either clone the shared filter as described in the section below or ask your JIRA administrator to change the ownership of the shared filter, although you should inform the current owner of the shared filter of your intentions.

To edit an existing issue filter:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Locate the filter you wish to edit, and click the name of the issue filter to run it and select it as your current filter.
3. Click the '**Edit**' link at the top of the left hand menu. The left hand menu will refresh and the search criteria of the filter will display.
4. Modify the search criteria as required.
    - If you modified your search criteria in '**simple** searching' mode, go the '**Summary**' tab and click the '**View & Hide**' button. The filter operations will be displayed on the left hand side of the page.
5. To overwrite the current filter with the modified search criteria, click the '**Save** changes to filter' link.

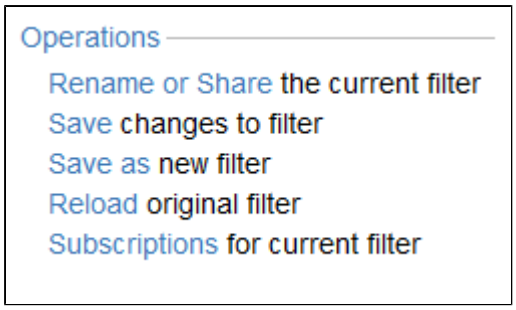


Cloning an Issue Filter

You can create a copy of any existing issue filter that was either created by you or was shared with you by its creator.

To clone an existing issue filter:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Locate the filter you wish to clone, and click the name of the issue filter to run it and select it as your current filter.
    - To clone the current filter with a new name (without modification to the filter's search criteria), ensure that the '**View**' link has been selected. Once the search results have displayed in the Issue Navigator on the right hand side of the page, click '**Save as** new filter' to clone the current filter with a new name and description.



OR:

- To clone a modified version of the current filter, first change the search criteria as you wish, then click the '**Create new** filter from current' link.

Defining filter-specific Issue Navigator Column Order

You can add an Issue Navigator Column Order to a saved filter. The results of a filter are displayed according to the saved column order, if the filter has one. Otherwise, the results are displayed according to your personal column order (if you have set this) or the system default.

> To display your configured column order in a filter subscription, you must select 'HTML' for the 'Outgoing email format' in your User Profile. If you are receiving text emails from JIRA, you will not be able to see your configured column order.

Adding a Column Order

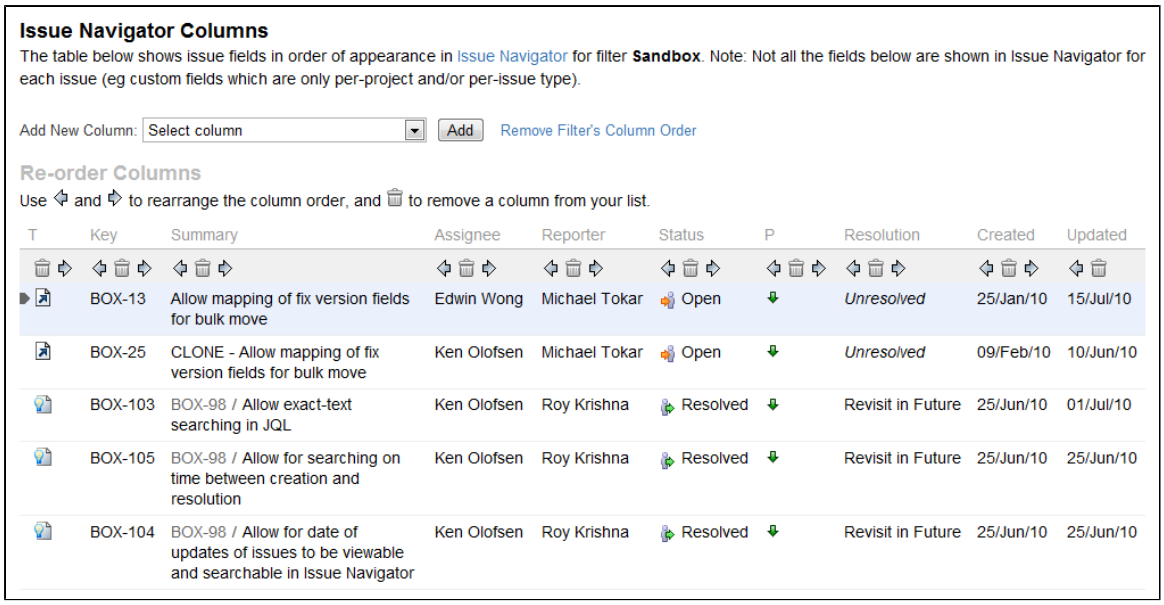
To add a column layout to a saved filter:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Select the '**My**' tab, locate the filter whose column layout you to reorder and save, and then click that filter's '**Columns**' link in the '**Operations**' column.
3. Configure the column order as desired. You can configure the column order the same way you would configure your personal Issue Navigator column order.

Removing Column Order

To remove a filter's saved column layout:

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. Select the '**My**' tab, locate the filter whose column layout you wish to remove and click that filter's '**Columns**' link in the '**Operations**' column.
3. Click the '**Remove Filter's Column Order**' link near the top of the page. The default column order will be restored.



Issue Navigator Columns

The table below shows issue fields in order of appearance in Issue Navigator for filter **Sandbox**. Note: Not all the fields below are shown in Issue Navigator for each issue (eg custom fields which are only per-project and/or per-issue type).

Add New Column: Select column Add Remove Filter's Column Order

Re-order Columns

Use and to rearrange the column order, and to remove a column from your list.

T	Key	Summary	Assignee	Reporter	Status	P	Resolution	Created	Updated
	BOX-13	Allow mapping of fix version fields for bulk move	Edwin Wong	Michael Tokar	Open		*Unresolved*	25/Jan/10	15/Jul/10
	BOX-25	CLONE - Allow mapping of fix version fields for bulk move	Ken Olofsen	Michael Tokar	Open		*Unresolved*	09/Feb/10	10/Jun/10
	BOX-103	BOX-98 / Allow exact-text searching in JQL	Ken Olofsen	Roy Krishna	Resolved		Revisit in Future	25/Jun/10	01/Jul/10
	BOX-105	BOX-98 / Allow for searching on time between creation and resolution	Ken Olofsen	Roy Krishna	Resolved		Revisit in Future	25/Jun/10	25/Jun/10
	BOX-104	BOX-98 / Allow for date of updates of issues to be viewable and searchable in Issue Navigator	Ken Olofsen	Roy Krishna	Resolved		Revisit in Future	25/Jun/10	25/Jun/10

Overriding Column Order

If a filter has a saved column order, the results will be presented using that column order when the filter is run. You can, however, choose to use your own column order (or the system default column order, if you do not have a personal one configured) to view the results. To do this, click the '**Use your default Column Order**' link on the right of the Issue Navigator search results screen.

To go back to using the filter's own column order, select the '**Use filter's Column Order**' link.

Exporting Column Ordered Issues

When the results of a saved filter are exported to Excel, the column order and choice of columns are those that were saved with the filter. Even if a user has configured a personal column order for the results on the screen, the **saved configuration** is used for the Excel export. To export using your own configuration, save a copy of the filter along with your configuration and then export the results to Excel.

Subscribing to a Filter

Please see Receiving Search Results via Email.

Managing Other User's Shared Filters

A **shared filter** is a filter whose creator has shared that filter with other users. Refer to Sharing a Filter above for details. When a shared filter is created by a user, that user:

- Initially 'owns' the shared filter.
- Being the owner, can edit and modify the shared filter.

If you have the **'JIRA Administrators'** global permission, you can manage shared filters that were created by other users.

To access the 'Shared Filters' feature:

1. Ensure that you are logged in as a user with the **JIRA Administrators** global permission.
2. On the top navigation bar, click the '**Issues**' dropdown and select '**Shared Filters**' from the list.
3. See Managing Shared Filters in the JIRA Administrator's Guide.

Receiving Search Results as an RSS Feed

JIRA enables you to subscribe to an RSS 2.0 feed that is based on any issue filter (saved search), or on your chosen search results, as displayed in the Issue Navigator.

You can choose either an RSS feed that contains issue data **(Summary, Description,** etc), or one that contains comments.

Note that the **tempMax** parameter can be used to control the maximum number of issues that are returned, e.g. sample RSS feed of the latest 15 issues reported on jira.atlassian.com.

On this page:

- Subscribing to an RSS Feed
- Accessing protected data
- Related Topics

Subscribing to an RSS Feed

To subscribe to an RSS feed,

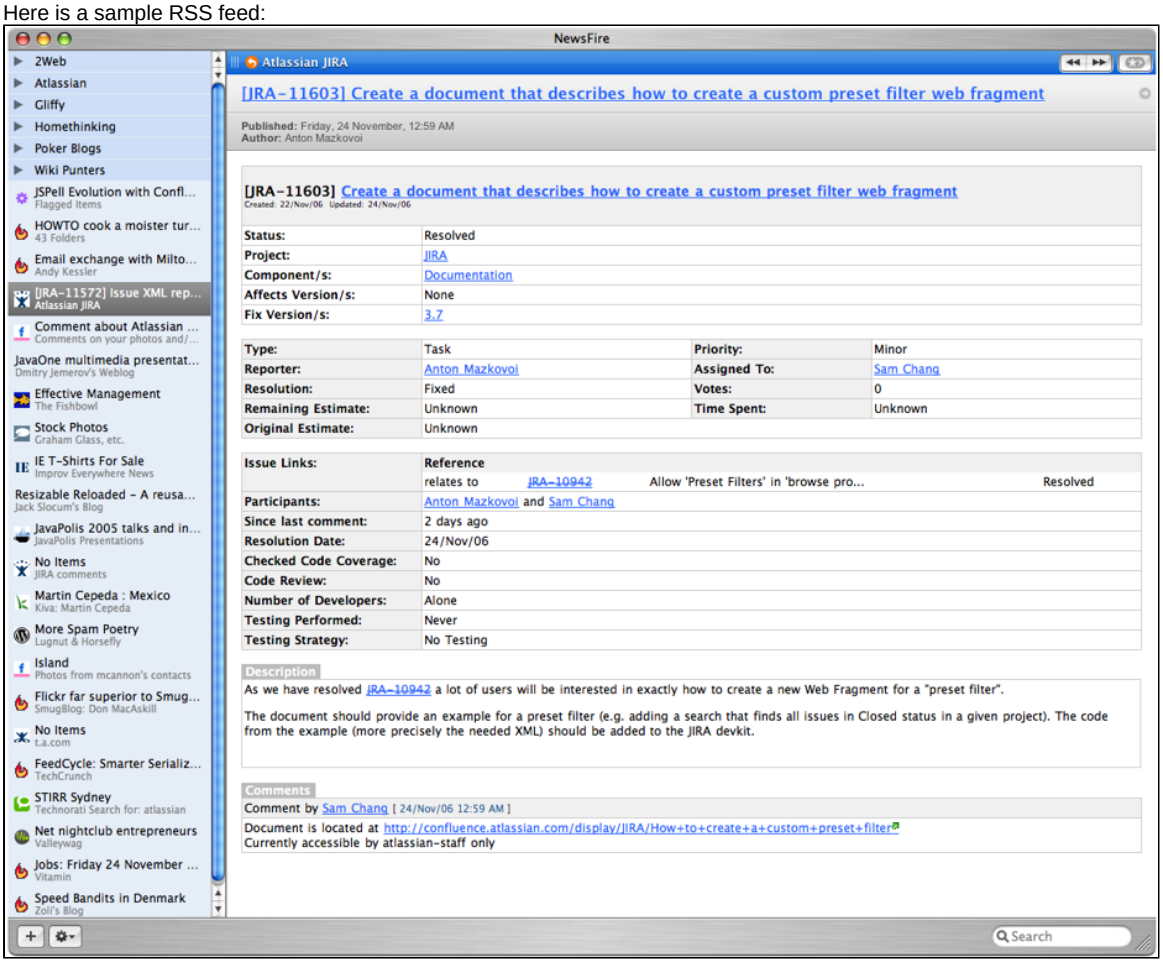
1. On the top navigation bar, click the **'Issues'** tab.
2. Refine your search, as described in *'Searching for Issues'*, until the required results are displayed in the Issue Navigator.
3. Click the **'Views'** menu, and select one of the following:
   - **'RSS (Issues)'** — this will create an RSS feed that contains just issue data.
   - **'RSS (Comments)'** — this will create an RSS feed that contains comments.

   ***HINT:** To only receive current comments, use the **Updated, Updated After** and **Updated Before** fields in the **'Dates and Times'** section of the search form. E.g. to only receive comments created in the last week, type `-1w` in the **From** field.*
4. Copy the URL that is currently displayed on your **Issue Navigator** screen.

   ***NOTE:** The **tempMax** parameter can be used to control the maximum number of issues returned in your RSS feed.*
5. Paste the URL into your RSS reader.

   ***NOTE:** If you are getting an empty RSS feed, make sure you have logged in to JIRA — see Accessing protected data (below).*

Here is a sample RSS feed:



Accessing protected data

When accessing data generated from JIRA, you may find that access to some resources requires user authentication (i.e. requires you to login). There are three options for this:

1. To enable access to data without logging in, your JIRA administrator may specify the 'Browse' permission for **Anyone.**
2. You can provide the parameters **os_username** and **os_password** in the request URL (e.g. `http://jira.atlassian.com/browse/TST-1?os_username=tester&os_password=tstpassword`). The problem with this method is that it transmits your username and password across the wire in clear text, which may not be an option for some users.
3. You can provide the request parameter **os_authType=basic** (e.g. `http://mycompany.com/anypage?os_authType=basic`). This will force the server to issue a challenge for user credentials (i.e. a login prompt) via the basic http authentication protocol. If you are running over SSL, you still need to specify the **os_authType=basic** parameter if you require the user to authenticate.

Related Topics

- Exporting Search Results to XML
- Adding the Activity Stream Gadget

Exporting Search Results to Microsoft Word

JIRA enables you to easily export your search results from the Issue Navigator to Microsoft Word. This can be a useful way to create reports in your own customised format.

On this page:

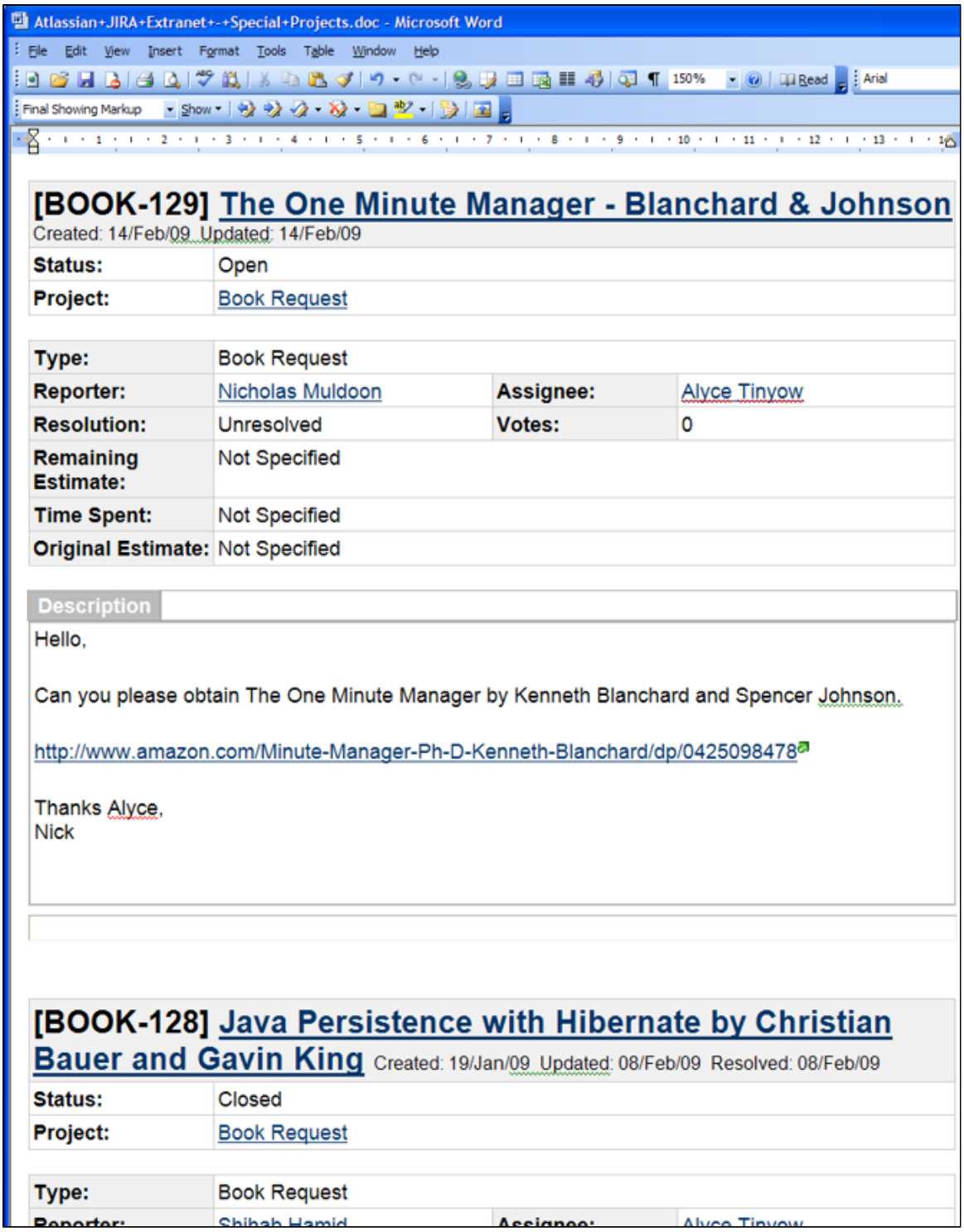
- Exporting to Microsoft Word
- Related Topics

Exporting to Microsoft Word

To export search results to Microsoft Word,

1. On the top navigation bar, click the **'Issues'** tab.
2. Refine your search, as described in *'Searching for Issues'*, until the required results are displayed in the Issue Navigator.
   Please note:
   - The export will include Description, Comments and all other issue data, not just the issue fields that are currently configured in your Issue Navigator.
   - Large exports (e.g. hundreds of issues) are not recommended.
3. Click the **'Views'** menu, and select **'Word'**.
4. A file called **'<My company's JIRA> - <My project>.doc'** will be created. Edit this file using Microsoft Word and/or save it as required.

Here is a sample exported file, viewed in Microsoft Word:



Related Topics

- Exporting Search Results to Microsoft Excel

Exporting Search Results to Microsoft Excel

JIRA enables you to easily export your search results from the Issue Navigator to Microsoft Excel. This can be a useful way to format data and create your own customised reports, graphs and charts.

On this page:

- Exporting to Microsoft Excel
- Related Topics

Exporting to Microsoft Excel

1. On the top navigation bar, click the **'Issues'** tab.
2. Refine your search, as described in *'Searching for Issues'*, until the required results are displayed in the Issue Navigator.

> Please note:
>
> - Large exports (e.g. many hundreds of issues) are not recommended.
> - To change the number of issues that are exported, change the value of the **tempMax** parameter in the URL.

3. Click the **'Views'** menu, and select one of the following:
   - **'Excel (All fields)'** — this will create a spreadsheet column for every issue field (excluding comments).
   - **'Excel (Current fields)'** — this will create a spreadsheet column for the issue fields that are currently displayed in your Issue Navigator.
4. A file called **'<My company's JIRA> - <My project>.xls'** will be created. Edit this file using Microsoft Excel and/or save it as required.

Here is a sample exported file, viewed in Microsoft Excel:

Microsoft Excel - Atlassian+JIRA+Extranet+-+Special+Projects-1.xls [Read-Only]

File Edit View Insert Format Tools Data Window Help

JIRA

Atlassian JIRA Extranet - Special Projects

Displaying **128** issues as at **16/Feb/09 10:54 PM**.

Issue Type	Key	Summary	Assignee	Reporter	Status	Resolution
Book Request	BOOK-129	The One Minute Manager - Blanchard & Johnson	Alyce Tinyow	Nicholas Muldoon	Open	UNRESOLVED
Book Request	BOOK-128	Java Persistence with Hibernate by Christian Bauer and Gavin King	Alyce Tinyow	Shihab Hamid	Closed	Fixed
Book Request	BOOK-127	Beginning iPhone Development: Exploring the iPhone SDK (Paperback)	Alyce Tinyow	Andreas Knecht	In Progress	UNRESOLVED
Book Request	BOOK-126	Please order 10 copies of Mike Cohn's "User Stories Applied"	Alyce Tinyow	Melanie Carasso	Closed	Fixed
Book Request	BOOK-125	The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws (Paperback)	Alyce Tinyow	Anton Mazkovoi	Open	UNRESOLVED
Book Request	BOOK-124	Groundswell	Alyce Tinyow	Robyn Munro	Open	UNRESOLVED
Book Request	BOOK-123	Web Security Testing Cookbook	Alyce Tinyow	Andrew Prentice	Closed	Fixed
Book Request	BOOK-122	Book request, Pulling Strings with Puppet	Alyce Tinyow	David Cheney	Closed	Fixed
Book Request	BOOK-121	Book request, Automating Linux System Administration	Alyce Tinyow	David Cheney	Closed	Fixed
Book Request	BOOK-120	Please order 4 (four) copies of "Do you Matter: How Great Design will make People Love your Company" for Design Team	Alyce Tinyow	Jay Rogers	Closed	Fixed
Book Request	BOOK-119	User Stories Applied: For Agile Software Development (Addison-Wesley Signature Series) (Paperback)	Alyce Tinyow	Audra Eng	Closed	Fixed
Book Request	BOOK-118	Practical API Design	Alyce Tinyow	Wesley-Smith	Closed	Fixed
Book Request	BOOK-117	The First 90 Days: Critical Success Strategies for New Leaders at All Levels	Alyce Tinyow	Jay Rogers	Closed	Fixed
Book Request	BOOK-116	Adrenaline Junkies and Template Zombies: Understanding Patterns of Project Behavior	Alyce Tinyow	Jay Rogers	Closed	Fixed
Book Request	BOOK-115	Please order another copy of "Java Power Tools [ILLUSTRATED] (Paperback)"	Alyce Tinyow	Alex Wei	Closed	Won't Fix
Book Request	BOOK-114	Programming-Collective-Intelligence	Alyce Tinyow	Nick Pellow	Closed	Task completed
Book Request	BOOK-113	Please buy this book!	Alyce Tinyow	Stephen Russell	Open	UNRESOLVED
Book Request	BOOK-112	SOA Using Java Web Services for IJUG sponsosrhip	Alyce Tinyow	Robyn Munro	Closed	Fixed
Book Request	BOOK-111	Quarterly marketing books	Rochell Lopez	Ben Naftzger	Closed	Task completed
Book Request	BOOK-110	2 Copies of Maven: The Defintive Guide	Rochell Lopez	James Dumay	Closed	Task completed
Book Request	BOOK-109	Book request: Greasemonkey Hacks	Alyce Tinyow	Jim Severino	Closed	Won't Fix
Book Request	BOOK-108	The Pragmatic Programmer: From Journeyman to Master (ISBN 0-201-61622-X)	Alyce Tinyow	Eric Dalgliesh	Closed	Won't Fix
Book Request	BOOK-107	Request for 'Influence: The Psychology of Persuasion' book by Robert B. Cialdini	Alyce Tinyow	David Soul	Resolved	Fixed
Book Request	BOOK-106	Request for 'Hug Your Customers' book by Jack Mitchell	Alyce Tinyow	David Soul	Resolved	Fixed
Book Request	BOOK-105	Ajax Security by Addison-Wesley Professional	Alyce Tinyow	Scott Harwood	Closed	Fixed
Book Request	BOOK-104	Network Warrior [ILLUSTRATED] (Paperback)	Alyce Tinyow	Steve Smith	Closed	Fixed
Book Request	BOOK-103	Practice of System and Network Administration, The (2nd Edition) (Paperback)	Alyce Tinyow	Steve Smith	Closed	Fixed

general_report

Ready

Related Topics

- Exporting Search Results to Microsoft Word

Displaying Search Results as a Chart

You can view any search results from the Issue Navigator in a variety of charting formats. You can also save them as a **Charting gadget** on your dashboard, as described below.

The **Charts** display visual representations of a filter in a variety of ways. In general, charts are:

- for a period of X days previous to the current date.
- broken down into incremental periods of hours, days, weeks, months, quarters or years.
- hyperlinked to relevant issues in the Issue Navigator.

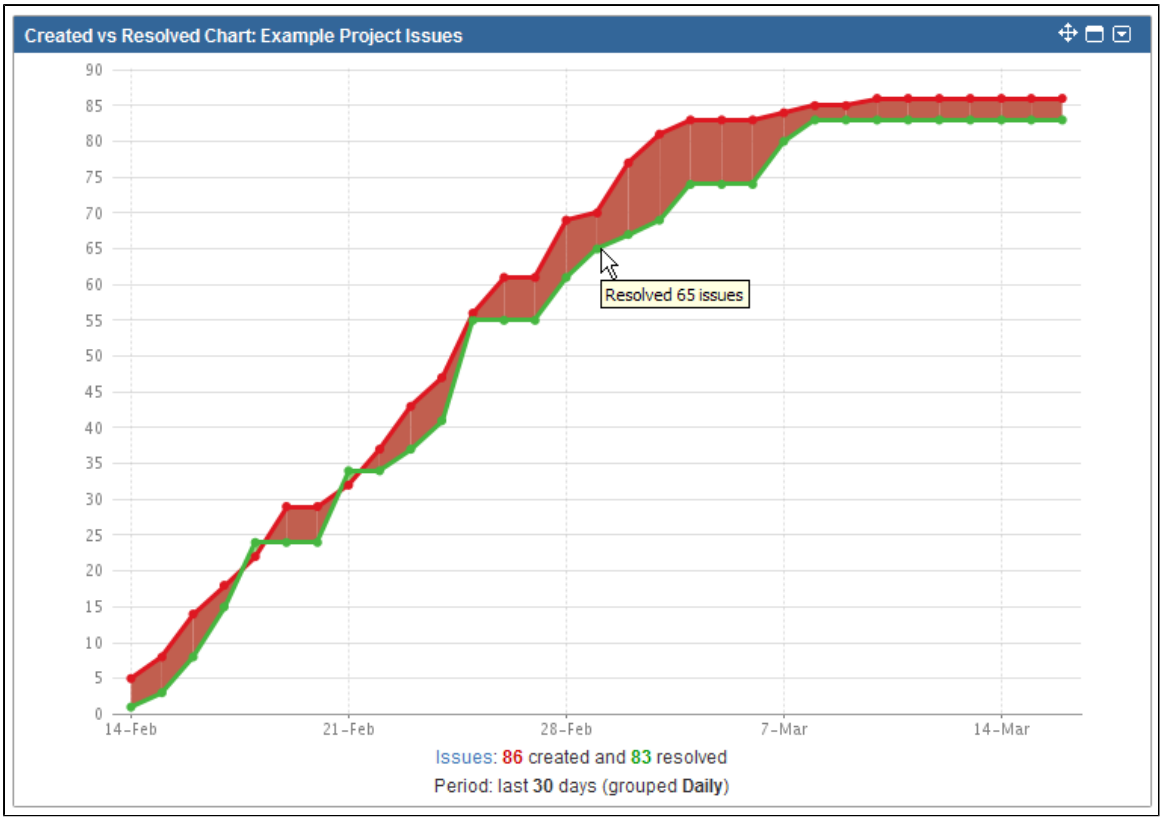
On this page:

- What do they look like?
- Viewing a Chart
- Adding Gadgets to your Dashboard
- Configuring your Internet Explorer cache settings

What do they look like?

The **'Created vs Resolved Issues'** chart, for example, appears as follows:

Screenshot: 'Created vs Resolved Issues' chart



This is just one of the available charting gadgets, which include:

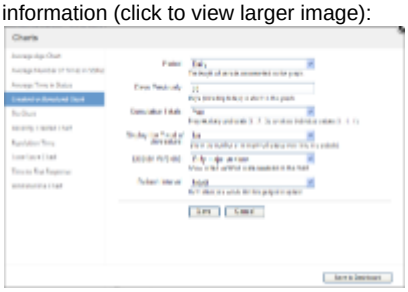
- **'Created vs Resolved Issues'** — a difference chart showing the issues created vs resolved over a given period.
  - This chart can either be cumulative or not.
  - Areas in red show periods where more issues were created than resolved, areas in green show periods where more were resolved than created.
  - Versions can also be added to this chart, showing you how issue creation and resolution related to version releases.
- **'Resolution Time'** — a bar chart showing the average resolution time (in days) of resolved issues.
  - This is useful to show you over time whether your team is getting better or worse at resolving issues in a timely fashion.
- **'Pie Chart'** — displays issues grouped by a statistic type in pie-chart format
  - The issues can be grouped by any statistic type (e.g. Status, Priority, etc.)
- **'Average-Age Open Issues'** — a bar chart showing the average number of days that issues have been unresolved
  - This chart displays the average of how long issues remain open on a specified interval (e.g. daily, weekly, etc.)
- **'Recently Created Issues'** — a bar chart showing the issues recently created.
  - The green portion of the bar shows the created issues which are resolved, the red portion shows created but as yet unresolved issues.
  - This visually shows both how quickly you're creating issues, as well as how many of those created issues are resolved.

- **'Time Since Issues'** — displays a bar chart showing the number of issues for which your chosen date field (e.g. 'Created', 'Updated', 'Due', 'Resolved', or a date custom field) was set on a given date.
- **'Average Time in Status'*** — displays the average number of days issues have spent in a status.
- **'Average Number of Times in Status'** [1] — displays the average number of times an issues has been in a status.
- **'Time to First Response'*** — displays the average number of times an issues has been in a status.

[1] **This particular chart will only be available if your JIRA administrator has installed the Charting plugin.**

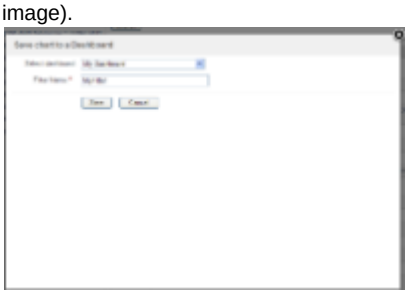
Viewing a Chart

To view your search results as a chart,

1. On the top navigation bar, click on the **'Issues'** tab.
2. Refine your search, as described in *'Searching for Issues'*, until the required results are displayed in the Issue Navigator.
3. Click the **'Views'** menu, and select **'Charts'.**
4. Your search results will be displayed as the default chart. If you wish to select a different type of chart,
    a. Select the desired **'Chart Type'**, as listed in the previous section.
    b. If you wish to edit the chart parameters click the cog icon and click '**Edit**' from the dropdown menu that displays. The configuration form for the chart will display. For example, the **'Created vs Resolved Issues'** chart requires the following information (click to view larger image):
    
    c. Update the chart settings as desired.
    d. Click **'Save'**.

Adding Gadgets to your Dashboard

To create a chart based on your search results and display it on your dashboard (note that this process will also create a saved filter):

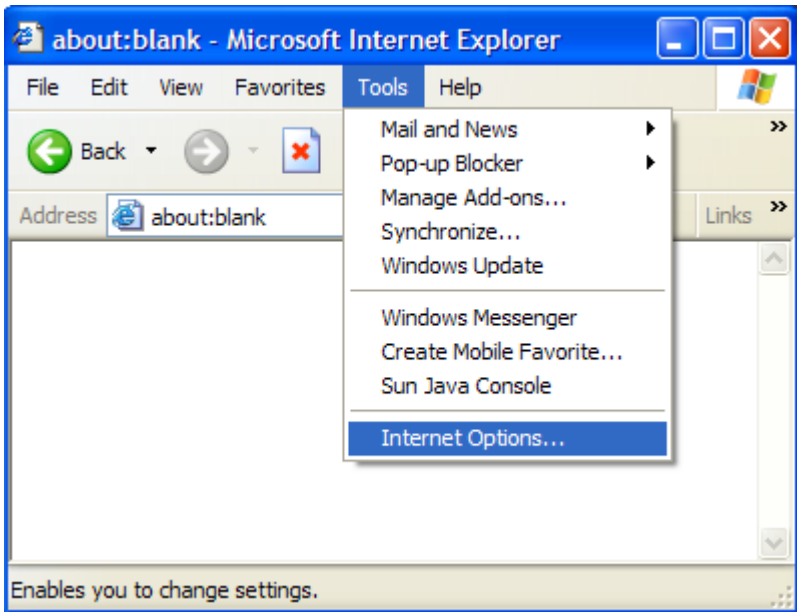
1. View the desired search results in your Issue Navigator.
2. Click the **'Views'** menu, and select **'Charts'.**
3. Your search results will be displayed as the default chart. If you wish to select a different type of chart, you can change the chart and chart settings as described in Viewing a Chart above.
4. Click the '**Save to Dashboard**' button. The 'Save chart to a Dashboard' screen will display as shown below (click to view larger image).

5. Select the dashboard to display the chart on in the '**Select dashboard:**' field and type a name for your search results in the **'Filter Name'** field, then click the **'Save'** button. The chart will now appear as a gadget on your dashboard.
6. If you wish, the gadget can be repositioned on the dashboard through the dashboard configuration page.

Further information on all JIRA dashboard gadgets is available in the Using Dashboard Gadgets documentation.

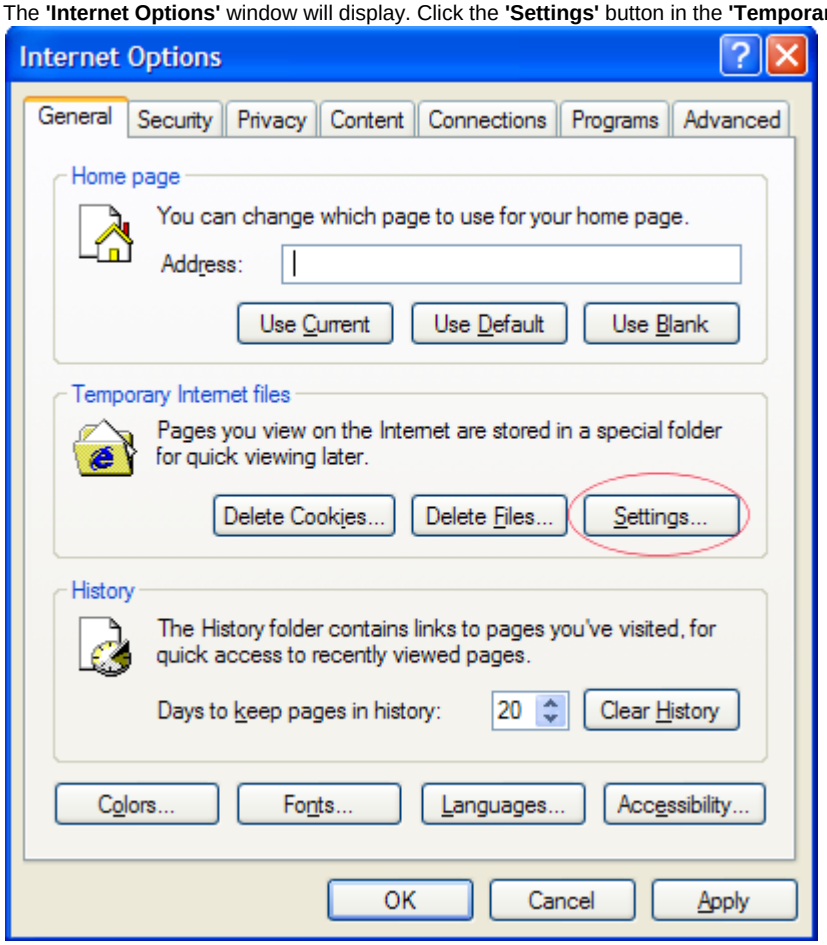
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

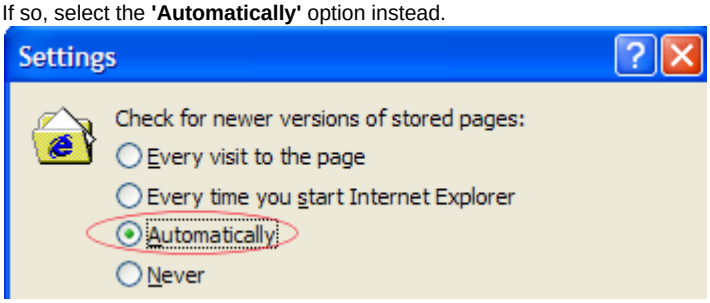
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Displaying Search Results in XML

The JIRA Issue Navigator enables you to display your search results in XML format, suitable for use with the Confluence JIRA Issues Macro, for example. (Also suitable for use as an RSS 0.9.2 feed. Note: for an RSS 2.0 feed, please see 'Receiving Search Results via RSS'.)

You can choose which fields to include in the XML output, as described below.

Exporting to XML

To display your search results in XML format,

1. On the top navigation bar, click the **'Issues'** tab.
2. Refine your search, as described in *'Searching for Issues',* until the required results are displayed in the Issue Navigator.
3. Click the **'Views'** menu, and select **'XML'**.
4. Your search results will be displayed in XML format. Note:
    - If you wish to restrict which fields are included in the XML output, use the * `field` * parameter as described below.
    - To choose how many issues are included in the XML output, change the value of the **`tempMax`** parameter. The default is 1,000 issues (i.e. `tempMax=1000`).
5. Copy the URL that is currently displayed on your screen.
6. If you are using the Confluence **JIRA Issues Macro,** paste the URL into your Confluence document as described in the Confluence documentation.

On this page:

- Exporting to XML
- Choosing which fields to include
- Accessing protected data

Choosing which fields to include

To restrict which issue fields are returned in the XML export, specify the `field` parameter in your URL. For example, to include only the **Issue key** and **Summary,** add `&field=key&field=summary` to the URL.

Note:

- If the `field` parameter is not specified, the XML output will include *all* the issue fields.
- If one or more `field` parameters are specified, the XML output will contain only the **Issue key** plus your chosen field(s).

The complete list of available values for the `field` parameter is as follows:

Value	Sample XML output
`title`	`<title>[TEST-4] This is a test</title>`
`link`	`<link>https://extranet.atlassian.com:443/jira/browse/TEST-4</link>`
`project` (or **`pid`**)	`<project id="10330" key="TST">Test</project>`
`description`	`<description>This is a detailed description of the issue.<descriptior`
`environment`	`<environment>Sydney network<environment>`
`key`	`<key id="22574">TEST-4</key>`

summary	`<summary>This is a test<summary>`
type (or **issuetype**)	`<type id="3" iconUrl="https://extranet.atlassian.com:443/jira/images/`
parent	`<parent id="22620">TEST-5</parent>`
priority	`<priority id="4"` `iconUrl="https://extranet.atlassian.com:443/jira/images/icons/priorit`
status	`<status id="5"` `iconUrl="https://extranet.atlassian.com:443/jira/images/icons/status_`
resolution	`<resolution id="1">Fixed</resolution>`

labels	`<labels>`
	`<label>focus</label>`
	`<labels>`

assignee	`<assignee username="jsmith">John Smith</assignee>`
reporter	`<assignee username="jsmith">John Smith</assignee>`
security	`<security id="10021">Private</security>`
created	`<created>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/created>`
updated	`<updated>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/updated>`
resolved (or **resolutiondate**)	`<resolved>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/resolved>`

due (or **duedate**)	`<due>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/due>`
version (or **versions**)	`<version>2.4.7</version>`
fixfor (or **fixVersions**)	`<fixVersion>2.6</fixVersion>`
component (or **components**)	`<component>Documentation</component>`
votes	`<votes>1</votes>`

comments (or **comment**)	`<comments>`
	`<comment id="39270" author="jsmith" created="Tue, 24 Feb 2009 16:`
	`familiar</comment>`
	`<comment id="39273" author="jbrown" created="Tue, 24 Feb 2009 16:`
	`too</comment>`
	`</comments>`

attachments (or **attachment**)	`<attachments>`
	`<attachment id="30318" name="Issue Navigator - Atlassian JIRA-2.pr`
	`created="Mon, 9 Feb 2009 13:32:58 -0600 (CST)"/>`
	`<attachment id="30323" name="Windows XP (with Firefox 3.0).jpg" si`
	`created="Tue, 10 Feb 2009 00:30:11 -0600 (CST)"/>`
	`</attachments>`

timeoriginalestimate	`<timeoriginalestimate seconds="600">10 minutes</timeoriginalestimate>`
timeestimate	`<timeestimate seconds="300">5 minutes</timeestimate>`
timespent	`<timespent seconds="300">5 minutes</timespent>`
aggregatetimeoriginalestimate	`<aggregatetimeoriginalestimate seconds="36000">10 hours</aggregatetim`
aggregatetimeestimate	`<aggregatetimeremainingestimate seconds="18000">5 hours</aggregatetim`

aggregatetimespent	`<aggregatetimespent seconds="18000">5 hours</aggregatetimespent>`

timetracking	`<timeoriginalestimate seconds="600">10 minutes</timeoriginalestimate>`
	`<timeestimate seconds="300">5 minutes</timeestimate>`
	`<timespent seconds="300">5 minutes</timespent>`
	`<aggregatetimeoriginalestimate seconds="36000">10 hours</aggregatetim`
	`<aggregatetimeremainingestimate seconds="18000">5 hours</aggregatetim`
	`<aggregatetimespent seconds="18000">5 hours</aggregatetimespent>`

issuelinks	`<issuelinks>`
	`  <issuelinktype id="10020">`
	`  <name>Duplicate</name>`
	`  <inwardlinks description="is duplicated by">`
	`  <issuelink>`
	`  <issuekey id="22477">INTSYS-1009</issuekey>`
	`  </issuelink>`
	`  </inwardlinks>`
	`  </issuelinktype>`
	`</issuelinks>`

subtasks (or **subtask**)	`<subtasks>`
	`  <subtask id="22623">TEST-8</subtask>`
	`</subtasks>`

customfield_xxxxx	`<customfields>`
	`  <customfield id="customfield_10112"`
	`key="com.atlassian.jira.plugin.system.customfieldtypes:select">`
	`  <customfieldname>Department</customfieldname>`
	`  <customfieldvalues>`
	`  <customfieldvalue>Adminstration</customfieldvalue>`
	`  </customfieldvalues>`
	`  </customfield>`
	`</customfields>`

allcustom	`<customfields>`
	`  <customfield id="customfield_10112"`
	`key="com.atlassian.jira.plugin.system.customfieldtypes:select">`
	`  <customfieldname>Department</customfieldname>`
	`  <customfieldvalues>`
	`  <customfieldvalue>Adminstration</customfieldvalue>`
	`  </customfieldvalues>`
	`  </customfield>`
	`  <customfield id="customfield_10111"`
	`key="com.atlassian.jira.plugin.system.customfieldtypes:select">`
	`  <customfieldname>Expenditure Type</customfieldname>`
	`  <customfieldvalues>`
	`  <customfieldvalue>Operating</customfieldvalue>`
	`  </customfieldvalues>`
	`  </customfield>`
	`</customfields>`

Accessing protected data

When accessing data generated from JIRA, you may find that access to some resources requires user authentication (i.e. requires you to login). There are three options for this:

1. To enable access to data without logging in, your JIRA administrator may specify the 'Browse' permission for **Anyone.**
2. You can provide the parameters **os_username** and **os_password** in the request URL (e.g. `http://jira.atlassian.com/browse/TST-1?os_username=tester&os_password=tstpassword`). The problem with this method is that it transmits your username and password across the wire in clear text, which may not be an option for some users.
3. You can provide the request parameter **os_authType=basic** (e.g. `http://mycompany.com/anypage?os_authType=basic`). This will force the server to issue a challenge for user credentials (i.e. a login prompt) via the basic http authentication protocol. If you are running over SSL, you still need to specify the **os_authType=basic** parameter if you require the user to authenticate.

Receiving Search Results via Email

JIRA enables you to *subscribe* to an issue filter (a saved search). JIRA will then run the search according to your specified schedule, and will email the results to you.

You can specify when and how often you would like to receive the search results, e.g. 'Every hour between 9.00AM-5.00PM, Monday-Friday' or 'The last Friday of every month at 7.00AM'.

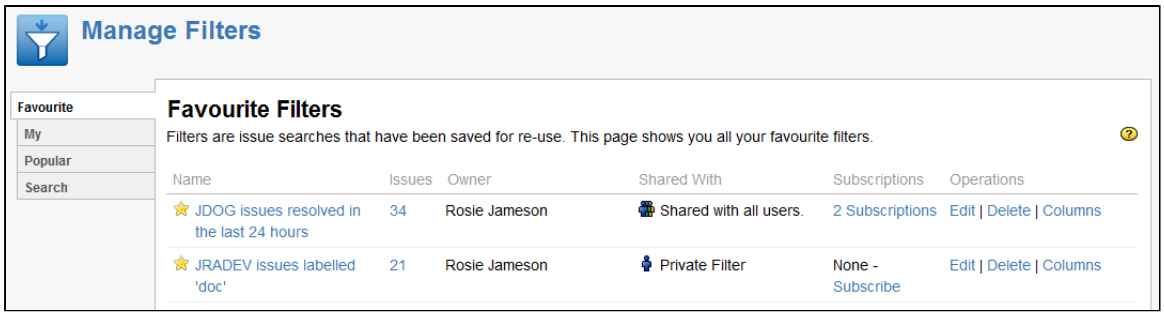
Emails can only be sent if your administrator has configured an SMTP mail server.

On this page:

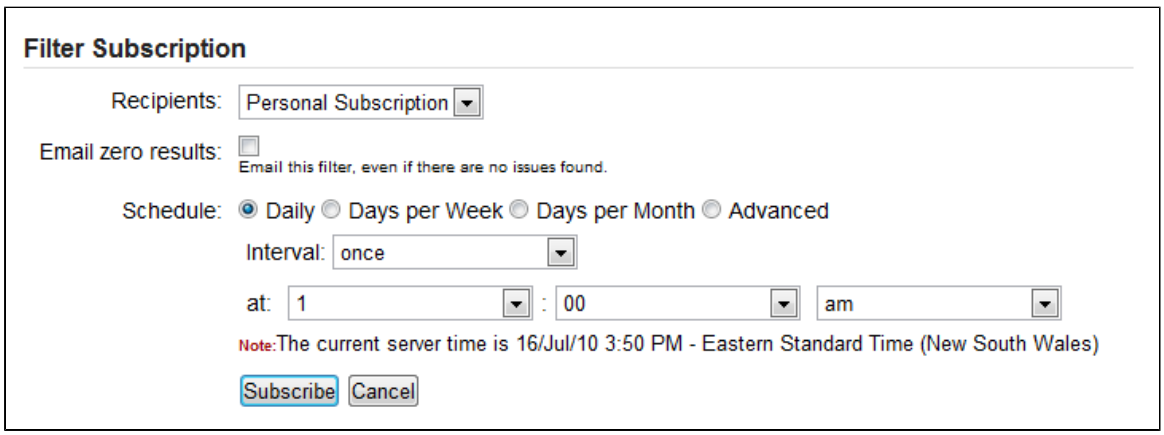
- Subscribing to a Filter
- Advanced scheduling ('cron')
- Managing Other User's Shared Filters

Subscribing to a Filter

1. On the top navigation bar, click the '**Issues**' dropdown and select '**Manage Filters**' from the list.
2. A list of available filters will be displayed:



3. Locate the filter you are interested in and click on its **'Subscribe'** link. The **'Filter Subscription'** form will be displayed:



4. In the **'Recipients'** box, either choose 'Personal Subscription' (if you only wish to subscribe yourself), or select a group of recipients from the drop-down list. Note that you cannot select a *group* unless your JIRA administrator has granted you the 'Manage Group Filter Subscriptions' global permission.
5. Select **'Email zero results'** if you would like the email to always be sent, even if there are no search results at that time.
6. Select one of the following types of schedule:
   - **'Daily'** — choose this if you want to receive one or more emails *every day*.

- **'Days per Week'** — choose this if you want to receive one or more emails on *particular days of every week*.
- **'Days per Month'** — choose this if you want to receive an email on *a particular day of every month*.
- **'Advanced'** — see *Advanced scheduling ('cron')* below.

7. Click the '**Subscribe**' button.
8. You will now be shown a subscription summary page. If you wish, you can click **'Run now'** to test your subscription.

Advanced scheduling ('cron')

You can use a **'Cron Expression'** to specify a custom schedule to suit your particular requirements.

Cron expressions consist of the following fields, separated by spaces:

Field	Allowed values	Allowed special characters
Second	0-59	, - * /
Minute	0-59	, - * /
Hour	0-23	, - * /
Day-of-month	1-31	, - * / ? L W C
Month	1-12 or JAN-DEC	, - * /
Day-of-week	1-7 or SUN-SAT	, - * / ? L C #
Year *(optional)*	1970-2099	, - * /

The special characters operate as follows:

Special character	Usage
,	Specifies a list of values. For example, in the **Day-of-week** field, 'MON,WED,FRI' means 'every Monday, Wednesday, and Friday'.
-	Specifies a range of values. For example, in the **Day-of-week** field, 'MON-FRI' means 'every Monday, Tuesday, Wednesday, Thursday and Friday'.
*	Specifies all possible values. For example, in the **Hour** field, '*' means 'every hour of the day'.
/	Specifies increments to the given value. For example, in the **Minute** field, '0/15' means 'every 15 minutes during the hour, starting at minute zero'.
?	Specifies no particular value. This is useful when you need to specify a value for one of the two fields **Day-of-month** or **Day-of-week,** but not the other.
L	Specifies the last possible value; this has different meanings depending on context. In the **Day-of-week** field, 'L' on its own means 'the last day of every week' (i.e. 'every Saturday'), or if used after another value, means 'the last xxx day of the month' (e.g. 'SATL' and '7L' both mean 'the last Saturday of the month). In the **Day-of-month** field, 'L' on its own means 'the last day of the month', or 'LW' means 'the last weekday of the month'.
W	Specifies the weekday (Monday-Friday) nearest the given day of the month. For example, '1W' means 'the nearest weekday to the 1st of the month' (note that if the 1st is a Saturday, the email will be sent on the nearest weekday *within the same month,* i.e. on Monday 3rd). 'W' can only be used when the day-of-month is a single day, not a range or list of days.
#	Specifies the nth occurrence of a given day of the week. For example, 'TUES#2' (or '3#2') means 'the second Tuesday of the month'.

Here are some sample cron expressions:

`0 15 8 ? * *`	Every day at 8.15 pm.
`0 15 8 * * ?`	Every day at 8.15 am.
`0 * 14 * * ?`	Every minute starting at 2.00 pm and ending at 2:59 pm, every day.
`0 0/5 14 * * ?`	Every 5 minutes starting at 2.00 pm and ending at 2:55 pm, every day.
`0 0/5 14,18 * * ?`	Every 5 minutes starting at 2.00 pm and ending at 2:55 pm, AND every 5 minutes starting at 6.00 pm and ending at 6:55 pm, every day.
`0 0-5 14 * * ?`	Every minute starting at 2.00 pm and ending at 2:05 pm, every day.
`0 0/10 * * * ? *`	Every 10 minutes, forever.
`0 10,44 14 ? 3 WED`	2:10 pm and 2:44 pm every Wednesday in the month of March.
`0 15 8 ? * MON-FRI`	8:15 am every Monday, Tuesday, Wednesday, Thursday and Friday.
`0 15 8 15 * ?`	8:15 am on the 15th day of every month.
`0 15 8 L * ?`	8:15 am on the last day of every month.
`0 15 8 LW * ?`	8:15 am on the last weekday of every month.
`0 15 8 ? * 6L`	8:15 am on the last Friday of every month.
`0 15 8 ? * 6#2`	8:15 am on the second Friday of every month.
`0 15 8 ? * 6#2 2007-2009`	8:15 am on the second Friday of every month during the years 2007, 2008 and 2009.

Cron expressions are not case-sensitive

Managing Other User's Shared Filters

A **shared filter** is a filter whose creator has shared that filter with other users. Refer to Sharing a Filter for details. When a shared filter is created by a user, that user:

- Initially 'owns' the shared filter.
- Being the owner, can edit and modify the shared filter.

If you have the **'JIRA Administrators'** global permission, you can manage shared filters that were created by other users.

To access the 'Shared Filters' feature:

1. Ensure that you are logged in as a user with the **JIRA Administrators** global permission.
2. On the top navigation bar, click the '**Issues**' dropdown and select '**Shared Filters**' from the list.
3. See Managing Shared Filters in the JIRA Administrator's Guide.

Sharing a Search Result

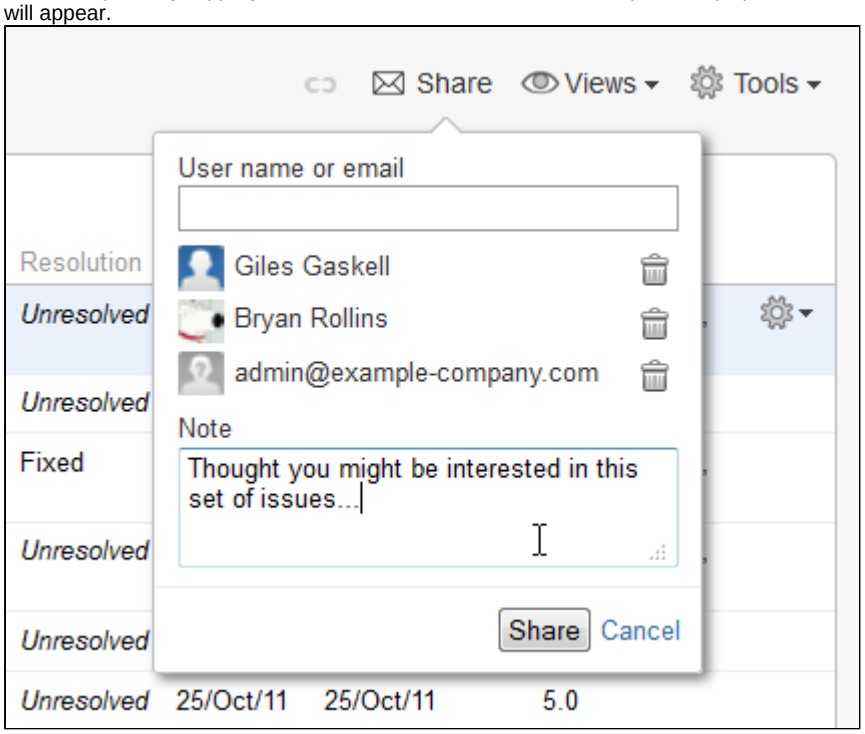
You can easily email other JIRA users (including any email address) a link to a search result or shared filter by 'sharing' the search result (or shared filter) with them. You can also add an optional note to the email message.

To access the **Share** button, your **JIRA System Administrator** must first have configured JIRA's SMTP mail server. Additionally, you also require the **Browse Users** global permission.

To share a search result with one or more JIRA users or any email addresses:

1. View the search result or shared filter you want to share.
2. Click the **Share** button at the top-right.
   Keyboard shortcut: **s**

3. Specify JIRA users (by typing their usernames or part/all of their full names as registered with their JIRA user accounts) or type *any* email addresses of people you want to share the issue with.
   When you begin typing a JIRA user's username or name, or a previously specified email address, an autocomplete dropdown will appear.



4. Add an optional **Note**.
5. Click the **Share** button.

Please Note:

- Recipients specified in the **User name or email** field will receive an email message whose body contains the content of the **Note** (if one was specified) as well as a link to the search result.
- A shared search result sent to JIRA users specified in the **User name or email** field will be sent to the email addresses registered with these user's respective JIRA accounts.
- The subject line of the email message will specify you as the JIRA user who 'shared' the issue with the recipients.
- If you are viewing a shared filter on the issue navigator and use the **Share** button to share that filter with other JIRA users who can also view this filter, then the email message that these JIRA users receive will contain a link to the filter instead of its search results. All other recipients will receive an email message containing a link to that filter's search results instead.
- You can also share an issue from the 'view issue' page. See Sharing an issue for details.

Generating Reports

JIRA provides reports that show statistics for particular people, projects, versions, or other fields within issues. The following reports are included with JIRA:

- Workload Pie Chart Report * — Shows the relative workload for assignees of all issues in a particular project or issue filter.
- User Workload Report * — Shows how much work a user has been allocated, and how long it should take.
- Version Workload Report * — Shows how much outstanding work there is (per user and per issue) before a given version is complete.
- Version Time Tracking Report * — Shows progress towards completing a given version, based on issues' work logs and time estimates.
- Single Level Group By Report — Shows the search results from an issue filter, grouped by a field of your choice.
- Created vs Resolved Issues Report — Shows the number of issues created vs number of issues resolved over a given period of time.
- Resolution Time Report — Shows the average time taken to resolve issues.
- Pie Chart Report — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- Average Age Report — Shows the average age (in days) of unresolved issues.
- Recently Created Issues Report — Shows the rate at which issues are being created.
- Time Since Issues Report — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.

** Only available if your JIRA administrator has enabled Time Tracking.*

Additional Reporting is available!

- In addition to the built-in reports, other reports (e.g. Gantt Chart Report, Timesheet Report, JIRA SQL Plugin) are available for download from the Atlassian Plugin Exchange.
- JIRA administrators can also create new reports with the plugin API — see our Plugin Tutorial – Creating a JIRA Report. If you're not wanting to build a plugin yourself, Atlassian Experts are available for custom projects.
- Issue Filters can be exported to Microsoft Excel, where they can be further manipulated into charts and reports. See Exporting Search Results to Microsoft Excel.
- Confluence can work as a tool for business reporting. See Confluence Reporting HOWTO, in conjunction with Confluence's SQL plugin and Example SQL queries for JIRA.

Workload Pie Chart Report

The 'Workload Pie Chart' report displays the relative workload for assignees of all issues in a particular project or issue filter.

Note: this report is only available if your JIRA administrator has enabled time-tracking and installed the JIRA Charting Plugin.

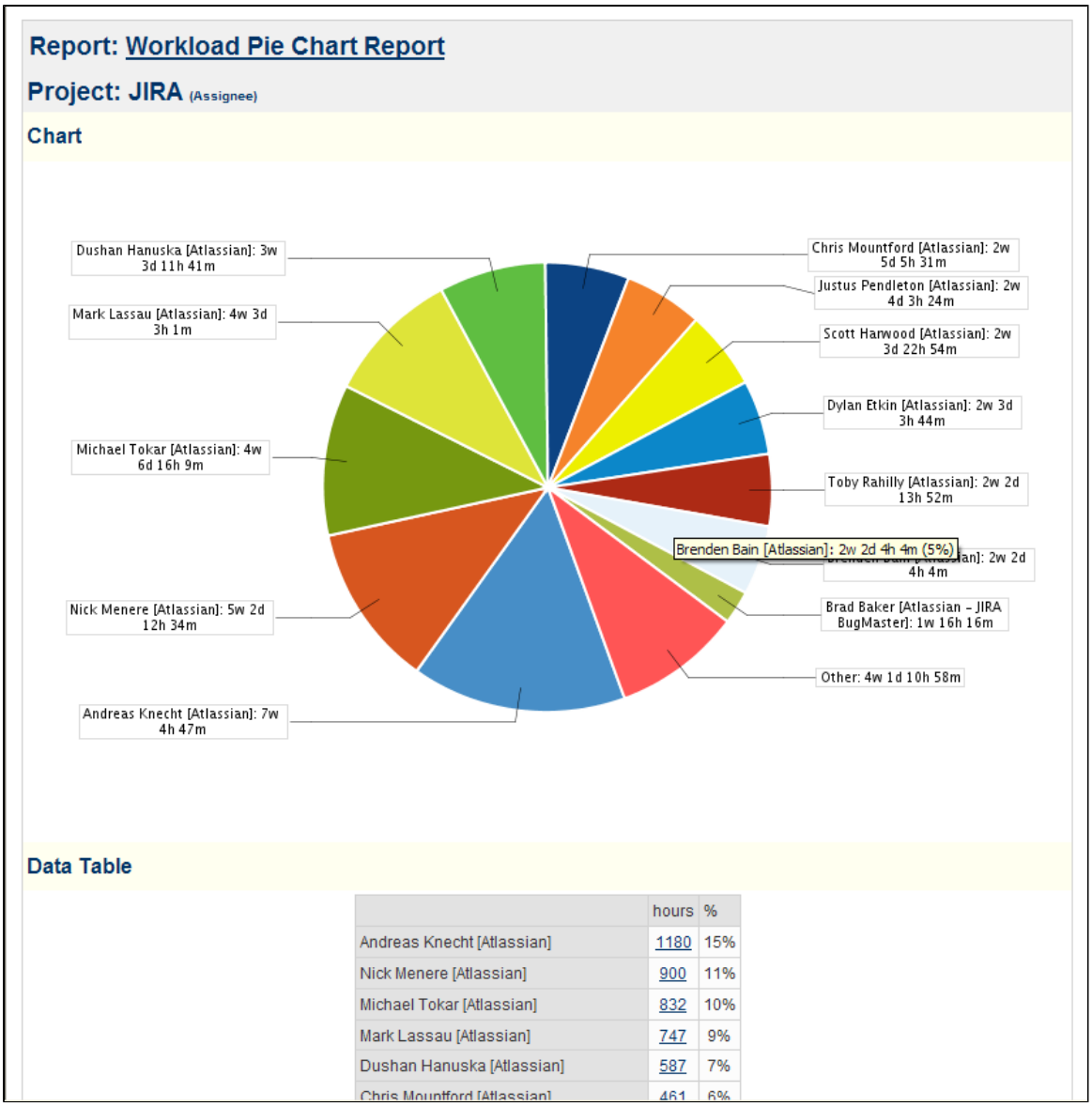
On this page:

- What does the 'Workload Pie Chart' report look like?
- Generating a 'Workload Pie Chart' report
- Configuring your Internet Explorer cache settings

What does the 'Workload Pie Chart' report look like?

The report generated will look something like this:

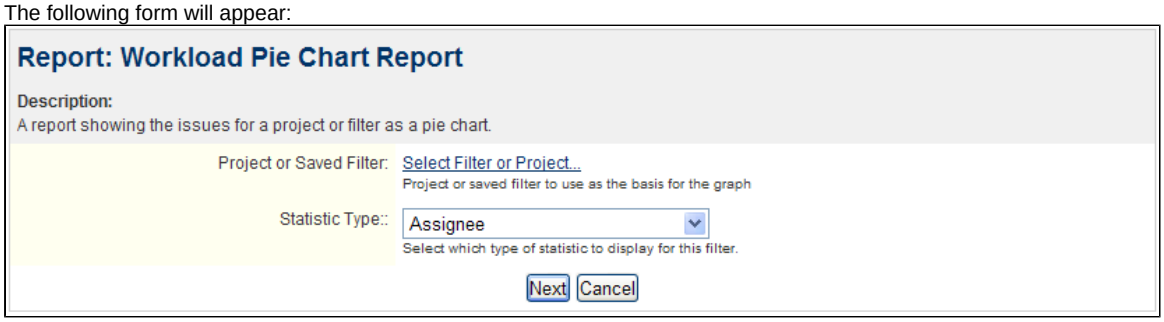
Screenshot: 'Workload Pie Chart' report



Generating a 'Workload Pie Chart' report

To generate a user workload report:

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   Tip: *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click '**Workload Pie Chart Report**' from the dropdown menu that displays. The following form will appear:

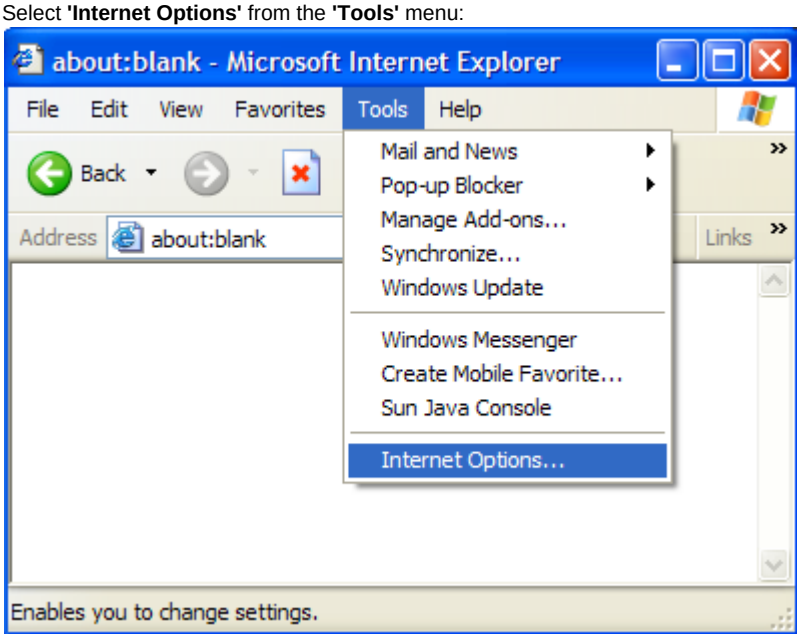


4. Select the project or issue filter for which you wish to generate a Workload Pie Chart report.
5. In the **'Statistic'** drop-down list, select the field on which the pie chart will be based (this will usually be Assignee).
6. Click '**Next**' to generate the report (see screenshot in previous section above).

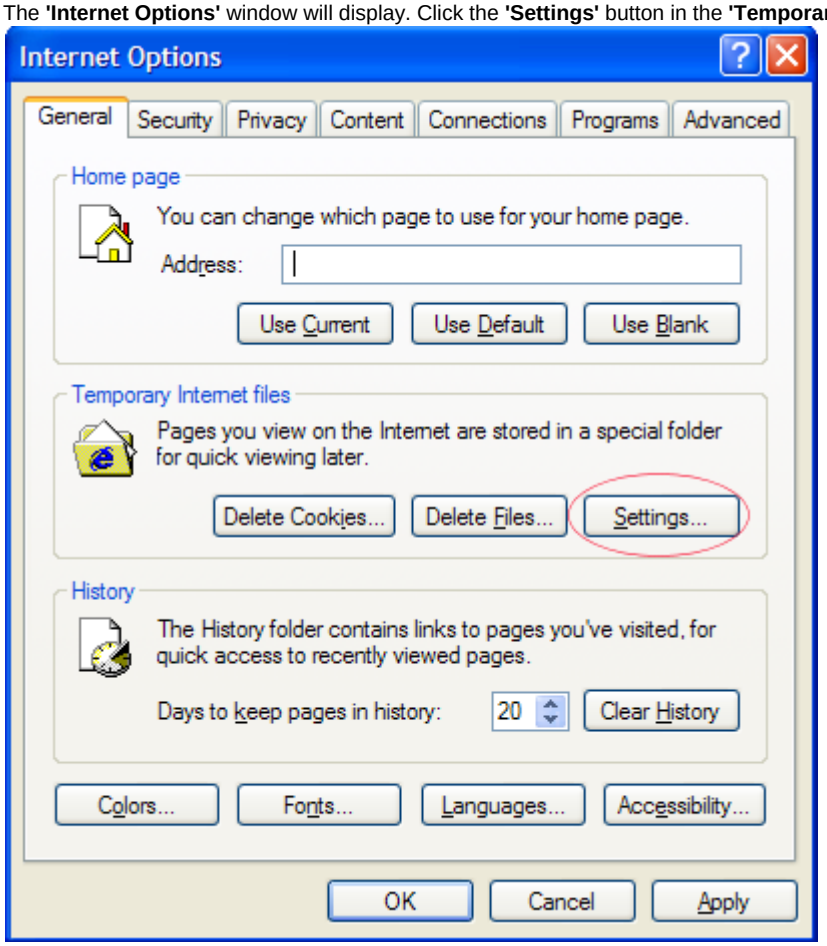
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

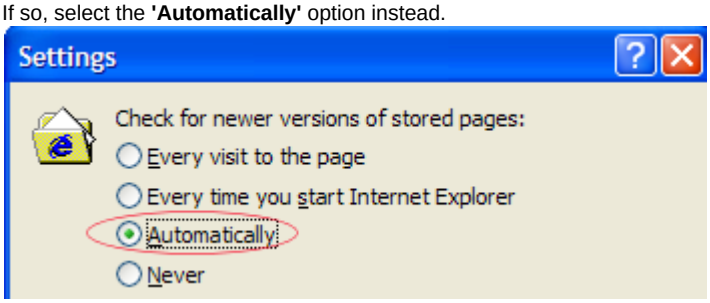
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



User Workload Report

The 'User Workload' report displays useful time tracking information on issues assigned to a particular user. It shows the number of unresolved issues assigned to the specified user, and the workload remaining, on a per-project basis.

> **Note:** this report is only available if time tracking has been enabled by your JIRA administrator.

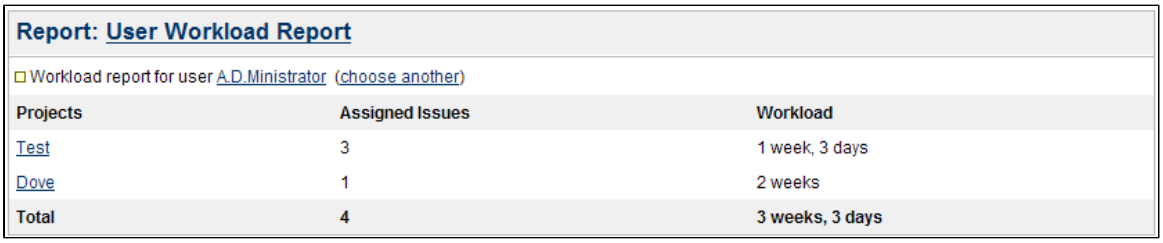
On this page:

- What does the 'User Workload' report look like?
- Generating a 'User Workload' report

What does the 'User Workload' report look like?

The report generated will look something like this:

Screenshot: 'User Workload' report

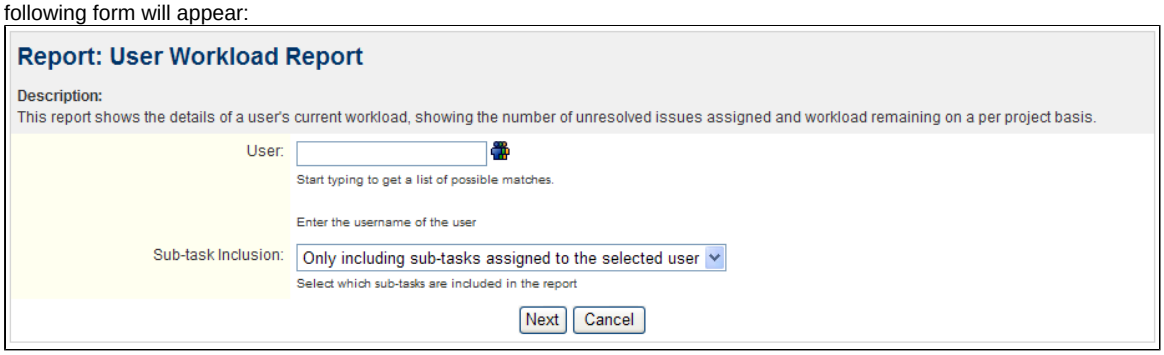


The table shows the number of unresolved issues assigned to the specified user, and the workload remaining, on a per-project basis. The last line in the table shows the total the number of issues and the total workload remaining for this user.

Generating a 'User Workload' report

To generate a user workload report:

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   ***Tip:** If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click '**User Workload Report**' from the dropdown menu that displays. The following form will appear:



4. Select or type the name of the user for whom you wish to generate a User Workload report.
5. In the **'Sub-task Inclusion'** drop-down list (**note:** this will only appear if sub-tasks are enabled), choose which sub-tasks will be included in the report, for all parent issues that belong to this version:
   - Select **'Only including sub-tasks assigned to the selected user'** to only include an issue's sub-tasks if the sub-tasks are assigned to the selected user; or
   - Select **'Also including unassigned sub-tasks'** to include an issue's sub-tasks regardless of whether the sub-tasks are assigned to the selected user or not.
6. Click '**Next**' to generate the report (See screenshot in previous section above).

Version Workload Report

The **Version Workload Report** displays useful time tracking information on the current workload for a specific version within a specific project. For the specified version, it shows a list of unresolved issues assigned to each user, each user's workload, and a summary of the total workload remaining for the version.

Note: this report is only available if time tracking has been enabled by your JIRA administrator.

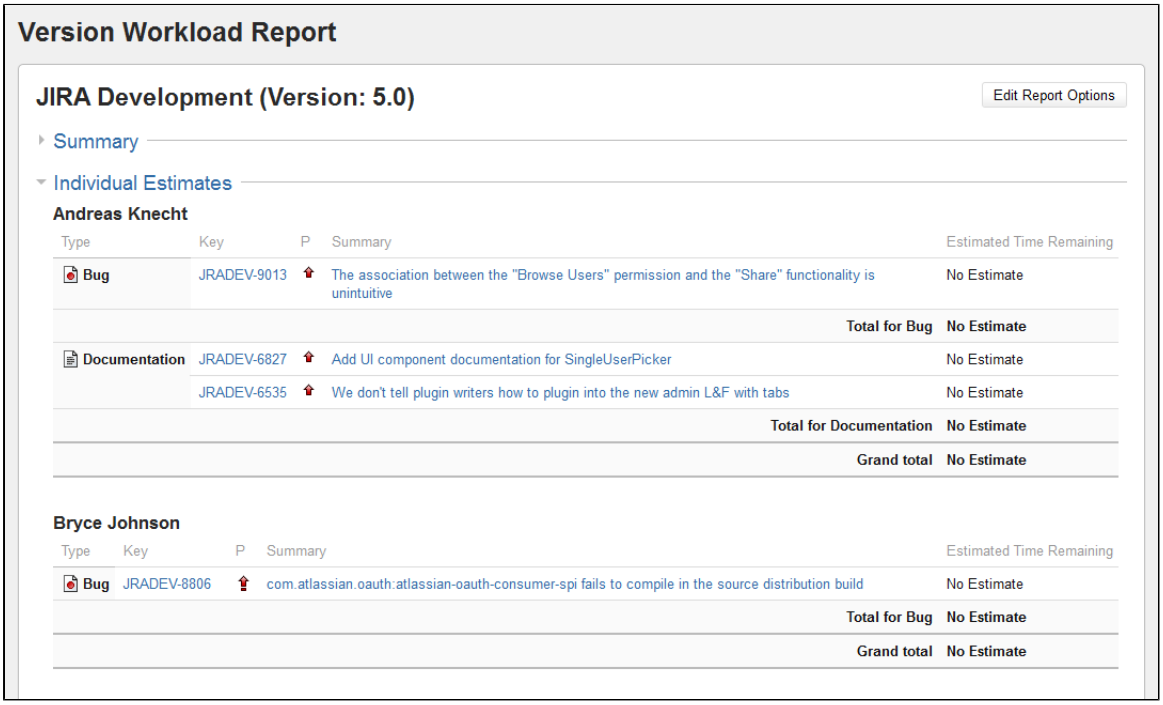
On this page:

- What does the Version Workload report look like?
- Generating a Version Workload Report

What does the Version Workload report look like?

The report generated will look something like this:

Screenshot: Version Workload Report



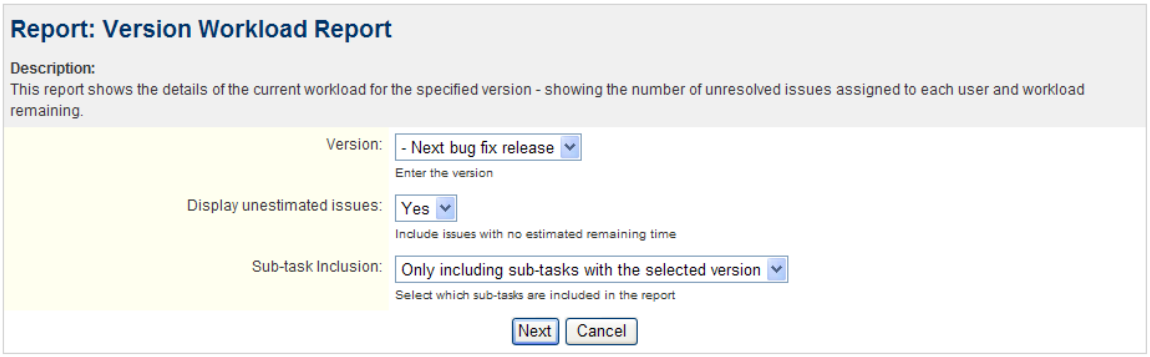
The first table summarises the workload for each user, broken down by issue type, for the specified version.

Following the summary, the report is composed of individual sections for each user --- with workload broken down by issue type. Each individual section begins with the workload total for the specific user. Finally, all unassigned issues (if any exist) are displayed.

Generating a Version Workload Report

To generate a version workload report:

1. On the top navigation bar, click the white triangle next to **Projects**. The projects dropdown will display.
   ***Tip:** If you click on the **Projects** link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click **View All Projects** — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **Reports** menu on the right of the page and click **Version Workload Report** from the dropdown menu that displays. The following form will appear:



4. In the **Version** drop-down list, select the version on which you wish to report. The report will include all issues that belong to this version, that is, all issues whose **Fix Version** is this version.
5. In the **Display unestimated issues** drop-down list, choose which issues will be included in the report:
    - Select **Yes** to show all unresolved issues, regardless of the value of their **Estimated Time Remaining** or **Original Estimate** fields.
    - Select **No** to exclude issues which are not time-tracked (i.e. do not have an **Original Estimate** specified).
6. In the **Sub-task Inclusion** drop-down list (**note:** this will only appear if sub-tasksare enabled), choose which sub-tasks will be included in the report, for all parent issues that belong to this version:
    - Select **Only include sub-tasks with the selected version** to only include an issue's sub-tasks if the sub-tasks belong to the same version as the issue; or
    - Select **Also include sub-tasks without a version set** to include an issue's sub-tasks if the sub-tasks belong to either the same version as the issue or to no version; or
    - Select **Include all sub-tasks** to include all of an issue's sub-tasks, regardless of whether the sub-tasks belong to the same version, some other version or no version.

Note: sub-tasks which belong to this version, but whose parent issues do not belong to this version, will always be included in the report.

Time Tracking Report

The Time Tracking Report displays useful time tracking information on issues for a particular version of a project. This report shows original and current time estimates for all the issues, and whether they are ahead of or behind the original schedule. (Note: this report is only available if time tracking has been enabled by your JIRA administrator).

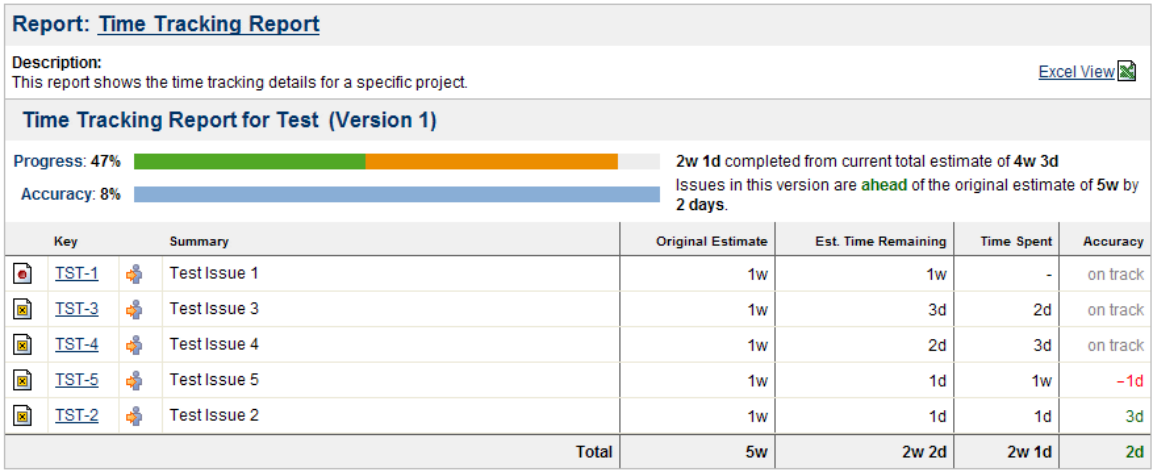
On this page:

- What does the 'Time Tracking' report look like?
- Generating a 'Time Tracking' report
- See Also

What does the 'Time Tracking' report look like?

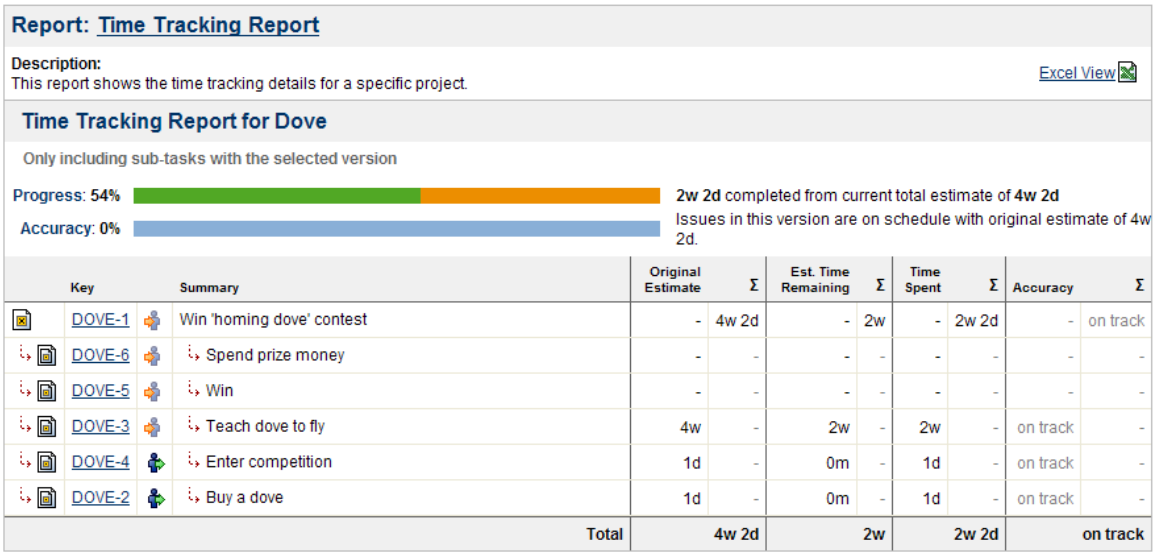
The report generated will look something like this:

Screenshot: 'Time Tracking' report



Key	Summary	Original Estimate	Est. Time Remaining	Time Spent	Accuracy
TST-1	Test Issue 1	1w	1w	-	on track
TST-3	Test Issue 3	1w	3d	2d	on track
TST-4	Test Issue 4	1w	2d	3d	on track
TST-5	Test Issue 5	1w	1d	1w	-1d
TST-2	Test Issue 2	1w	1d	1d	3d
	Total	5w	2w 2d	2w 1d	2d

Or, if sub-tasks are enabled, the report will look something like this:



Report: Time Tracking Report

Description:
This report shows the time tracking details for a specific project.

Excel View

Time Tracking Report for Dove

Only including sub-tasks with the selected version

Progress: 54% — **2w 2d** completed from current total estimate of **4w 2d**

Accuracy: 0% — Issues in this version are on schedule with original estimate of 4w 2d.

Key	Summary	Original Estimate	Σ	Est. Time Remaining	Σ	Time Spent	Σ	Accuracy	Σ
DOVE-1	Win 'homing dove' contest	-	4w 2d	-	2w	-	2w 2d	-	on track
DOVE-6	Spend prize money	-	-	-	-	-	-	-	-
DOVE-5	Win	-	-	-	-	-	-	-	-
DOVE-3	Teach dove to fly	4w	-	2w	-	2w	-	on track	-
DOVE-4	Enter competition	1d	-	0m	-	1d	-	on track	-
DOVE-2	Buy a dove	1d	-	0m	-	1d	-	on track	-
	Total		4w 2d		2w		2w 2d		on track

The table in the report shows the issues within the version:

- There are four time tracking fields as follows:
  - **Original Estimate** - The original estimate of the total amount of time it would take to complete this issue.
  - **Estimated Time Remaining** - The current estimate of the remaining amount of time it would take to complete this issue.
  - **Time Spent** - The amount of time spent on the issue. This is the aggregate amount of time which has been logged against this issue.
  - **Accuracy**- The accuracy of the original estimate compared to the current estimate for the issue. It is the difference between the sum of the **Time Spent** and **Estimated Time Remaining** fields, and the **Original Estimate** field.
- If sub-tasks are enabled, the *'*column at the right of the field shows the aggregate time tracking information for each 'parent' issue (i.e. the sum of the issue's own values plus those of its sub-tasks).
- The last line of the table shows the aggregate time tracking information for the whole version.

The report also includes two bar-graphs (above the table) which represent the aggregate time tracking information for the version:

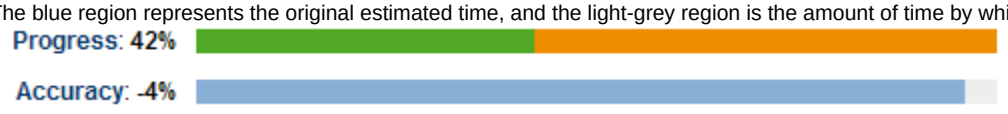
- The first bar-graph **('Progress')** shows the percentage of completed issues (green) and incomplete issues (orange) in this version:

- The second bar-graph **('Accuracy'** -blue) shows the accuracy of the original estimates.

The length of the **Accuracy** bar compared to the **Progress** bar indicates whether the issues in this version are ahead of or behind schedule. There are three cases:

1. *The issues are on schedule with the original estimate.* The **Accuracy** bar is completely blue and is the same length as the **Progress** bar above it.

2. *The issues are behind the original estimate (i.e. will take longer than originally estimated).* The **Progress** graph is longer than the **Accuracy**graph. The blue region represents the original estimated time, and the light-grey region is the amount of time by which issues are behind.

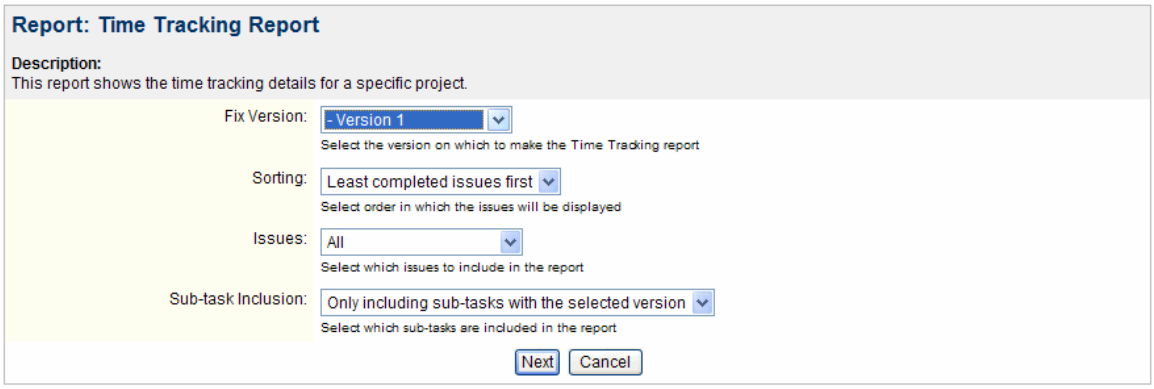


3. *The issues are ahead of the original estimate (i.e. will take less time than originally estimated).* The **Accuracy** graph is longer than the **Progress**graph. The blue bar represents the original estimated time, and the light-grey region represents the amount of time by which the original estimates were overestimated.

Generating a 'Time Tracking' report

To generate a Time Tracking Report:

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   Tip: *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click '**Time Tracking Report**' from the dropdown menu that displays. The following form will appear:



4. In the **'Version'** drop-down list, select the version on which you wish to report. The report will include all issues that belong to this version, that is, all issues whose **'Fix Version'** is this version.
5. In the **'Sorting'** drop-down list, choose how the issues in the report will be sorted:
    - Select **'Least completed issues first'** to show issues with the highest **'Estimated Time Remaining** first; or
    - Select **'Most completed issues first'** to show issues with the lowest **'Estimated Time Remaining** first.
6. In the **'Issues'** drop-down list, choose which issues will be included in the report:
    - Select **'All'** to include all issues assigned to this version; or
    - Select **'Incomplete issues only'** to exclude issues which are either completed (i.e. have an **'Estimated Time Remaining'** of zero), or are not time-tracked (i.e. do not have an **'Original Estimate').** Note that issue status does not affect which issues are displayed.
7. In the **'Sub-task Inclusion'** drop-down list *(note: this will only appear if sub-tasks are enabled),* choose which sub-tasks will be included in the report, for all parent issues that belong to this version:
    - Select **'Only include sub-tasks with the selected version'** to only include an issue's sub-tasks if the sub-tasks belong to the same version as the issue; or
    - Select **'Also include sub-tasks without a version set'** to include an issue's sub-tasks if the sub-tasks belong to either the same version as the issue or to no version; or
    - Select **'Include all sub-tasks'** to include all of an issue's sub-tasks, regardless of whether the sub-tasks belong to the same version, some other version or no version.Note: sub-tasks which belong to this version, but whose parent issues do *not* belong to this version, will always be included in the report.

See Also

- In addition to the built-in JIRA reports, other reports (e.g. Gantt Chart Report, Timesheet Report) are available for download from the Atlassian Plugin Exchange and the JIRA Extensions site. JIRA administrators can also create new reports with the plugin API — see How to create a JIRA Report.
- You may also find the Dashboard Gadgets useful, e.g. the Two-Dimensional Filter Statistics Gadget displays statistical data based on a specified issue filter, in a configurable table format.

Single Level Group By Report

The **Single Level Group By** report displays issues returned from a specified issue filter of your choice, grouped by a specified field. For example, an issue filter can be created to retrieve all open issues for a particular version of a particular project. The **Single Level Group By** report can then be used to display these issues grouped by a specified field (e.g. Assignee).

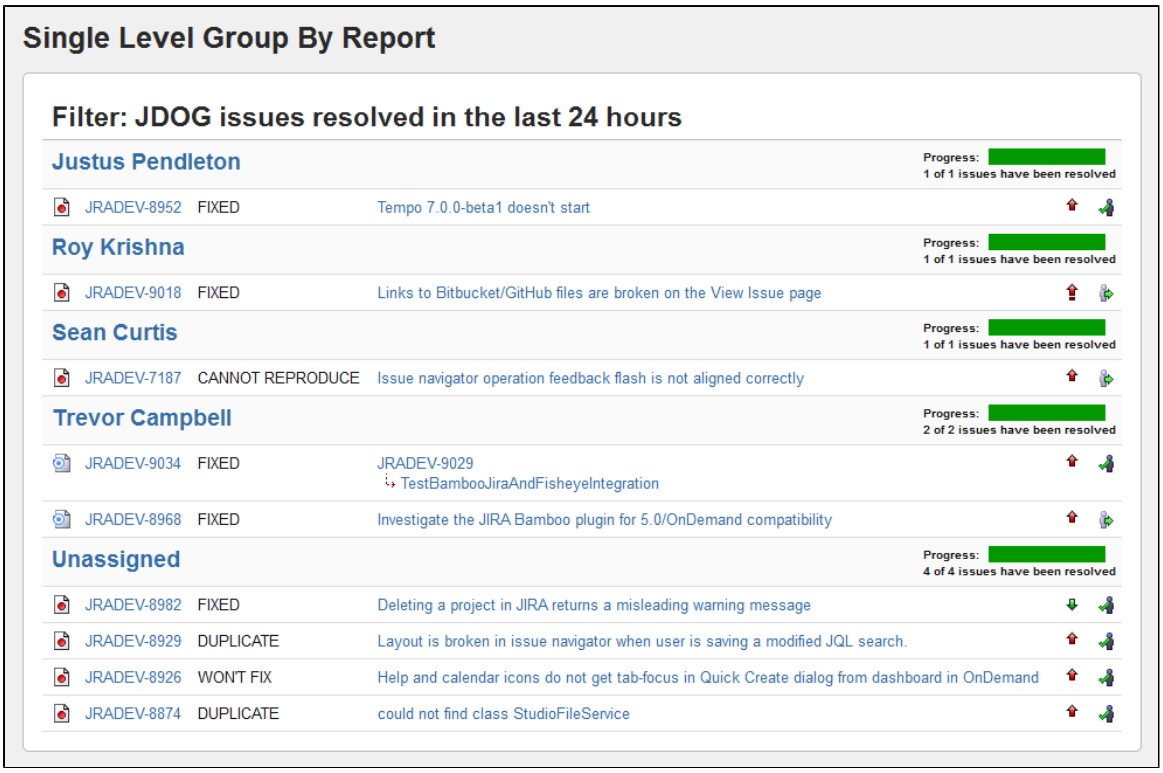
On this page:

- What does the Single Level Group By report look like?
- Generating a Single Level Group By report

What does the Single Level Group By report look like?

The report generated will look something like this:

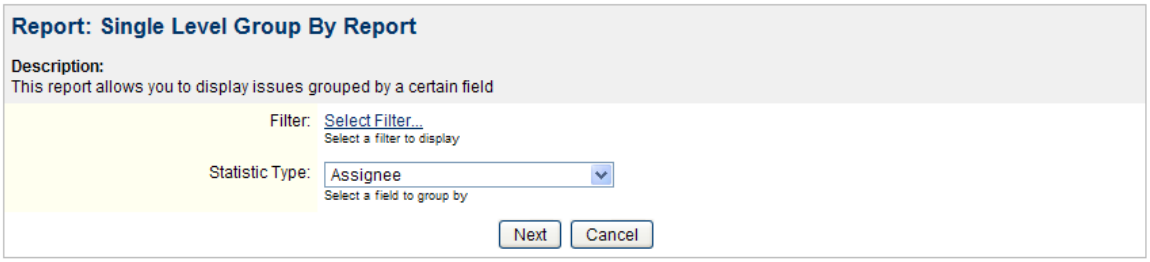
Screenshot: Single Level Group By Report



The report displays the issues returned by the specified filter, grouped by the specified field.

Generating a Single Level Group By report

1. On the top navigation bar, click the white triangle next to **Projects**. The projects dropdown will display.
   ***Tip:** If you click on the **Projects** link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click **View All Projects** — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **Reports** menu on the right of the page and click **Single Level Group By Report** from the dropdown menu that displays. The following form will appear:



4. Click **Select Filter...**
5. The popup **Filter Picker** will appear. Select the issue filter. You will then be returned to the form.
6. In the **Statistic Type** field, select the field by which the report will group the issues returned from your chosen issue filter.

Related topics:

The following gadgets can be added to your dashboard to display similar information to the 'Single Level Group By' report:

- Filter Results Gadget
- Issue Statistics Gadget
- Two-Dimensional Filter Statistics Gadget

Created vs Resolved Issues Report

The 'Created vs Resolved Issues' report is a difference chart showing the number of issues created vs number of issues resolved over a given period of time. The report is based on your choice of project or issue filter, and the chart can either be cumulative or not.

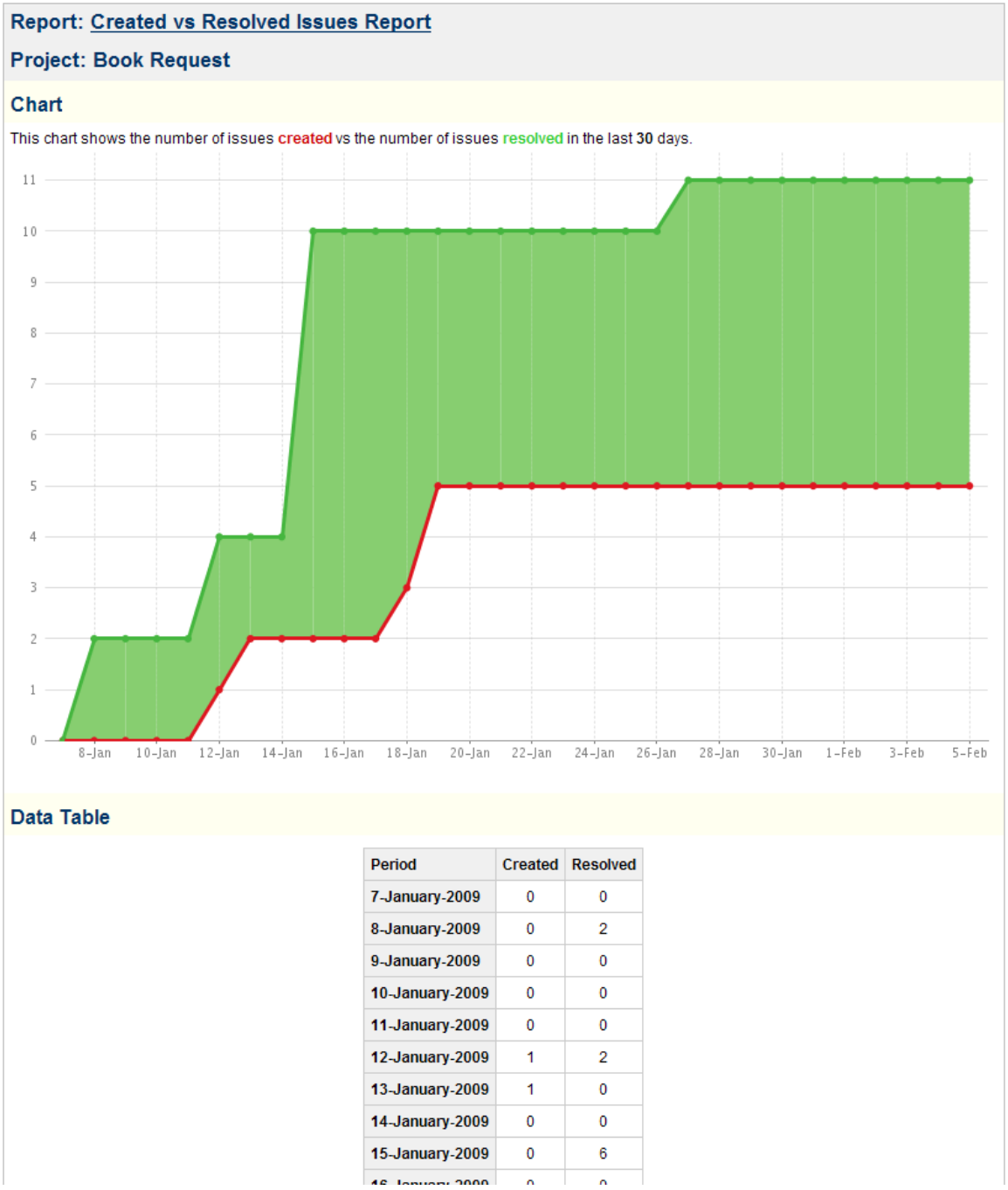
On this page:

- What does the 'Created vs Resolved Issues' report look like?
- Generating a 'Created vs Resolved Issues' report
- Configuring your Internet Explorer cache settings

What does the 'Created vs Resolved Issues' report look like?

The report generated will look something like this:

Screenshot: 'Created vs Resolved Issues' report

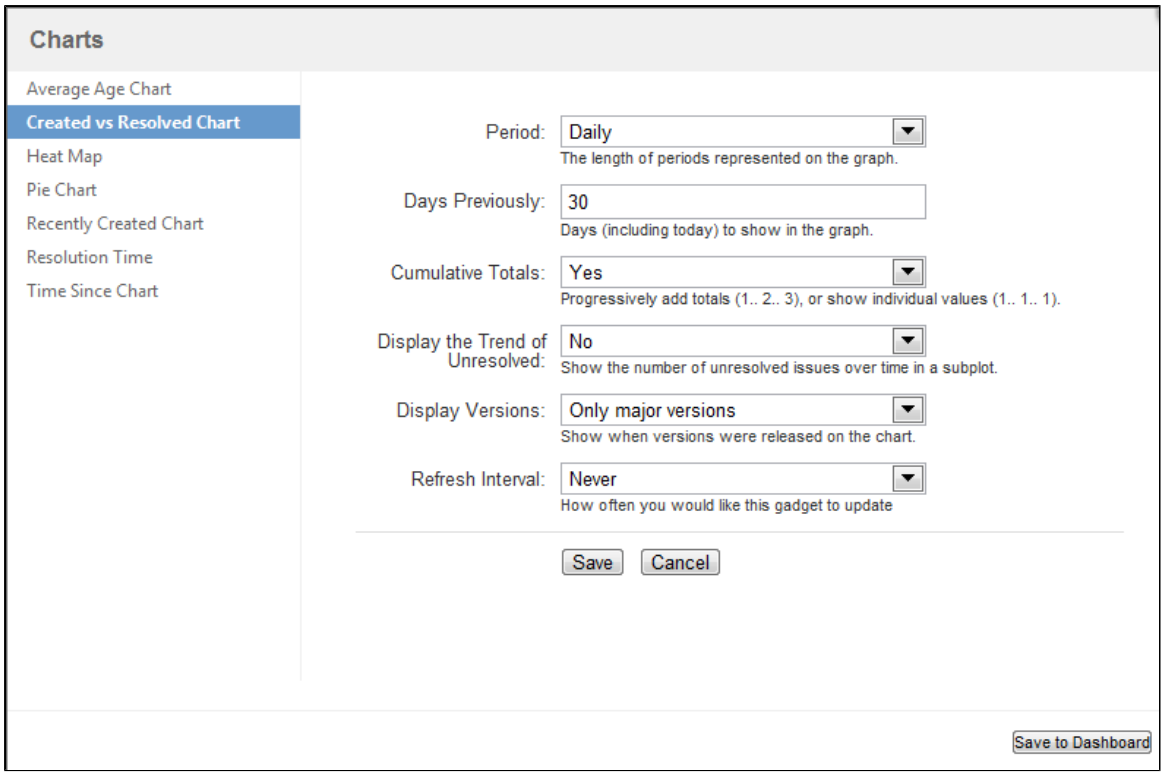


Period	Created	Resolved
7-January-2009	0	0
8-January-2009	0	2
9-January-2009	0	0
10-January-2009	0	0
11-January-2009	0	0
12-January-2009	1	2
13-January-2009	1	0
14-January-2009	0	0
15-January-2009	0	6

Areas in red show periods where more issues were created than resolved. Areas in green show periods where more were resolved than created.

Generating a 'Created vs Resolved Issues' report

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   ✅ ***Tip:*** *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click **'Created vs Resolved Issues Report'** from the dropdown menu that displays. The following form will appear:

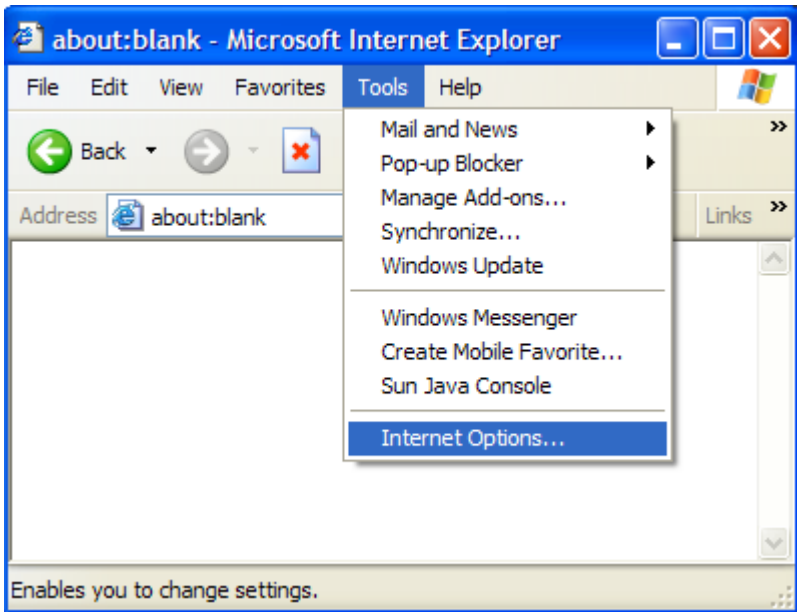


4. Click **'Select Filter or Project'.**
5. The popup **Filter or Project Picker** will appear. Select the project, or issue filter, in which you are interested. You will then be returned to the form.
6. In the **'Period'** field, select the timeframe on which the report will be based:
    - **'Hourly'**
    - **'Daily'**
    - **'Weekly'**
    - **'Quarterly'**
    - **'Yearly'**
7. In the **'Days Previously'** field, enter the number of days' worth of data (counting backwards from today) to be included in the report.
8. In the **'Cumulative Totals?'** field, choose either:
    - **'Yes'** to progressively add data to the preceding column; or
    - **'No'** to show just a single value in each column.
9. In the **'Display the trend of Unreserved'** field, choose either:
    - **'Yes'** to show the number of unresolved issues over time in a subplot; or
    - **'No'** otherwise.
10. In the **'Display Versions?'** field, choose either:
    - **'All versions'** to show version release dates on the chart, for all released versions; or
    - **'Only major versions'** to show version release dates on the chart, for released versions that are named 'x.x' only; or
    - **'None'** to not show version release dates on the chart.

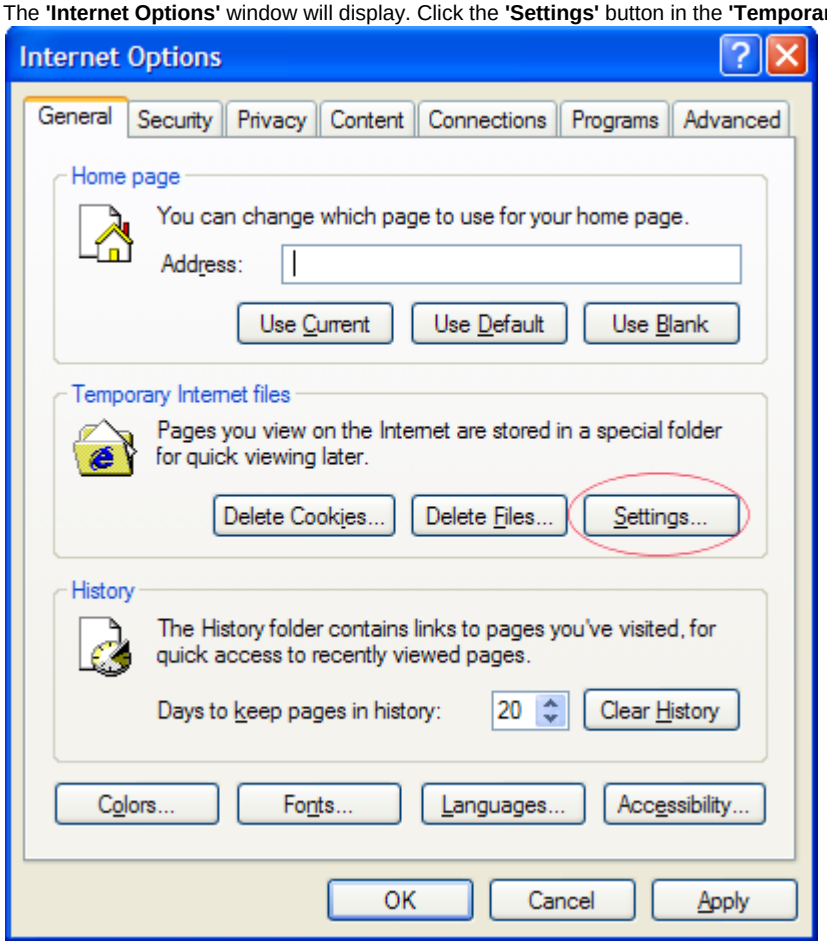
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

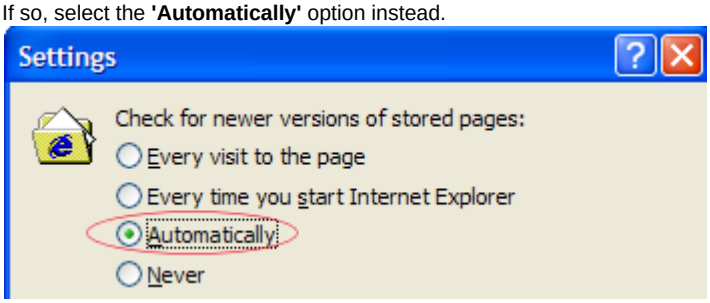
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Resolution Time Report

The 'Resolution Time' report is a bar chart showing the average time taken to resolve issues. This is useful to show you the trends in resolution time. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).

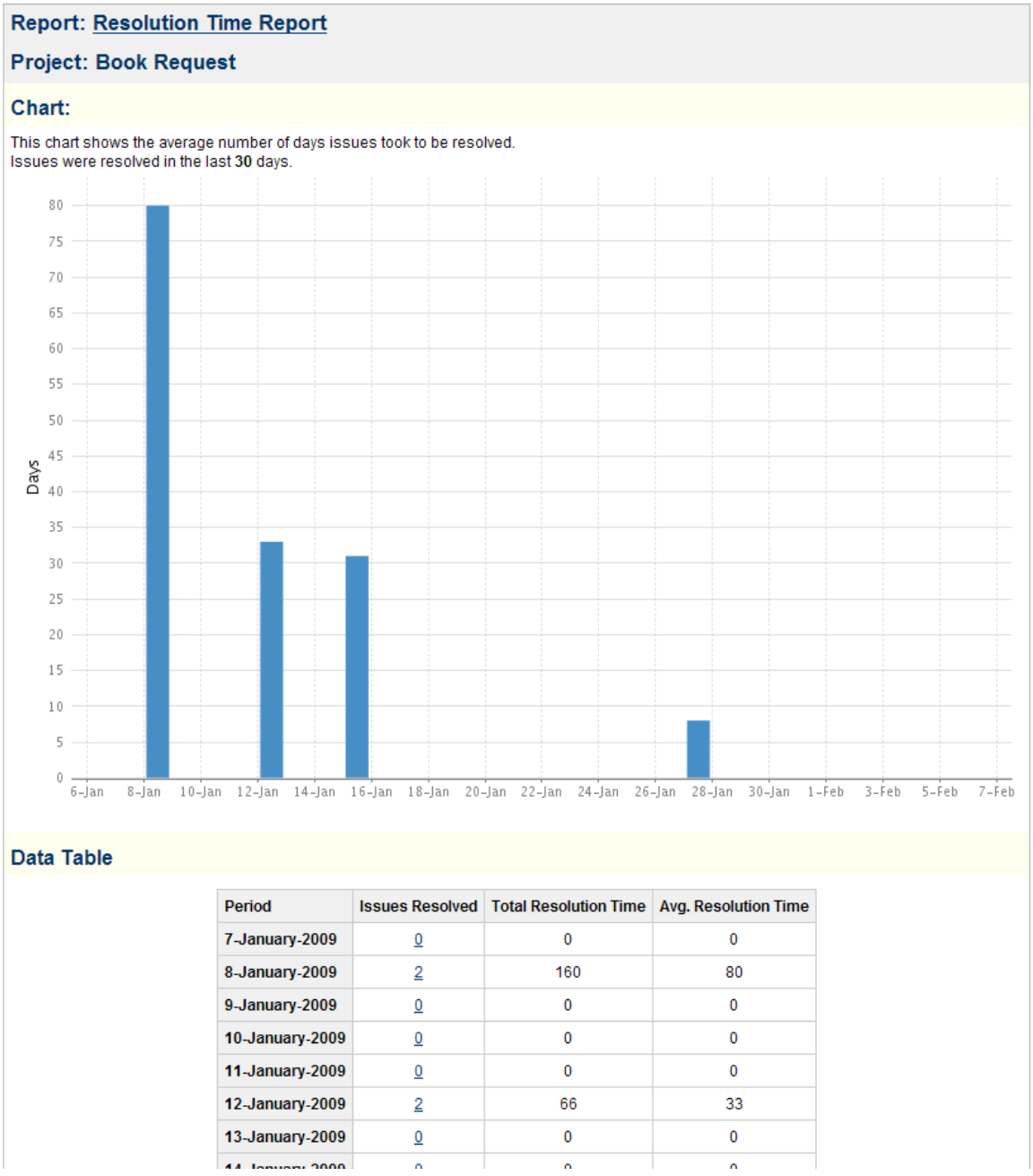
On this page:

- What does the 'Resolution Time' report look like?
- Generating a 'Resolution Time' report
- Configuring your Internet Explorer cache settings

What does the 'Resolution Time' report look like?

The report generated will look something like this:

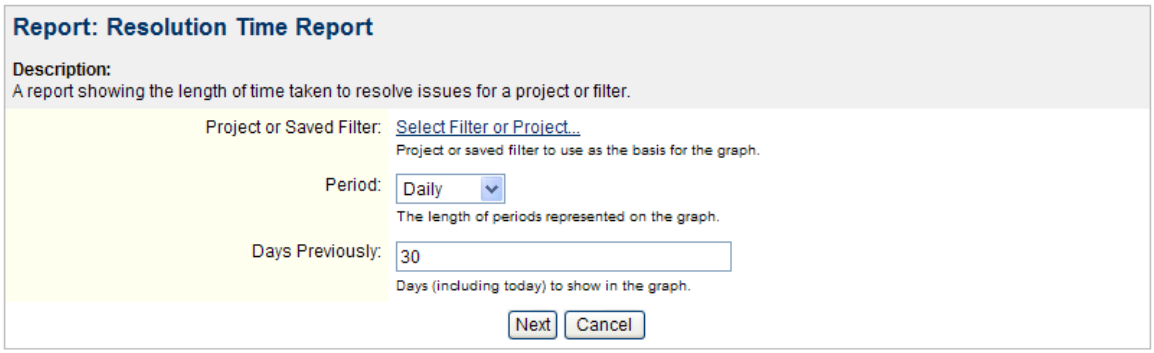
Screenshot 'Resolution Time' report:



Period	Issues Resolved	Total Resolution Time	Avg. Resolution Time
7-January-2009	0	0	0
8-January-2009	2	160	80
9-January-2009	0	0	0
10-January-2009	0	0	0
11-January-2009	0	0	0
12-January-2009	2	66	33
13-January-2009	0	0	0

Generating a 'Resolution Time' report

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   ✅ ***Tip:*** *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click **'Resolution Time Report'** from the dropdown menu that displays. The following form will appear:

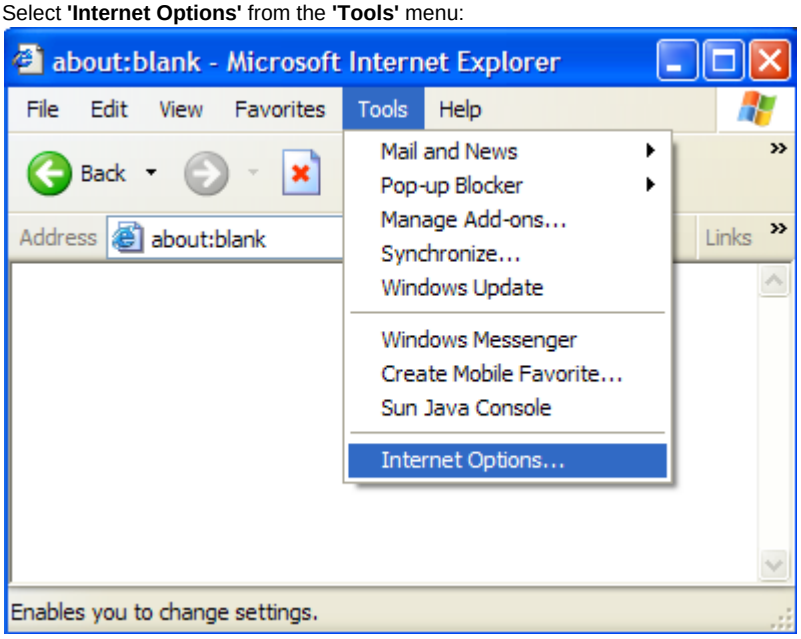


4. Click **'Select Filter or Project'.**
5. The popup **Filter or Project Picker** will appear. Select the project, or issue filter, in which you are interested. You will then be returned to the form.
6. In the **'Period'** field, select the timeframe on which the report will be based:
    - **'Hourly'**
    - **'Daily'**
    - **'Weekly'**
    - **'Quarterly'**
    - **'Yearly'**
7. In the **'Days Previously'** field, enter the number of days' worth of data (counting backwards from today) to be included in the report.

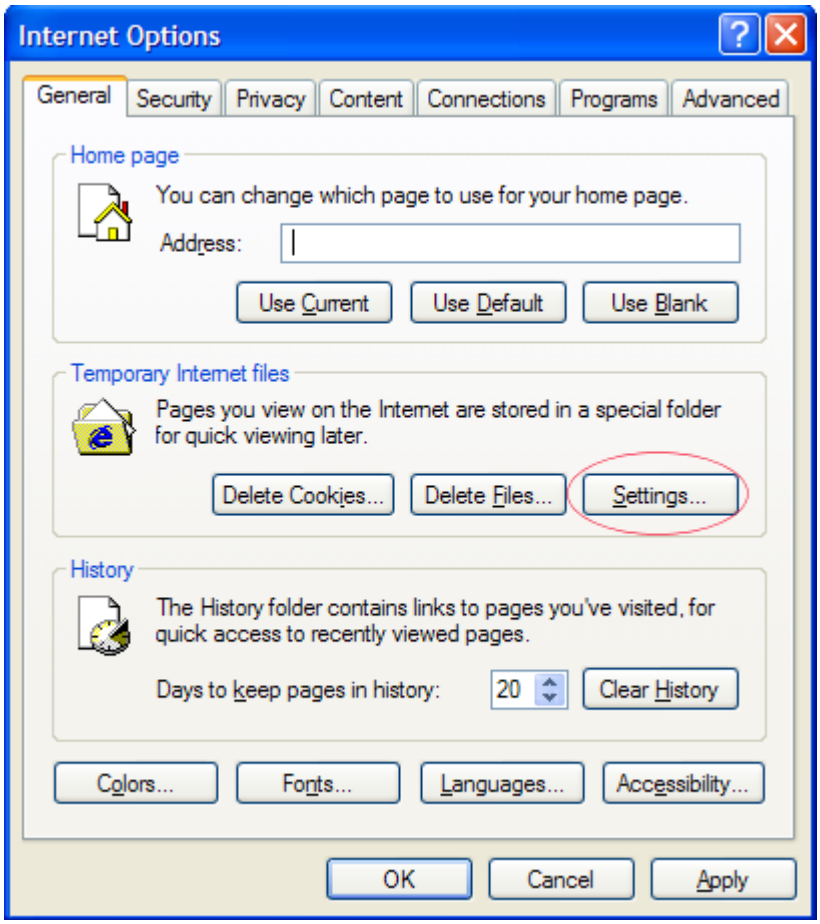
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

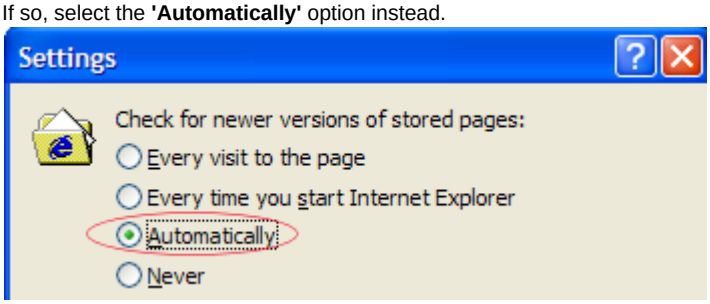
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Pie Chart Report

The 'Pie Chart' report displays issues returned from a specified project or issue filter, grouped by a specified field. For example, an issue filter can be created to retrieve all open issues for a particular version of a particular project. The 'Pie Chart' report can then be used to display these issues grouped by a specified field (e.g. Assignee).

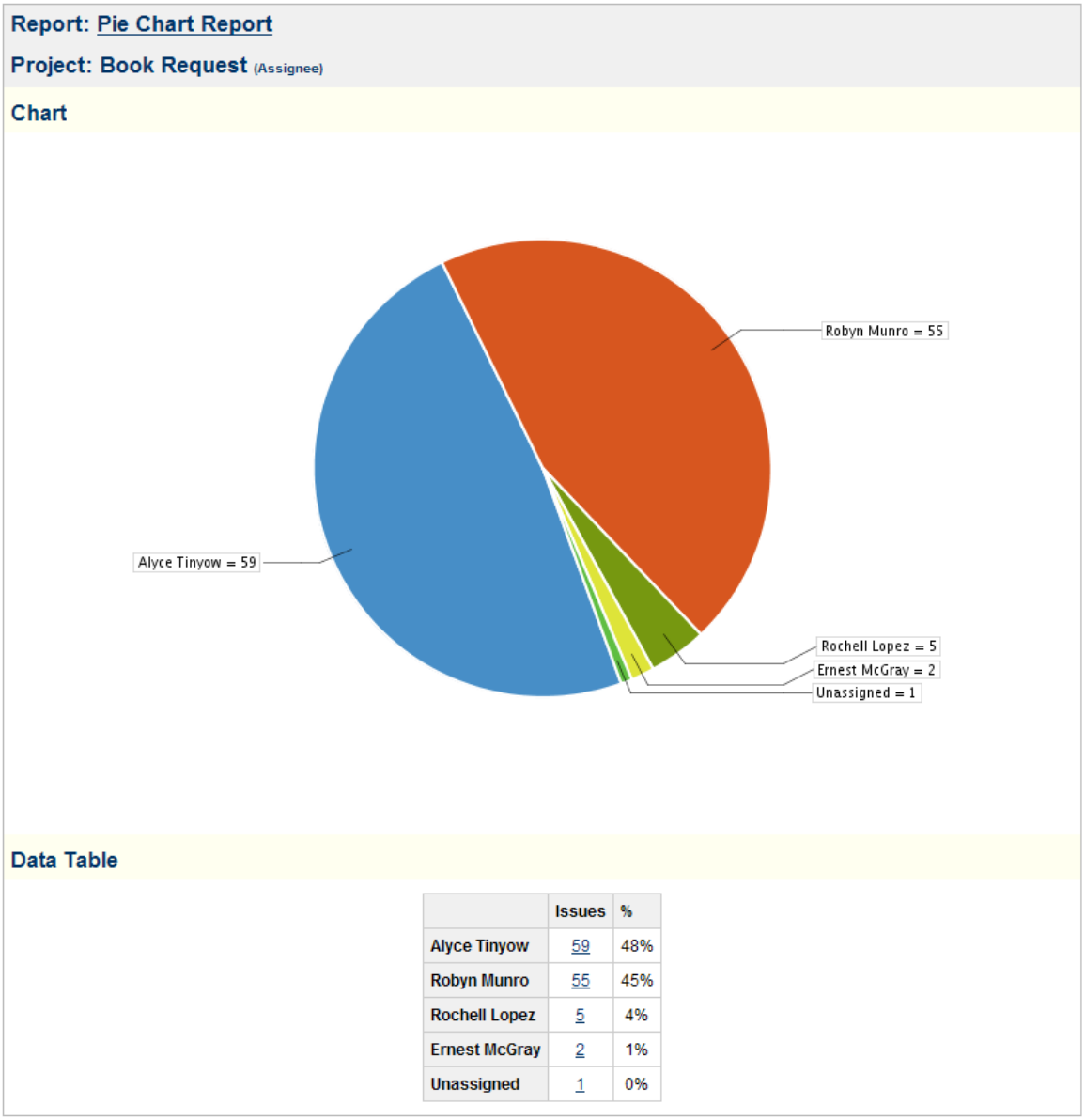
On this page:

- What does the 'Pie Chart' report look like?
- Generating a 'Pie Chart' report
- Configuring your Internet Explorer cache settings

What does the 'Pie Chart' report look like?

The report generated will look something like this:

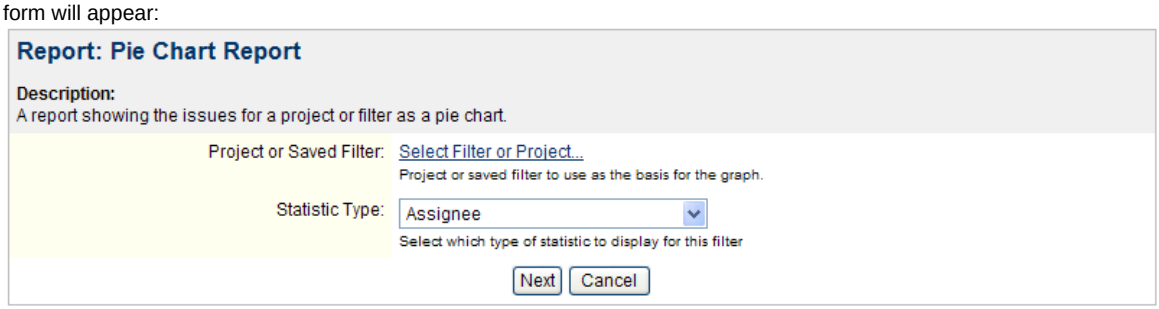
Screenshot: 'Pie Chart' report



	Issues	%
Alyce Tinyow	59	48%
Robyn Munro	55	45%
Rochell Lopez	5	4%
Ernest McGray	2	1%
Unassigned	1	0%

Generating a 'Pie Chart' report

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   Tip: *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click **'Pie Chart Report'** from the dropdown menu that displays. The following form will appear:

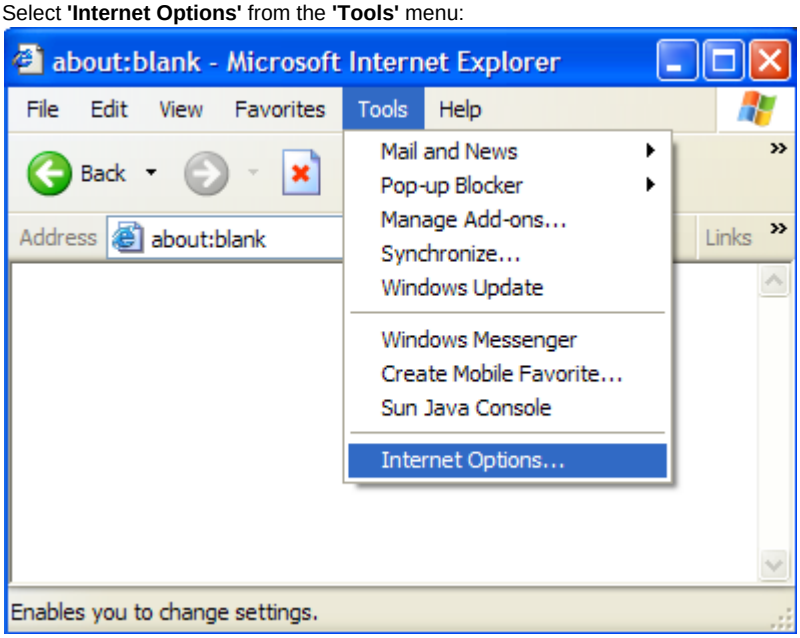


4. Click **'Select Filter or Project'.**
5. The popup **Filter or Project Picker** will appear. Select the project, or issue filter, in which you are interested. You will then be returned to the form.
6. In the **'Statistic Type'** field, select the field on which the pie chart will be based. (Note that you can choose only fields which have finite values).

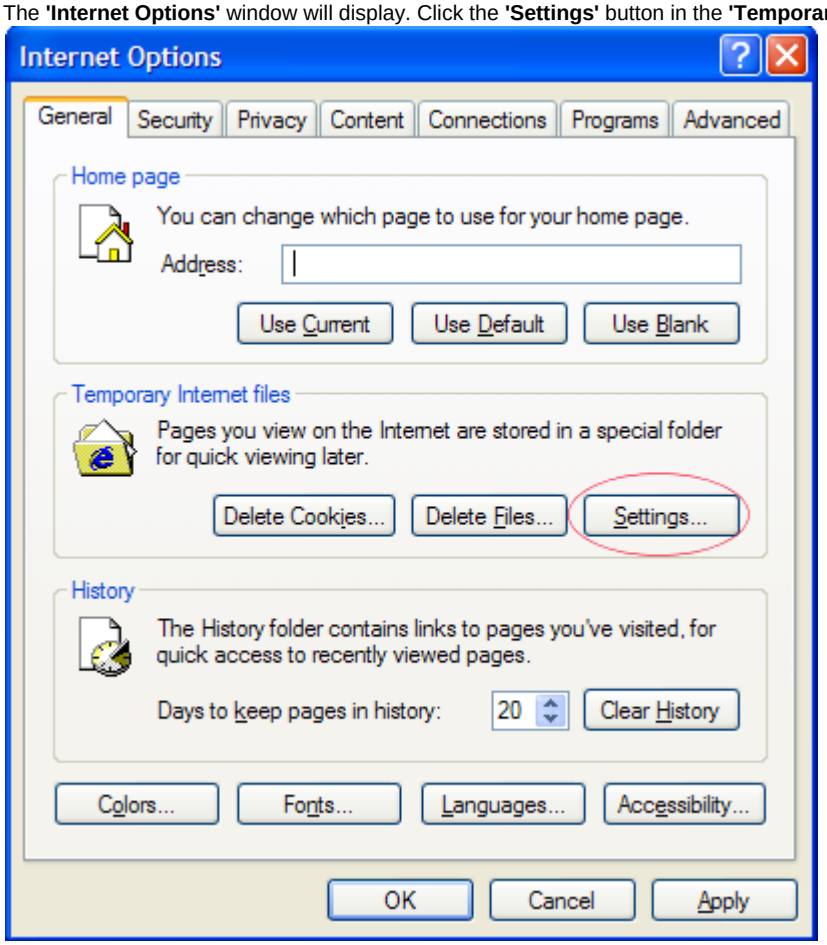
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

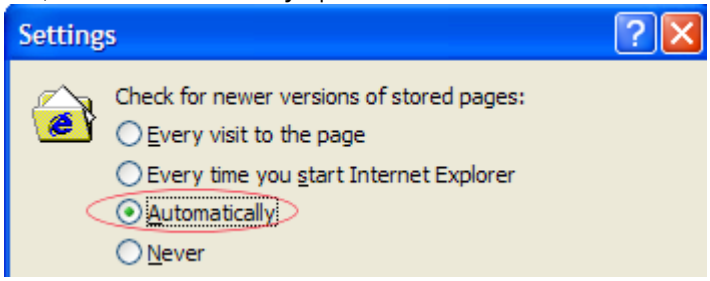
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Average Age Report

The 'Average Age' report is a bar chart showing the average age (in days) of unresolved issues at given points in time. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).

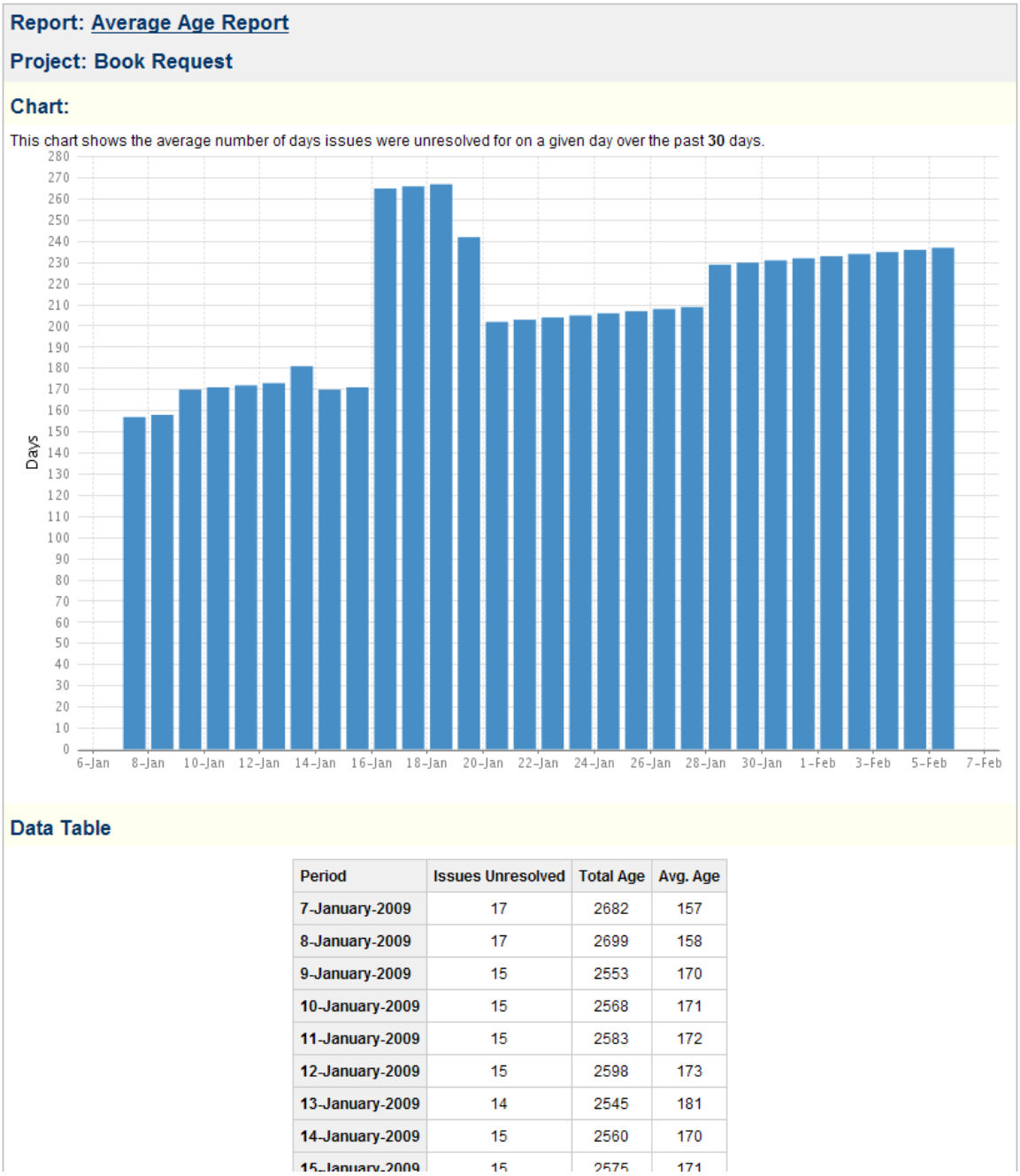
On this page:

- What does the 'Average Age' report look like?
- Generating an 'Average Age' report
- Configuring your Internet Explorer cache settings

What does the 'Average Age' report look like?

The report generated will look something like this:

Screenshot: 'Average Age' report

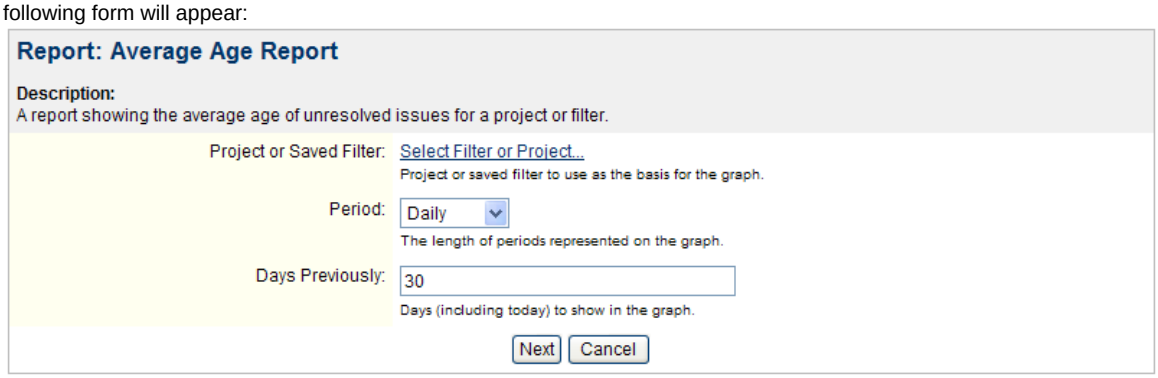


Period	Issues Unresolved	Total Age	Avg. Age
7-January-2009	17	2682	157
8-January-2009	17	2699	158
9-January-2009	15	2553	170
10-January-2009	15	2568	171
11-January-2009	15	2583	172
12-January-2009	15	2598	173
13-January-2009	14	2545	181
14-January-2009	15	2560	170
15-January-2009	15	2575	171

Generating an 'Average Age' report

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   ✅ ***Tip:*** *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click **'Average Age Report'** from the dropdown menu that displays. The

following form will appear:

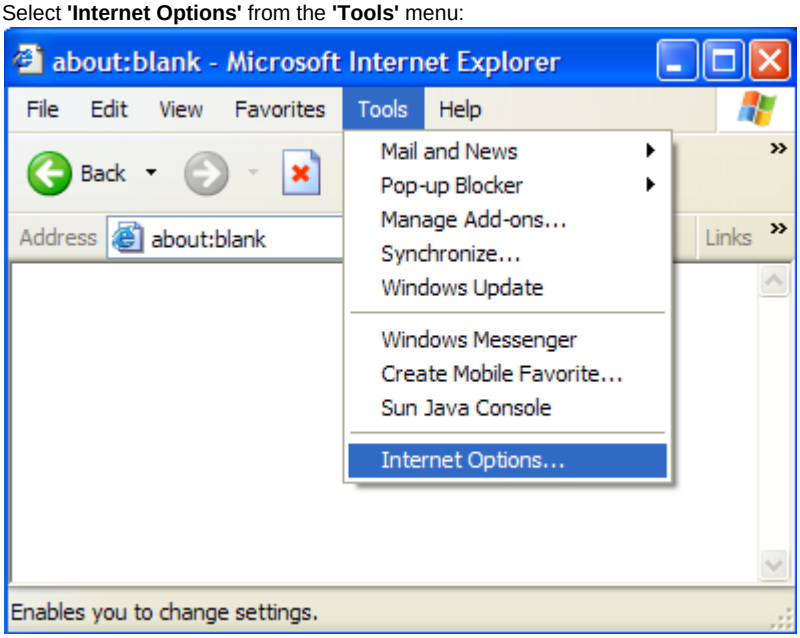


4. Click **'Select Filter or Project'.**
5. The popup **Filter or Project Picker** will appear. Select the project, or issue filter, in which you are interested. You will then be returned to the form.
6. In the **'Period'** field, select the timeframe on which the report will be based:
    - **'Hourly'**
    - **'Daily'**
    - **'Weekly'**
    - **'Quarterly'**
    - **'Yearly'**
7. In the **'Days Previously'** field, enter the number of days' worth of data (counting backwards from today) to be included in the report.

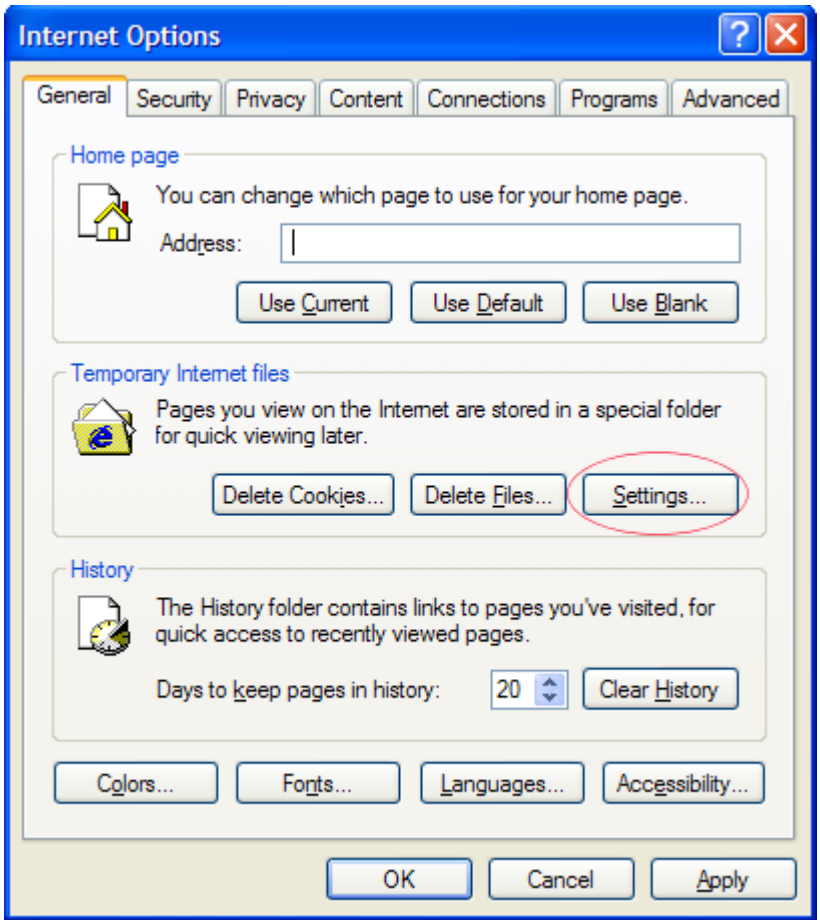
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

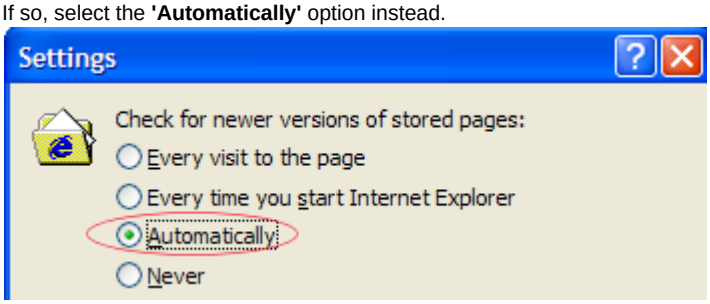
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Recently Created Issues Report

The 'Recently Created Issues' report is a bar chart showing the rate at which issues are being created, as well as how many of those created issues are resolved. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).

On this page:

- What does the 'Recently Created Issues' report look like?
- Generating a 'Recently Created Issues' report
- Configuring your Internet Explorer cache settings

What does the 'Recently Created Issues' report look like?

The report generated will look something like this:

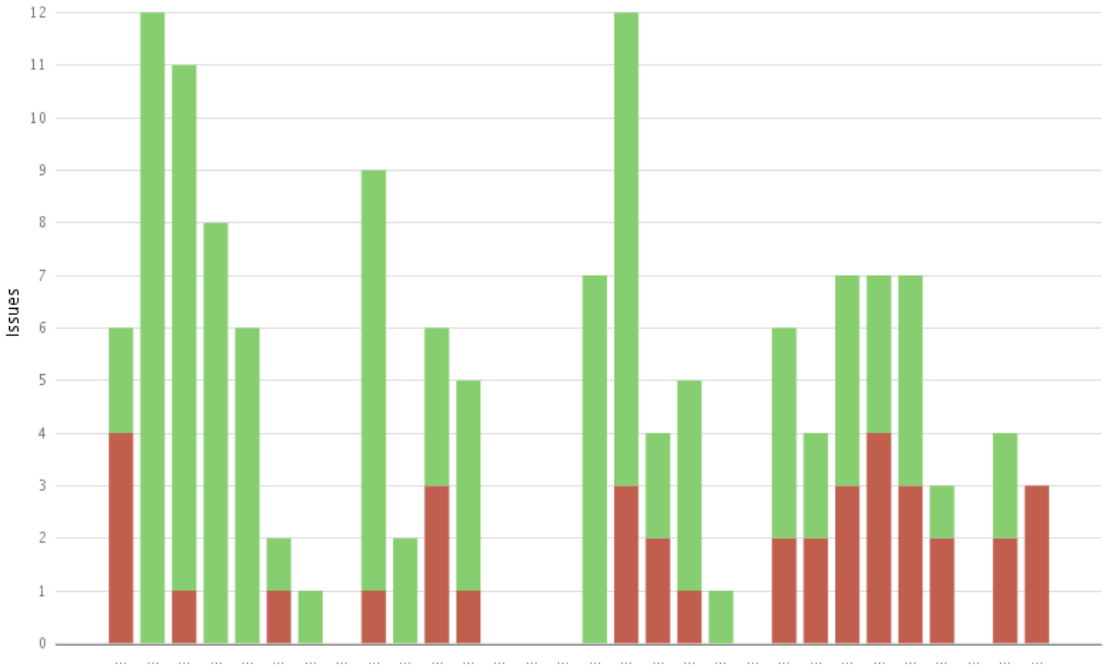
Screenshot: 'Recently Created Issues' report

Report: Recently Created Issues Report

Project: SysAdmin

Chart

This chart shows the issues created in the last **30** days



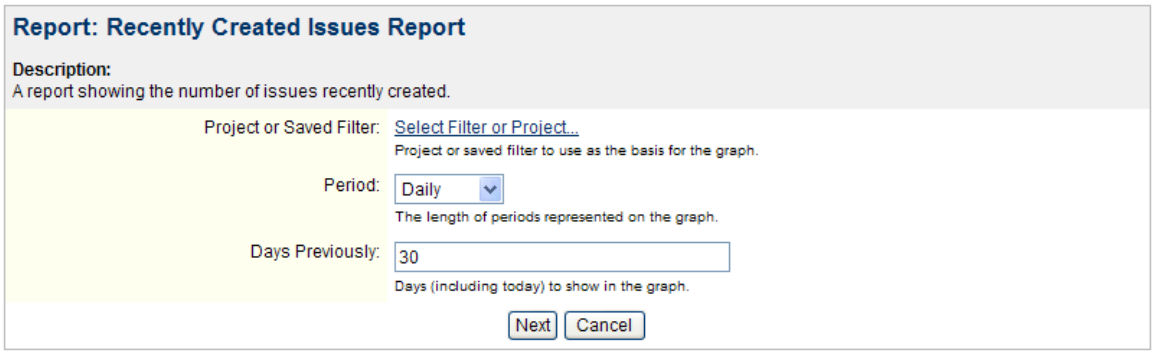
Data Table

Period	Created Issues (Unresolved)	Created Issues (Resolved)
11-January-2009	4	2
12-January-2009	0	12
13-January-2009	1	10
14-January-2009	0	8
15-January-2009	0	6
16-January-2009	1	1
17-January-2009	0	1
18-January-2009	0	0
19-January-2009	1	8
20-January-2009	0	2

The green portion of the bar shows the created issues which are resolved. The red portion shows created but as yet unresolved issues.

Generating a 'Recently Created Issues' report

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   ✅ ***Tip:*** *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click **'Recently Created Issues Report'** from the dropdown menu that displays. The following form will appear:

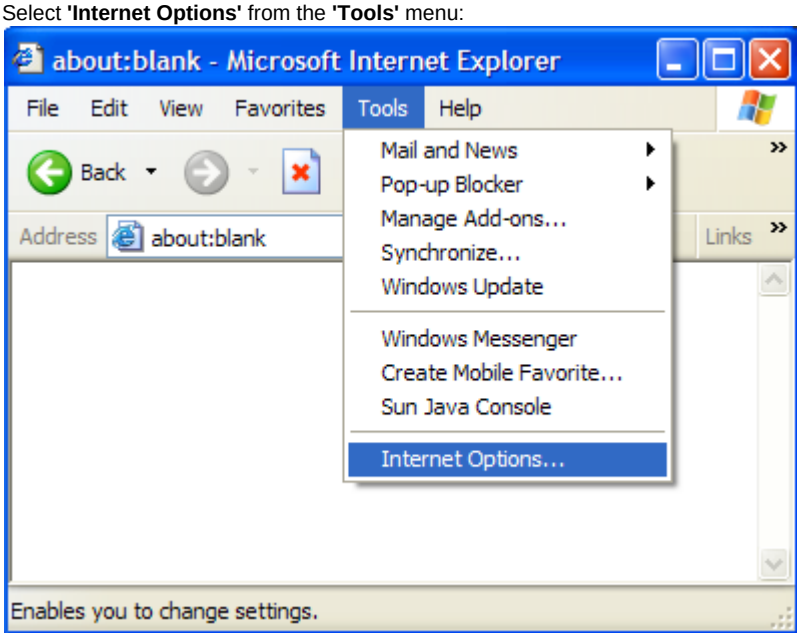


4. Click **'Select Filter or Project'.**
5. The popup **Filter or Project Picker** will appear. Select the project, or issue filter, in which you are interested. You will then be returned to the form.
6. In the **'Period'** field, select the timeframe on which the report will be based:
    - **'Hourly'**
    - **'Daily'**
    - **'Weekly'**
    - **'Quarterly'**
    - **'Yearly'**
7. In the **'Days Previously'** field, enter the number of days' worth of data (counting backwards from today) to be included in the report.

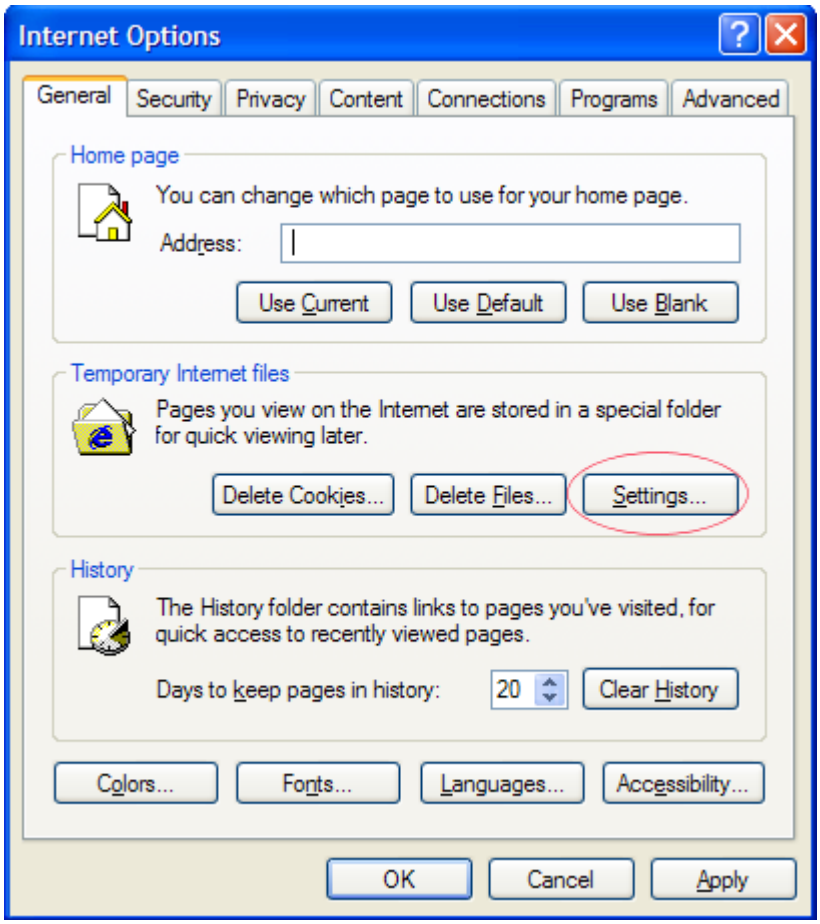
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

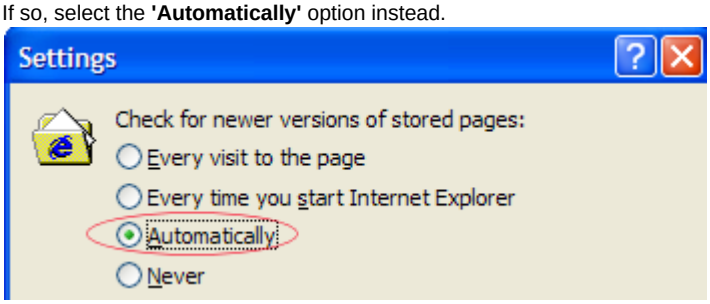
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Time Since Issues Report

The 'Time Since Issues' report is a bar chart showing the number of issues for which your chosen date field (e.g. 'Created', 'Updated', 'Due', 'Resolved', or a custom field) was set on a given date. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).

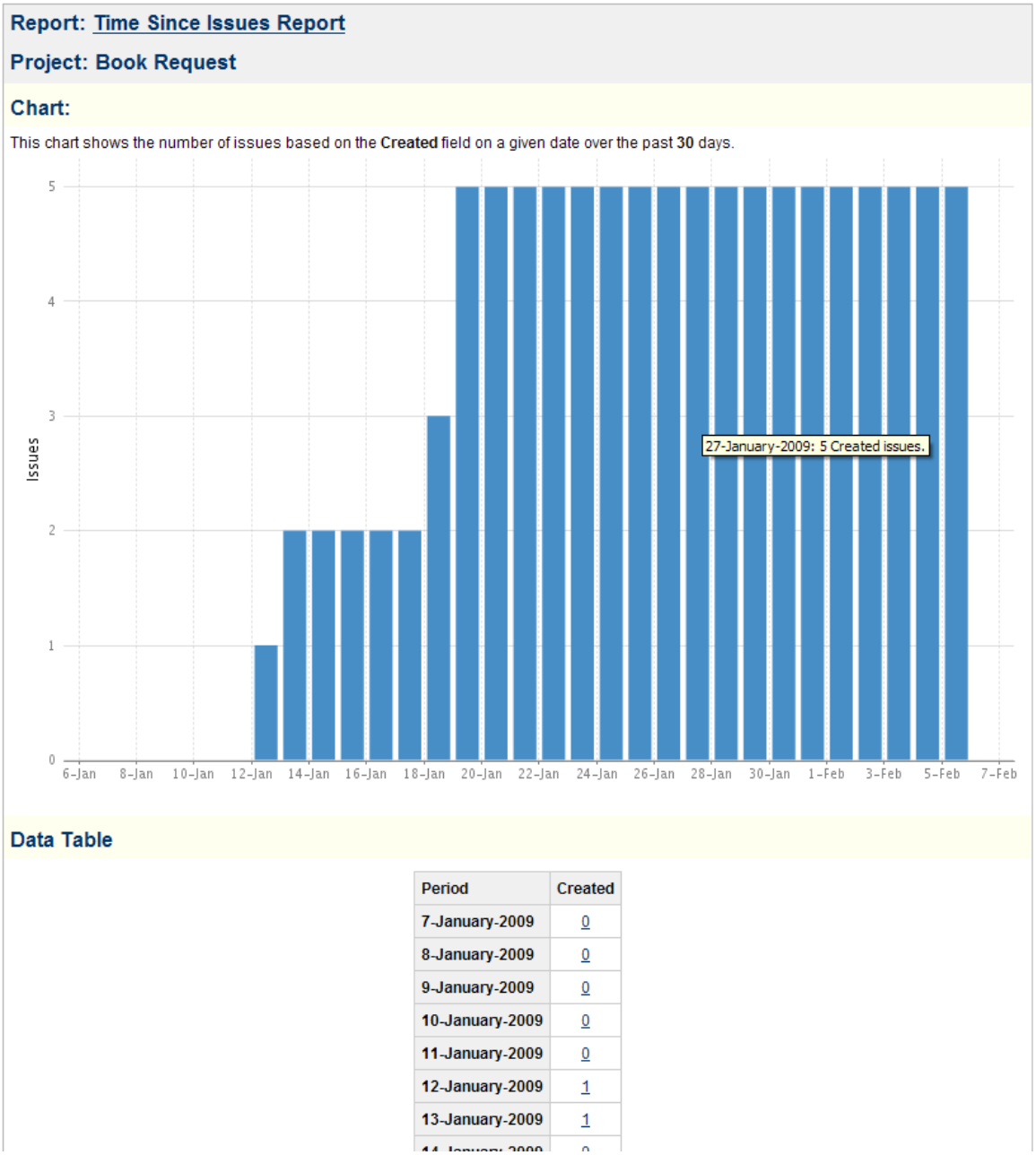
On this page:

- What does the 'Time Since Issues' report look like?
- Generating a 'Time Since Issues' report
- Configuring your Internet Explorer cache settings

What does the 'Time Since Issues' report look like?

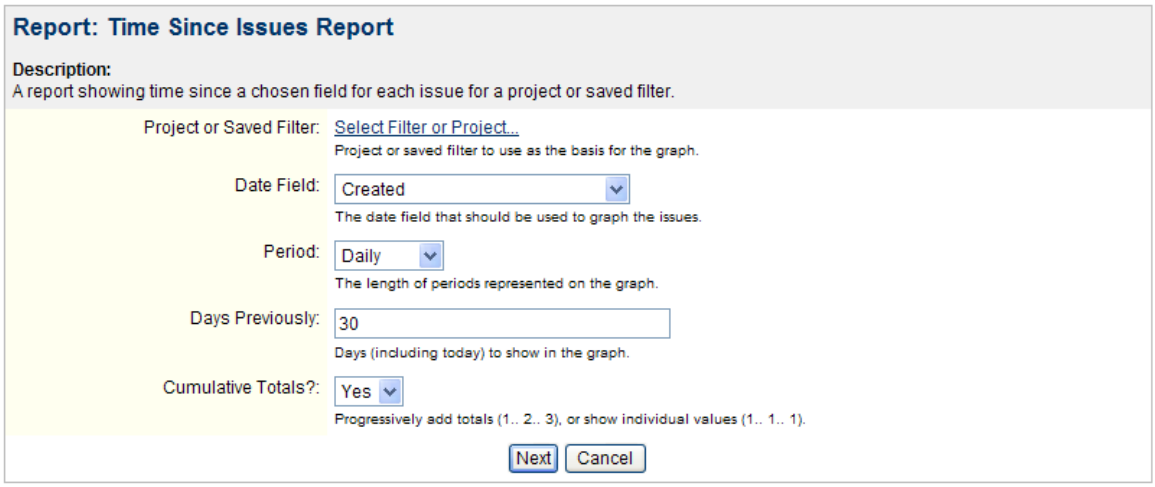
The report generated will look something like this:

Screenshot: 'Time Since Issues' report



Generating a 'Time Since Issues' report

1. On the top navigation bar, click the white triangle next to '**Projects**'. The projects dropdown will display.
   Tip: *If you click on the '**Projects**' link instead of the triangle, the summary for your current project will display.*
2. Click the project that you wish to browse. If it is not displayed in the dropdown, click '**View All Projects**' — you will be able to view a list of all projects set up in your JIRA instance and select your project from there. (The choice of project will not affect the report.)
3. Click the **'Reports'** menu on the right of the page and click **'Time Since Issues Report'** from the dropdown menu that displays. The following form will appear:

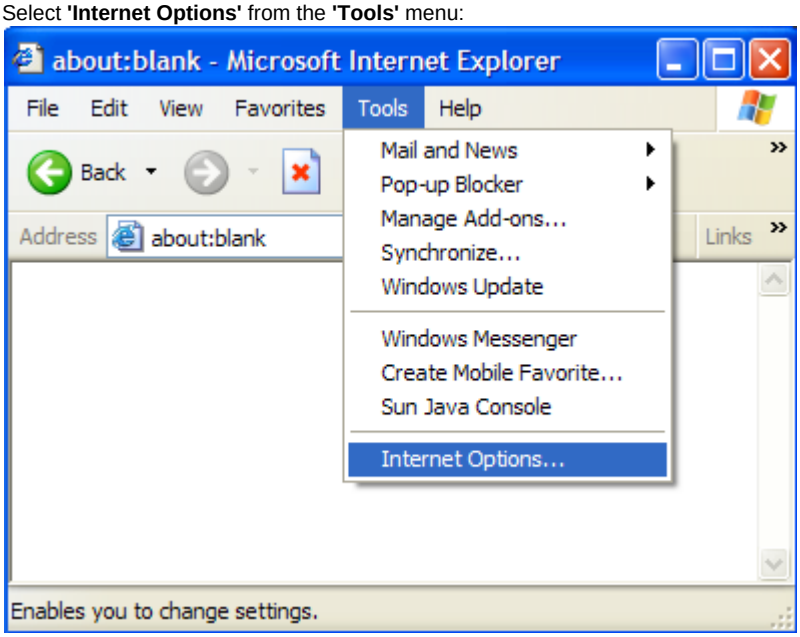


4. Click **'Select Filter or Project'.**
5. The popup **Filter or Project Picker** will appear. Select the project, or issue filter, in which you are interested. You will then be returned to the form.
6. In the **'Date Field'** field, select the date in which you are interested (e.g. **'Created', 'Updated', 'Due' , *'Resolved',** or a custom field of type 'Date').
   *Note: only available if time tracking has been enabled by your JIRA administrator.
7. In the **'Period'** field, select the timeframe on which the report will be based:
   - **'Hourly'**
   - **'Daily'**
   - **'Weekly'**
   - **'Quarterly'**
   - **'Yearly'**
8. In the **'Days Previously'** field, enter the number of days' worth of data (counting backwards from today) to be included in the report.
9. In the **'Cumulative Totals?'** field, choose either:
   - **'Yes'** to progressively add data to the preceding column; or
   - **'No'** to show just a single value in each column.

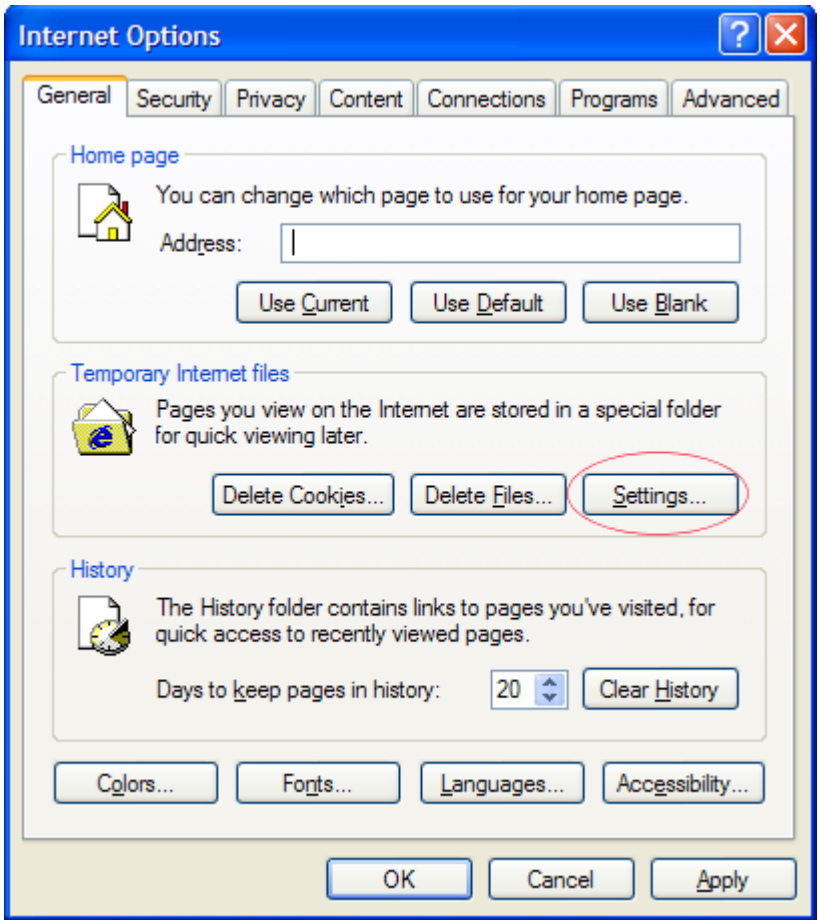
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

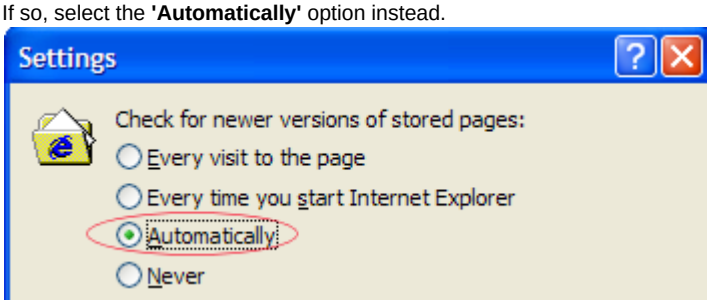
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Browsing a Project

The project browser screen allows you to browse a project or its components or versions, the latter of which shows you summaries of your project's progress.

This screen provides a general overview of your project, with a variety of easily accessible reports for your project's issues, builds and source code reviews, from which you can 'dig down' into further detail.

From the project browser screen, you can browse the following:

- **Project:**
  - Summary — Shows recent activity in your project, plus a list of issues that are due soon.
  - Issues — Shows a summary of all issues in a project grouped by Status. Also shows summaries of all unresolved issues, grouped by Assignee, Priority, Version and Component.
  - Road Map — Shows unresolved issues for upcoming versions of a project.
  - Change Log — Shows resolved issues for previous versions of a project.
  - Popular Issues — Shows a project's unresolved issues, ordered by popularity (votes).
  - Versions * — Shows a summary of recent versions for a given project.
  - Components * — Shows a summary of all components for a given project.
  - Builds * — Shows recent Bamboo builds for a given project.
  - Source ** — Shows recent FishEye changesets for a given project.
  - Reviews ** — Shows recent Crucible code for a given project.
- **Version:**
  - Version Summary — Shows recent activity in a given version of a project, plus a list of issues that are due soon.
  - Version Issues — Shows issues belonging to a given version of a project.
  - Version Popular Issues — Shows unresolved issues for a given version, ordered by popularity (votes).

- Version Builds * — Shows recent Bamboo builds for a given version.

- **Component:**
  - Component Summary — Shows recent activity in a given component of a project, plus a list of issues that are due soon.
  - Component Issues — Shows issues belonging to a given component of a project.
  - Component Road Map — Shows unresolved issues for a given component, for upcoming versions of the project.
  - Component Change Log — Shows resolved issues for a given component, for previous versions of the project.
  - Component Popular Issues — Shows unresolved issues for a given component, ordered by popularity (votes).

** Only available if your organisation uses Atlassian Bamboo and your administrator has integrated Bamboo with JIRA.*

*** Only available if your organisation uses Atlassian FishEye and your administrator has integrated FishEye with JIRA.*

See also

- JIRA Reports Overview

Browsing a Project's Summary

The **Summary** page for a project in JIRA shows recent activity in the project, plus a list of versions and issues that are due soon.

To browse a project's summary,

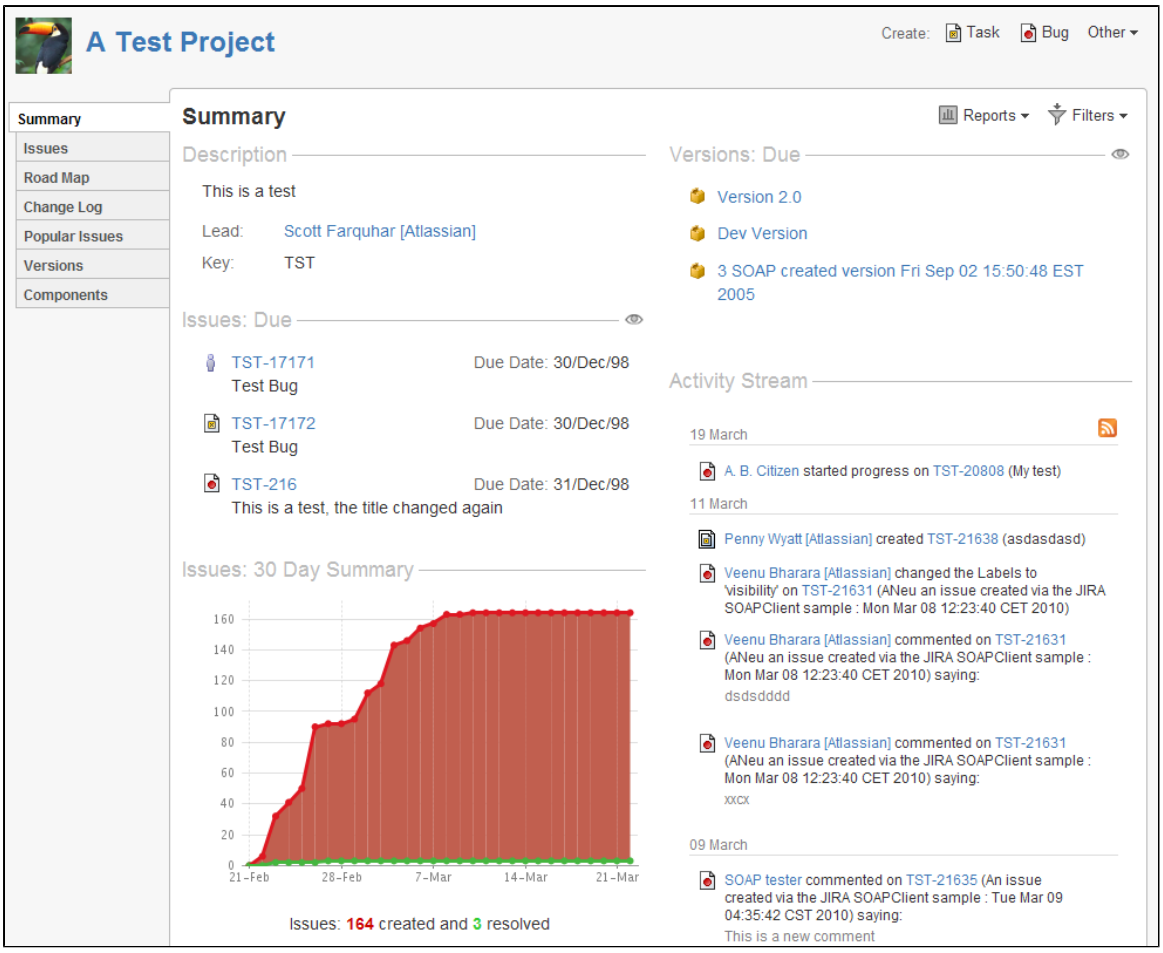
1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Ensure that the 'Summary' tab page is displayed (see screenshot below). If not, click the **Summary**tab on the left to show this page.
   - Click the appropriate icon (e.g. **Bug**, **Improvement**, **Task,** etc) next to the 'Create' label in the top right to create an issue of that issue type.
   - Click the **Reports** menu and click the relevant dropdown menu item to generate reports for the project. See Generating Reports for more information.
   - Click the **Filters** menu and click the relevant dropdown menu item to view issues in the Issue Navigator with the relevant filter applied. See Saving Searches ('Issue Filters') for more information on filters.
   - In the 'Activity Stream' section,
     - Click the RSS icon to generate an RSS feed of information that is relevant to this project.
     - Click any item to jump to recent activity associated with this project. The Activity Stream can include:
       - Issues in your local JIRA system.
       - Issues in another JIRA system (provided your administrator has configured a two-way Application Link).
       - Activity from another Atlassian application, such as:
         - document updates (from Confluence)
         - code commits (from FishEye)
         - code reviews (from Crucible)
         - builds (from Bamboo)

           Note that this requires your administrator to configure a two-way Application Link.
       - Activity from remote applications (note that your administrator will need to set this up via the REST API or the provider plugin API, or locally via Java).

     Note that your administrator will also need to configure appropriate Project Links.

> **Extending your Project Summary**
>
> The Project Summary page can be easily extended via plugins. For example, you can add a Calendar tab or a Labels tab via the JIRA Calendar plugin and JIRA Labels plugin respectively. Check out the Atlassian Plugin Exchange for more information.

Screenshot: 'Summary' page for a Project



***Tip:** You can click the icon in the 'Issues: Due' or 'Issues: Updated recently' sections to view an extended list of issues through the Issue Navigator. Clicking this icon in the 'Versions: Due' section takes you through to the versions tab.*

Related Topics

- Browsing a Project
- JIRA Reports Overview

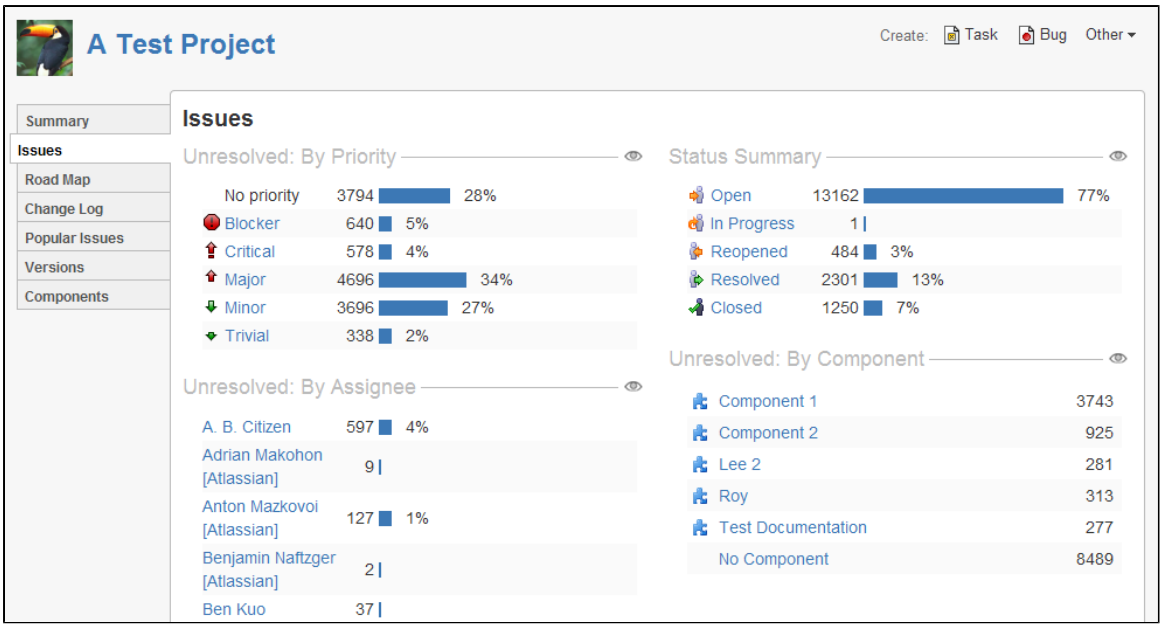
Browsing a Project's Issues

JIRA's **Issues** report shows a summary of all issues in a project grouped by Status, as well as summaries of all unresolved issues, grouped by Assignee, Priority, Version and Component.

To browse a project's Issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the '**Issues**' tab on the left of the page. The summary of issues for your project will display (see screenshot below):

Screenshot: Viewing the Issues Summary for a Project



To see which issues have a particular priority, assignee or status, or belong to a particular component or version of the project, click the name of the relevant priority/assignee/status/component/version.

Related Topics

- Browsing a Project
- JIRA Reports Overview

Browsing a Project's Road Map

JIRA provides a **Road Map** for each project, which shows issues scheduled for the next ten unreleased versions (whereas the Change Log shows released versions). The Road Map provide an overview of progress made towards releasing a version.

If your administrator has hidden the '**Fix For Version**' field, the Road Map report will not be available.

To browse a project's Road Map,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Road Map'** tab on the left of the page. The road map for your project will display (see screenshot below).
   - Click the grey arrow next to any version to expand the list of issues related to that version.
   - Click the **'View personal road map'** link to see issues assigned to you for all unreleased versions of a project.

Screenshot: Viewing a project's road map

A live version of this example can be seen online.

Related Topics

- The Change Log — looking back at recent releases rather than forward
- Browsing a Project
- JIRA Reports Overview

Browsing a Project's Change Log

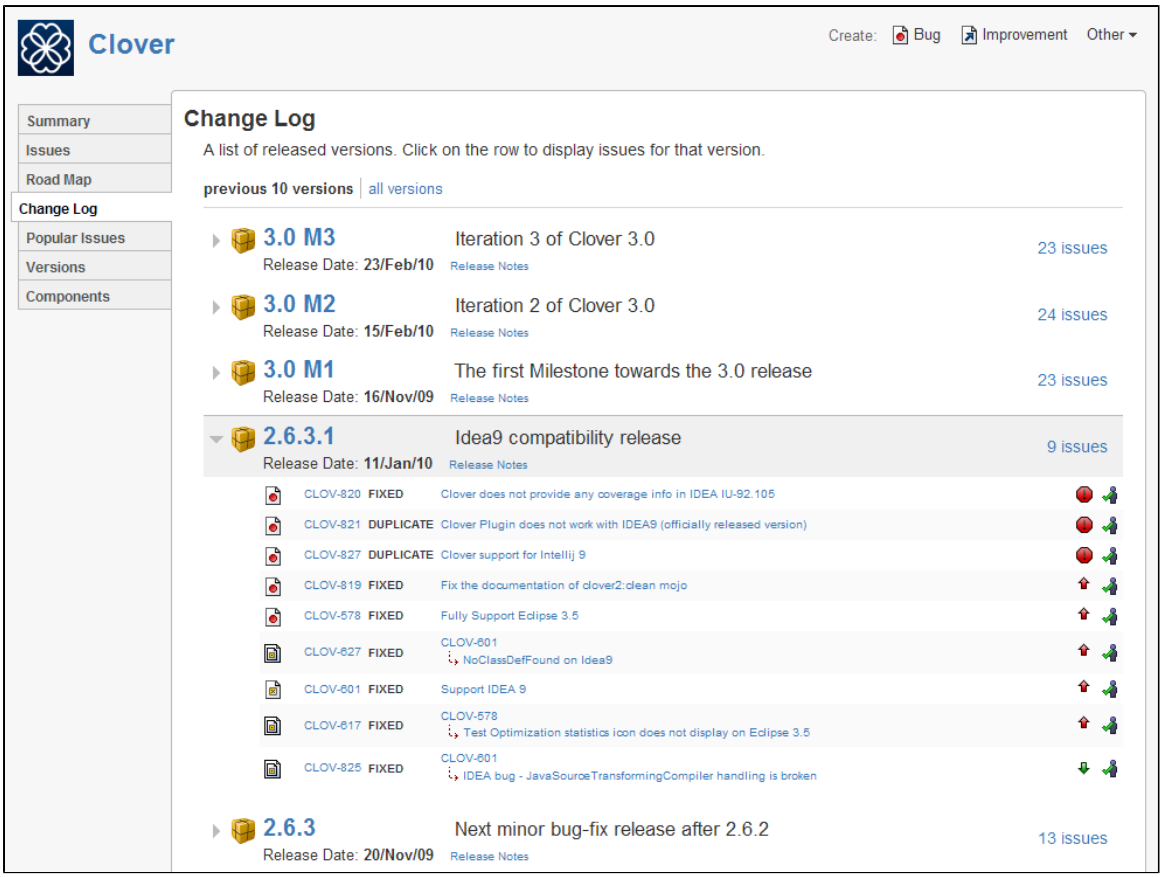
JIRA's **Change Log** report shows resolved issues in the last ten released versions of a project. Whereas the Road Map looks forward, the Change Log looks back, giving an overall view of issues resolved in recent versions.

> If your administrator has hidden the '**Fix For Version**' field, the Change Log report will not be available.

To browse a project's Change Log,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Change Log'** tab on the left of the page. The change log for your project will display (see screenshot below).

Screenshot: Viewing a project's change log



A live version of this example can be seen online.

Related Topics

- The Road Map — looking forward to future releases
- Browsing a Project
- JIRA Reports Overview

Browsing a Project's Popular Issues

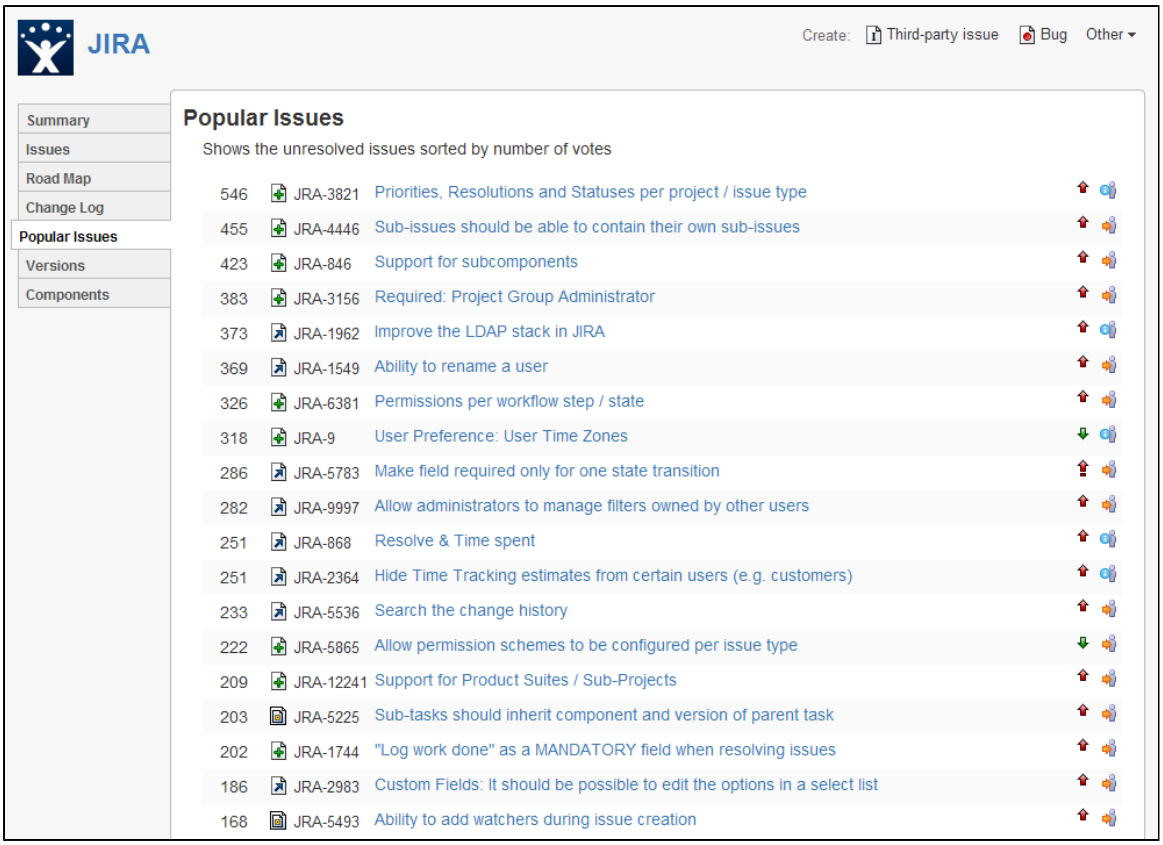
The **Popular Issues** page for a project in JIRA shows unresolved issues in a project, sorted by number of votes.

> Please note, this report is only visible if voting is enabled in your JIRA instance.

To browse a project's popular issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Popular Issues'** tab on the left of the page. The popular issues for your project will display (see screenshot below):

Screenshot: 'Popular Issues' page for a Project



Related Topics

- Browsing a Project
- JIRA Reports Overview

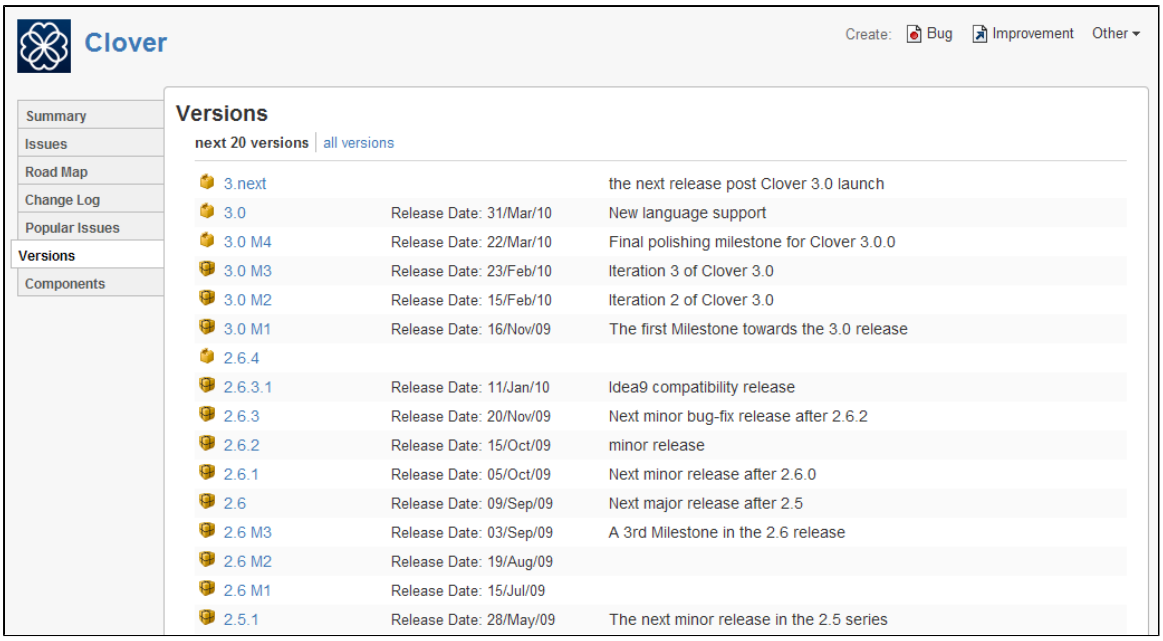
Browsing a Project's Versions

JIRA's **Versions** report shows a summary of all versions (if any have been created) in a project.

To browse a project's versions,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Versions'** tab on the left of the page. A list of versions for your project will display (see screenshot below).
   - Click the link for a version to browse that version.

Screenshot: 'Versions' page for a Project



A live version of this example can be seen online.

For each version, see also:

- Browsing a Version's Summary
- Browsing a Version's Issues
- Browsing a Version's Popular Issues
- Browsing a Version's Bamboo Builds

Related Topics

- Browsing a Project
- JIRA Reports Overview

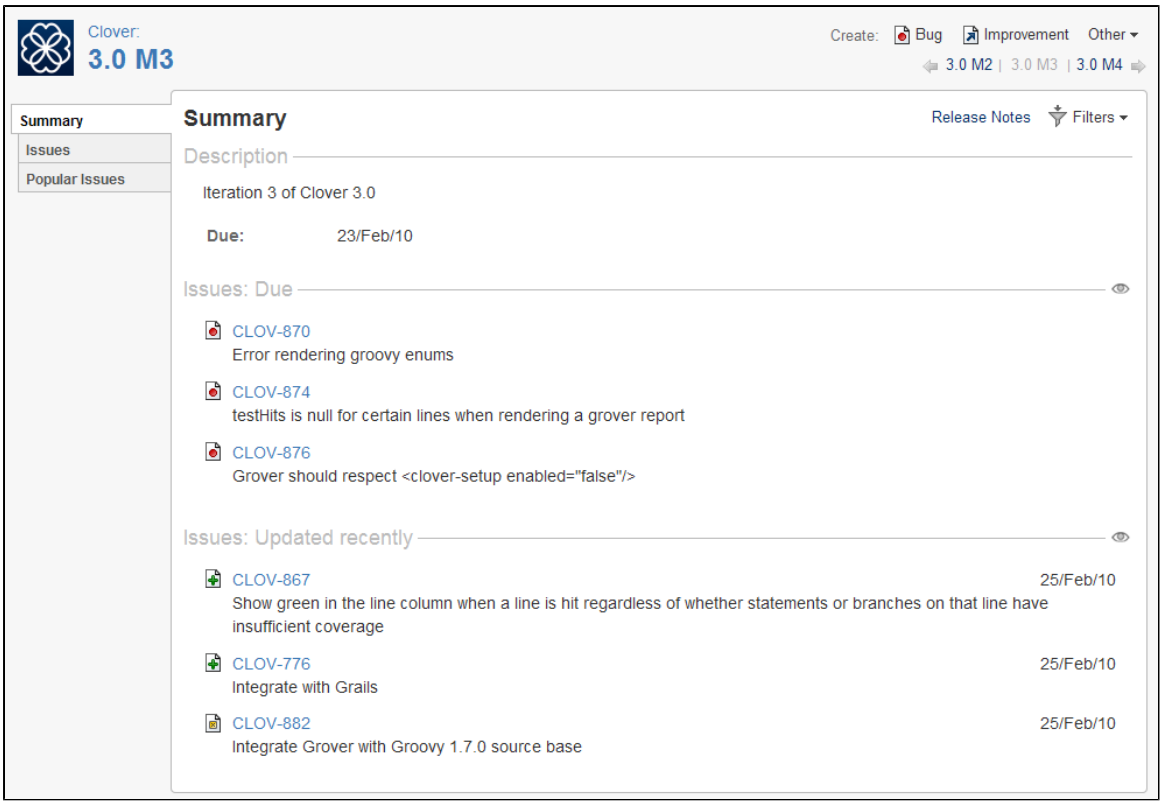
Browsing a Version's Summary

JIRA provides a **Summary** of each version of a project, which shows recent activity in that version, plus a list of issues that are due soon.

To browse a version's summary,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Versions'** tab on the left of the page. Click the name of the version in which you are interested.
4. Click the **'Summary'** tab. The summary for your version will display (see screenshot below) displaying the recently updated issues related to the version, by default.
   - Click the '**Release Notes**' link to view the release notes for the version (if released).
   - Click the '**Filters**' menu and select the filter to apply to the issues displayed on the screen.
   - Click any of the icons to go through to the Issue Navigator and see the full list of issues.
   - You can also navigate to the version summaries for the versions prior to the current version and the versions following the current version by clicking the version links above the 'Release Notes' and 'Filters' menus.

Viewing a project version's summary



Related Topics

- Browsing a Project
- JIRA Reports Overview

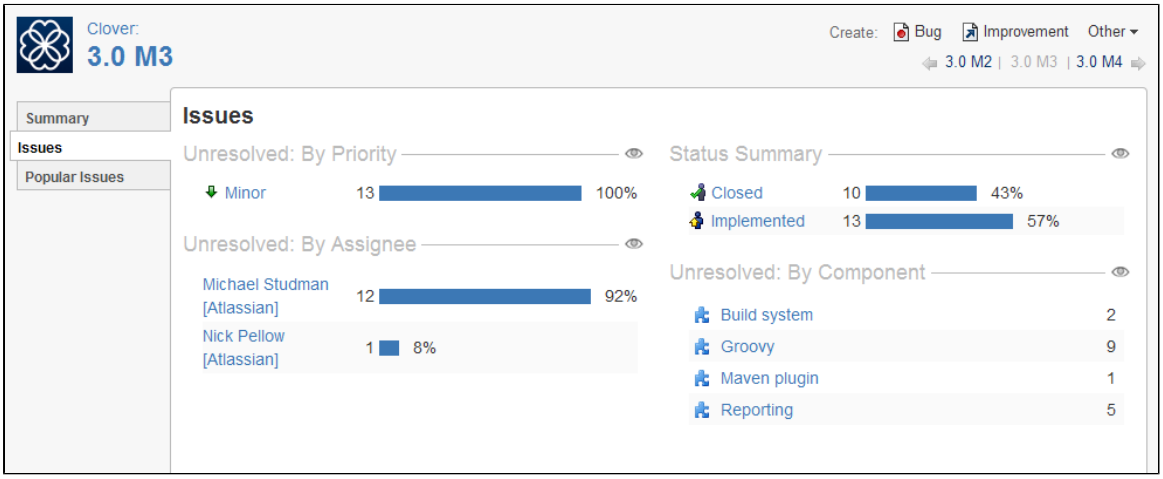
Browsing a Version's Issues

JIRA provides a list of issues for each version of a project.

To browse a version's issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Versions'** tab.
4. A list of versions will be displayed. Click the name of the version in which you are interested.
5. Click the **'Issues'** tab on the left of the page. The issues summary for your version will display (see screenshot below).
   - To see which issues have a particular priority, assignee or status, or belong to a particular component of the project, click the name of the relevant priority/assignee/status/component.

Viewing the issues summary for a version



Related Topics

- Browsing a Project
- JIRA Reports Overview

Browsing a Version's Popular Issues

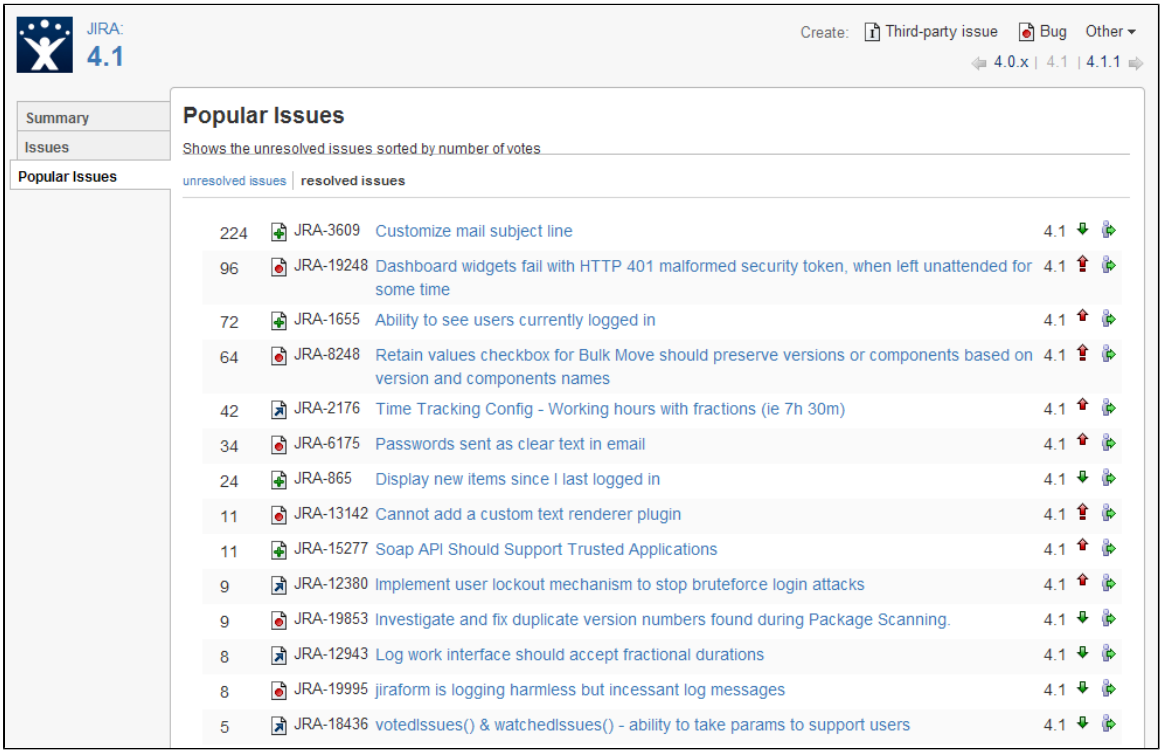
JIRA's **Popular Issues** report shows unresolved issues in a given version of a project, sorted by number of votes. It is particularly useful on public JIRA installations.

> This report is only visible if your JIRA administrator has enabled voting in your JIRA instance.

To browse a version's popular issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Versions'** tab on the left of the page.
4. Click the name of the version in which you are interested.
5. Click the **'Popular Issues'** tab. The unresolved popular issues for your version will display (see screenshot below). To see resolved popular issues (instead of unresolved popular issues), click **'resolved issues'.**

Screenshot: Popular issues for a version



Related Topics

- Browsing a Project
- JIRA Reports Overview

Browsing a Version's Bamboo Builds

If your organisation uses Atlassian's Bamboo and your administrator has integrated Bamboo with JIRA, JIRA enables you to view the Bamboo build plan status and recent build activity for a version of a project. The **Builds** tab provides you with a list of the builds which are related to the project version, including:

- the list of the builds which are related to the version, i.e. builds that have issues from the project version linked to them (either as 'Fixed' or 'Related'). See the Bamboo documentation for instructions on linking issues to builds.
- the latest status of the build plans for the related builds, i.e. the build plan contains a build that has a project issue linked to it. The status of a build plan for a version is determined as follows:
  - If the project version has not been released — the build plan status is the status of the latest build in the Bamboo build plan, regardless of whether the latest build is related to the version (i.e. has issues from the project version linked to it).

- If the project version has been released — the build plan status is the status of the latest build in the Bamboo build plan, that is related to the version (i.e. has issues from the project version linked to it) and is prior to or equal to the release date (or current date, if there is no release date).

To view the Bamboo build information related to a version,

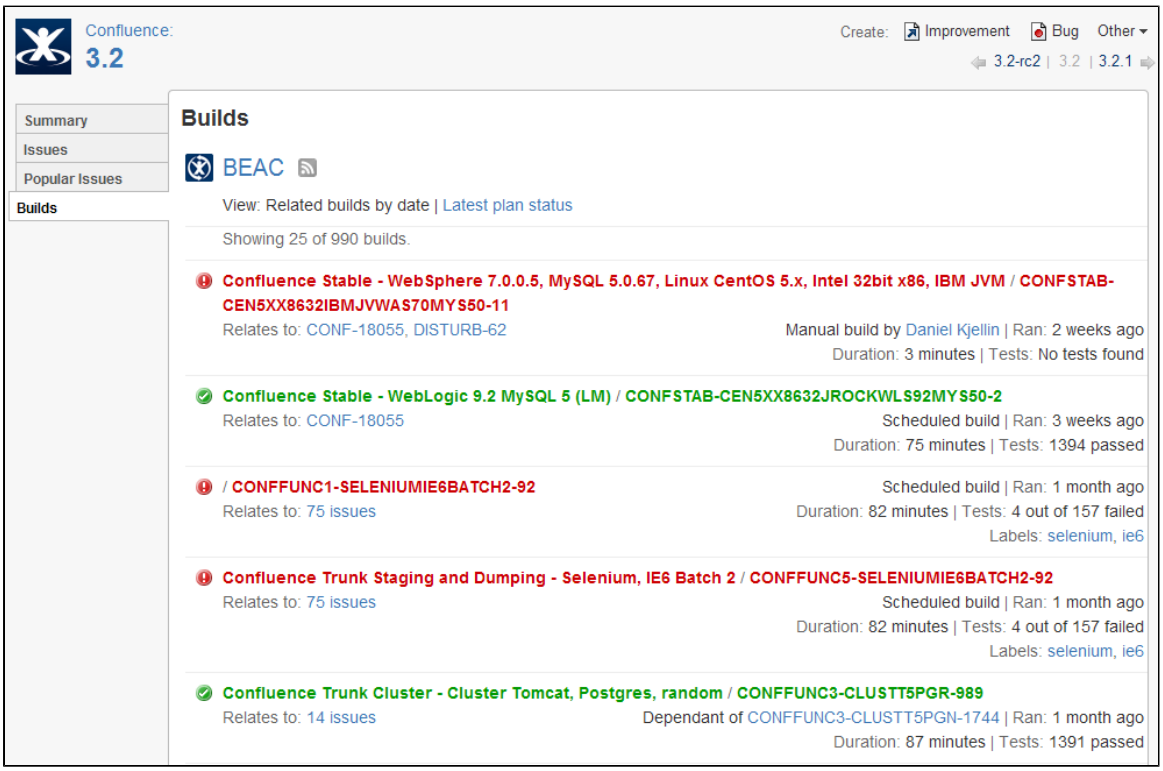
1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Versions'** tab.
4. A list of versions will be displayed. Click the name of the version in which you are interested.
5. Click the **Builds** tab. You will be able to view the following information:
   - Builds related to the Project (displayed by default)
   - Status of Build Plans related to the Project (click the **'Latest plan status'** link at the top of the '**Builds**' page)

If you cannot see the **Builds** tab, your administrator may need to add the **'View Version Control'** permission to your project.

Viewing the Builds related to the Project Version

By default, the **Builds** tab will display the list of related builds, ordered by build date in descending order.

Screenshot: Viewing the builds related to a version



Setting up an RSS feed to track Builds related to the Version

You can set up an RSS feed to track this information by clicking on the RSS icon in the top left section of the page. Each entry in the list will display information about the related build, including:

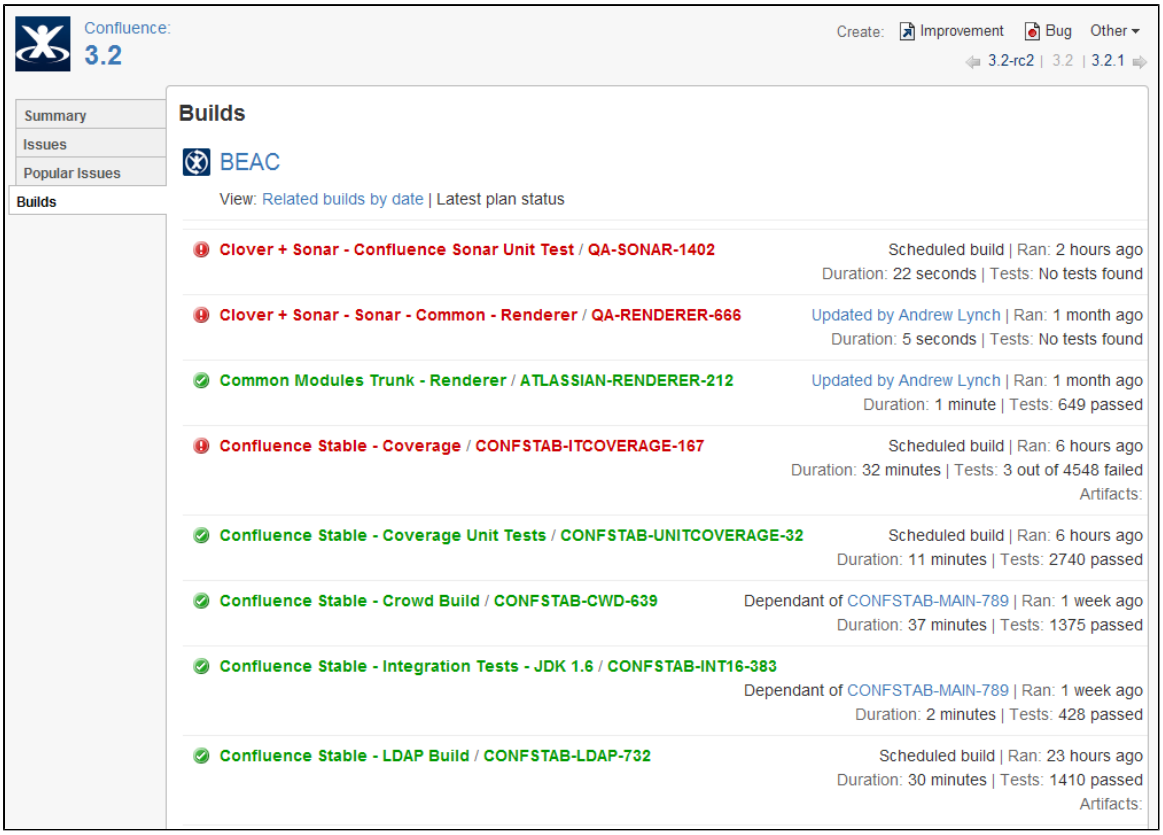
- the build name and name of the build plan
- when the build was last run
- summary information, such as related builds, duration of the build, tests passed
- build labels (if any)
- links to build artifacts (if any)

Viewing the Status of Build Plans related to the Project Version

To view the status of build plans related to the project version, click the **'Latest plan status'** link at the top of the **Build** tab. The build plans listed will show the status of the Build Plan, including information about the latest build in the plan (similar to the diagram above). Build plans

will be sorted by plan name.

Screenshot: Viewing the status of builds related to a version



Related Topics

- Viewing the Bamboo Builds related to an Issue
- Browsing a Project's Bamboo Builds

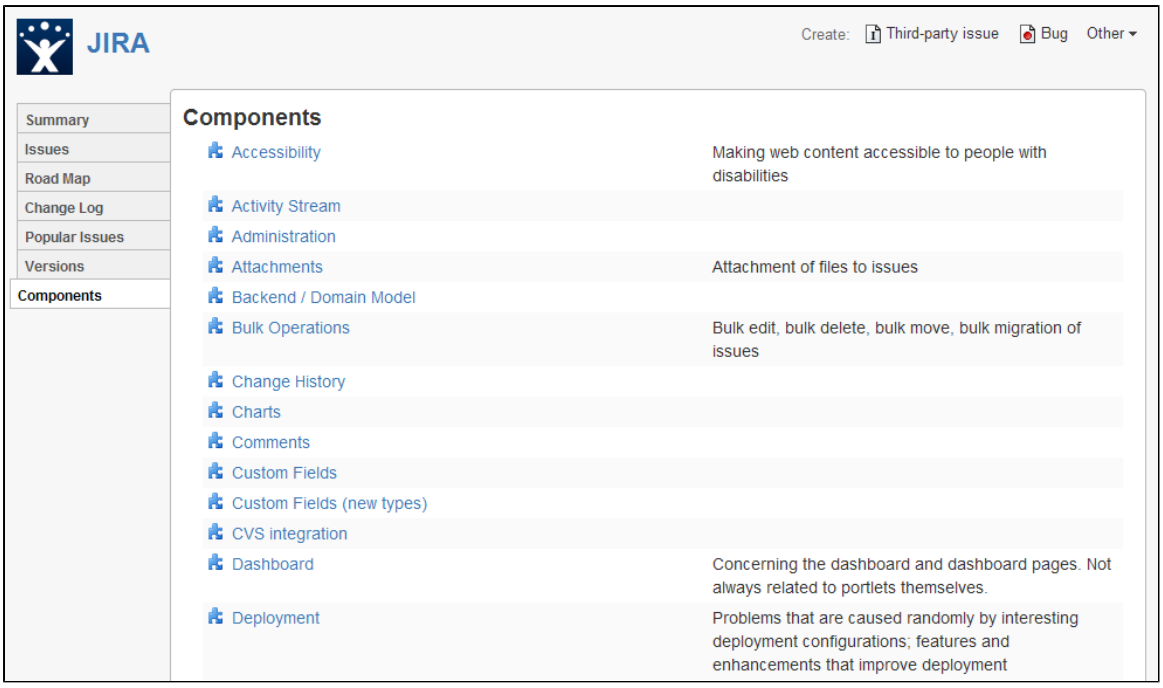
Browsing a Project's Components

JIRA's **Components** report shows a summary of all components (if any have been created) in a project.

To browse a project's components,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Components'** tab on the left of the page. A list of components for your project will display (see screenshot below).
   - Click the name of a component to browse that component.

Screenshot: 'Components' page for a Project



A live version of this example can be seen online.

For each component, see also:

- Browsing a Component's Summary
- Browsing a Component's Issues
- Browsing a Component's Road Map
- Browsing a Component's Change Log
- Browsing a Component's Popular Issues

Related Topics

- Browsing a Project
- JIRA Reports Overview

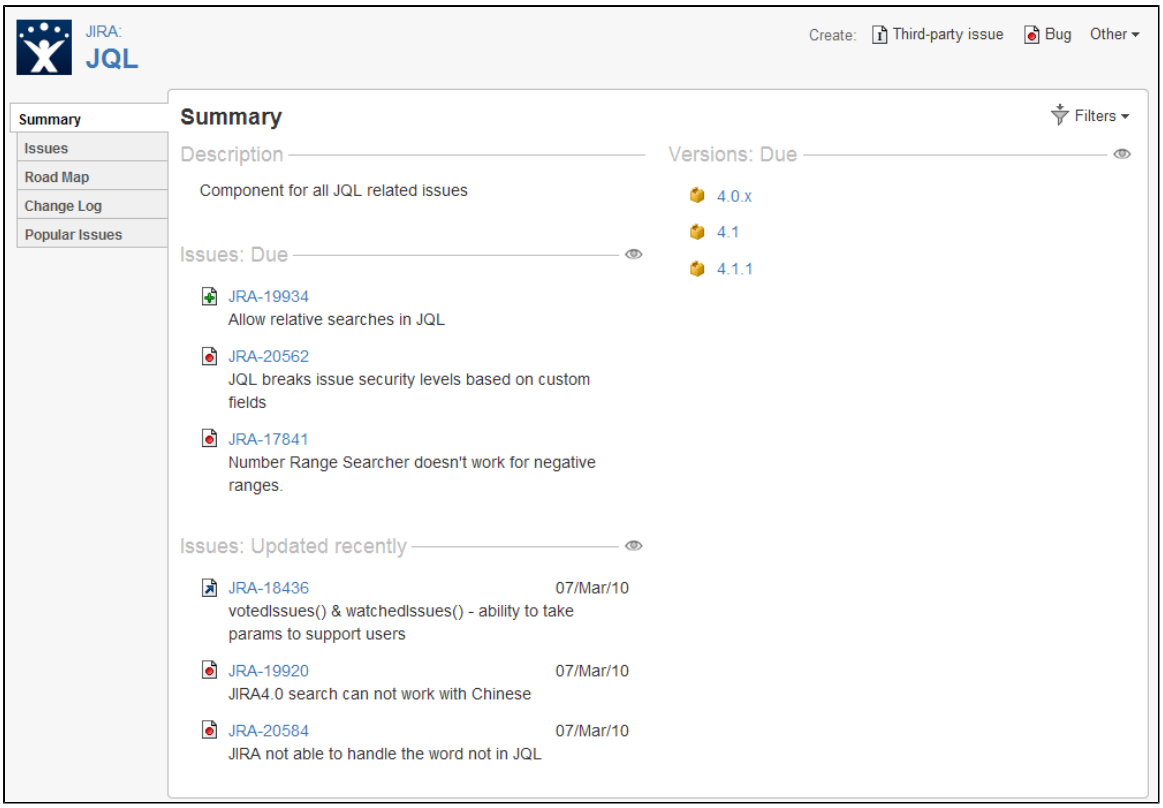
Browsing a Component's Summary

JIRA provides a **Summary** of each component of a project, which shows recent activity in the component, plus a list of issues that are due soon.

To browse a component's summary,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Components'** tab on the left of the page. Click the name of the component in which you are interested.
4. Click the **'Summary'** tab. The summary for your component will display (see screenshot below).
   - Click the icon in the '**Issues: Due**' section to go through to the Issue Navigator and see the full list of due issues.
   - Click the icon in the '**Issues: Updated recently**' section to go through to the Issue Navigator and see the full list of issues updated recently.
   - Click the icon in the '**Versions: Due**' section to view the versions in the project.

Viewing a project component's summary



Related Topics

- Browsing a Project
- JIRA Reports Overview

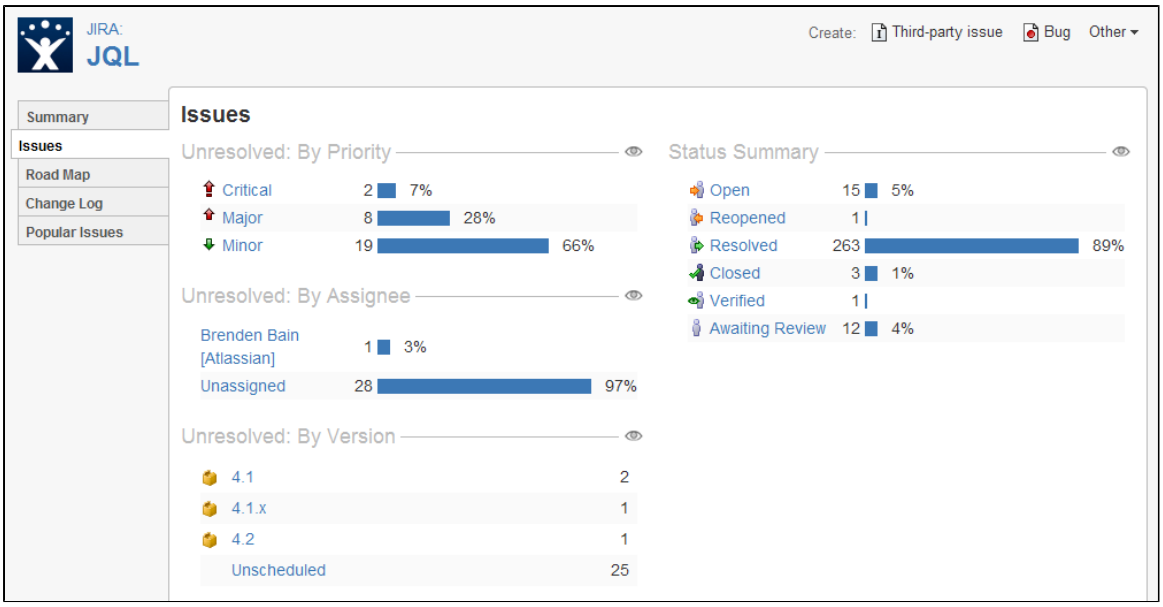
Browsing a Component's Issues

JIRA provides a list of all the issues for each component of a project.

To browse a component's issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Components'** tab.
4. A list of components will be displayed. Click the name of the component in which you are interested.
5. Click the **'Issues'** tab on the left of the page. The issues summary for your component will display (see screenshot below).
   - Click the icon in the '**Unresolved: By Priority**' section to go through to the Issue Navigator and see the full list of unresolved issues by priority.
   - Click the icon in the '**Unresolved: By Assignee**' section to go through to the Issue Navigator and see the full list of unresolved issues by assignee.
   - Click the icon in the '**Unresolved: By Version**' section to go through to the Issue Navigator and see the full list of unresolved issues by version.
   - Click the icon in the '**Status Summary**' section to go through to the Issue Navigator and see the full list of unresolved issues by status.

Viewing the issues summary for a component



Related Topics

- Browsing a Project
- JIRA Reports Overview

Browsing a Component's Road Map

JIRA provides a **Road Map** for each component of a project, which shows issues scheduled for the next ten unreleased versions (whereas the Change Log shows released versions). The Road Map provides an overview of progress made towards releasing a version.

> If your administrator has hidden the '**Fix For Version**' field, the Road Map report will not be available.

To browse a component's Road Map,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Components'** tab on the left of the page.
4. Click the name of the component in which you are interested.
5. Click the **'Road Map'** tab. The road map for your component will display (see screenshot below)
   - Click the grey arrow next to any version to expand the list of issues related to that version.
   - Click the **'View personal road map'** link to see issues assigned to you for the next four unreleased versions of a project.

Screenshot: Viewing a component's road map

Related Topics

- The Change Log — looking back at recent releases rather than forward
- Browsing a Project
- JIRA Reports Overview

Browsing a Component's Change Log

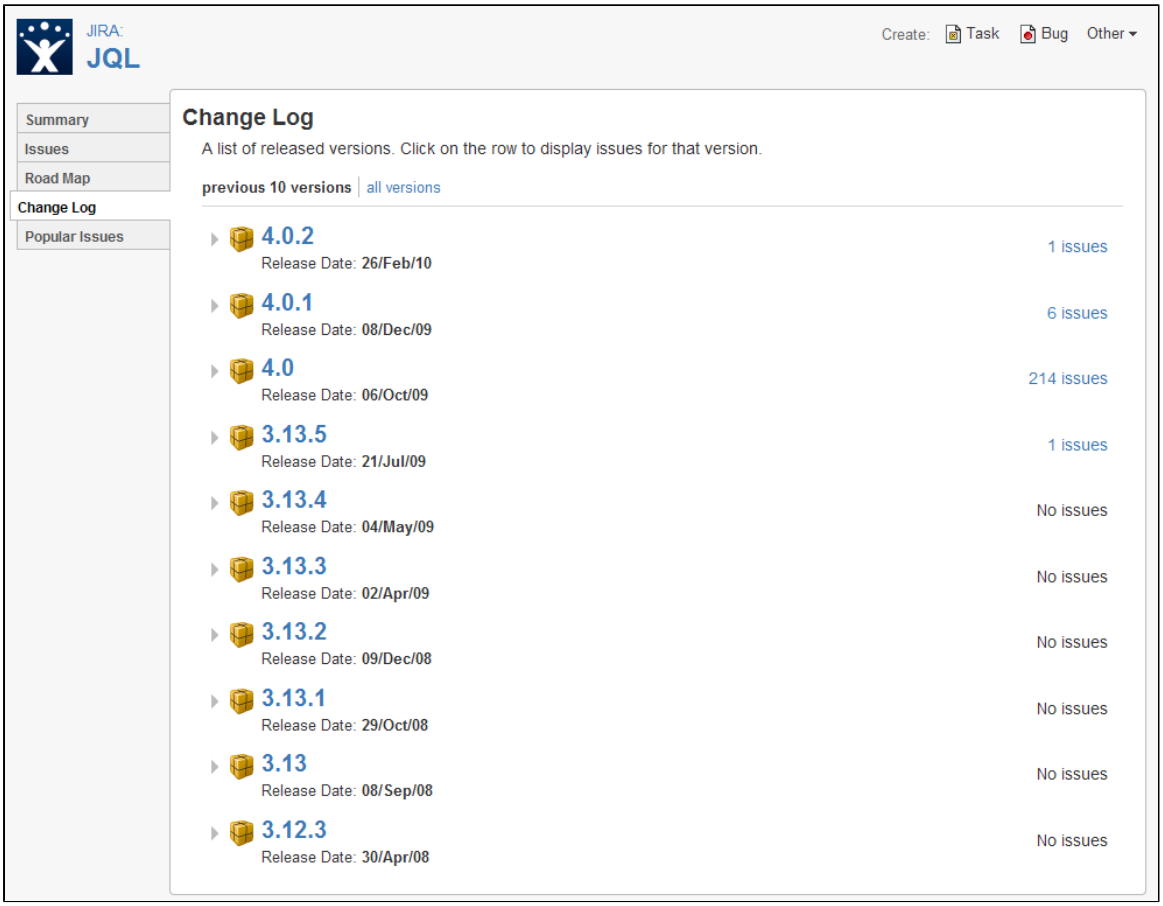
JIRA's **Change Log** report shows resolved issues in the last ten released versions of a project. Whereas the Road Map looks forward, the Change Log looks back, giving an overall view of issues resolved in recent versions.

> If your administrator has hidden the '**Fix For Version**' field, the Change Log report will not be available.

To browse a component's Change Log,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Components'** tab on the left of the page.
4. Click the name of the component in which you are interested.
5. Click the **'Change Log'** tab. The change log for your component will display (see screenshot below)
   - Click **'all versions'** to see the Change Log for all released versions (not just the latest ten).
   - Click the grey arrow next to any version to expand the list of issues related to that version.

Screenshot: Viewing a component's change log



Related Topics

- The Road Map — looking forward to next releases
- Browsing a Project
- JIRA Reports Overview

Browsing a Component's Popular Issues

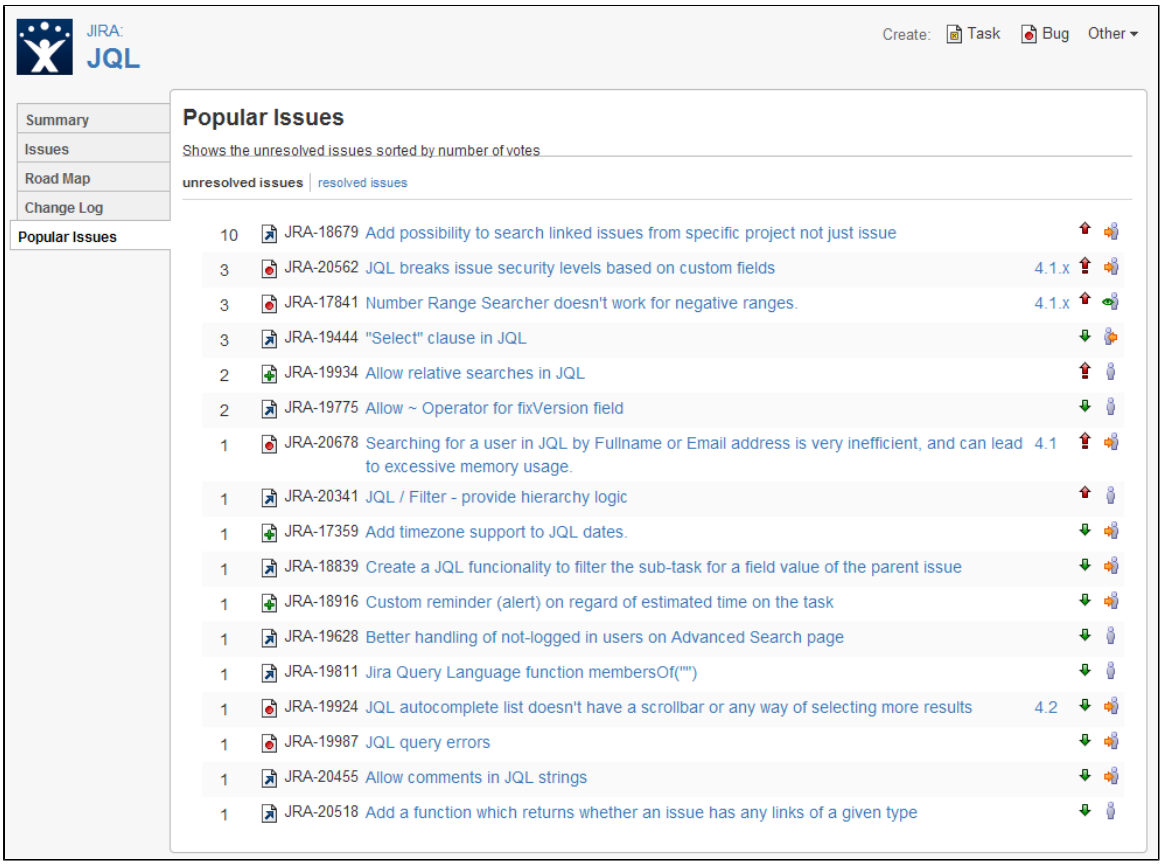
JIRA's **Popular Issues** report shows unresolved issues in a given component of a project, sorted by number of votes. It is particularly useful on public JIRA installations.

> This report is only visible if your JIRA administrator has enabled voting in your JIRA instance.

To browse a component's popular issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Components'** tab on the left of the page.
4. Click the name of the component in which you are interested.
5. Click the **'Popular Issues'** tab. The unresolved popular issues for your component will display (see screenshot below). To see resolved popular issues (instead of unresolved popular issues), click **'resolved issues'.**

Screenshot: Popular issues for a component



Related Topics

- Browsing a Project
- JIRA Reports Overview

Browsing a Project's Labels

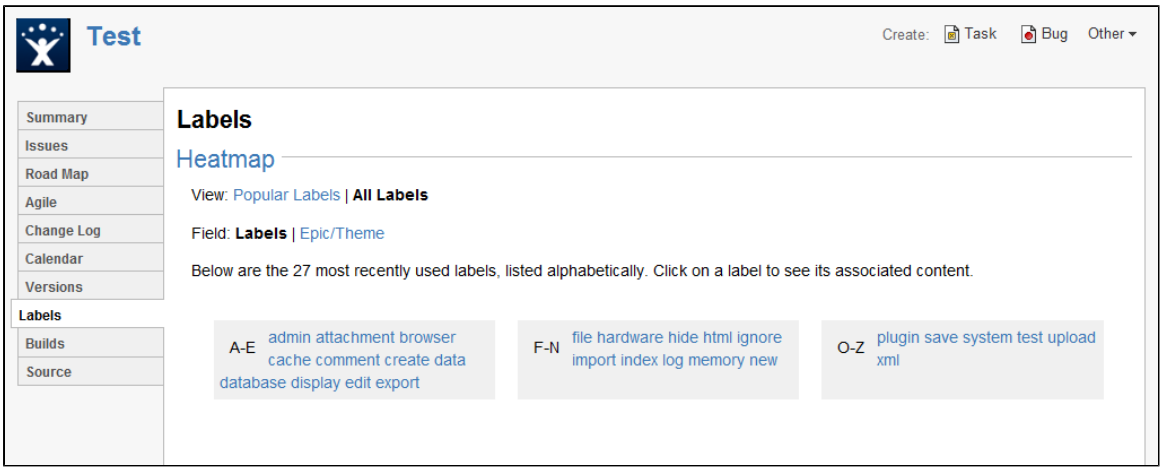
The **Labels** page for a project in JIRA shows labelled issues in a project.

To browse a project's labelled issues,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Labels'** tab on the left of the page. The labelled issues for your project will display (see screenshot below).
   - Next to the word '**View**', click '**Popular Labels**' / '**All Labels**' to toggle between the different views.

Screenshot: 'Popular Labels' view for a Project

Screenshot: 'All Labels' view for a Project



Next to the word '**Field**', the name '**Labels**' will be shown — this represents any labels that have been applied to your issues as described in Labelling an Issue (hidden draft for 4.2). If any other names are shown, you can click them to see any other Labels custom fields in your JIRA system. For example, the screenshot above contains a Labels custom field called '**Epic/Theme**'.

Related Topics

- Browsing a Project
- JIRA Reports Overview

Browsing a Project's Bamboo Builds

If your organisation uses **Atlassian's Bamboo** and your administrator has integrated Bamboo with JIRA, JIRA enables you to view the Bamboo build plan status and recent build activity for a project. The **Builds** tab provides you with the build information related to the project, including:

- the list of the builds which are related to the project, i.e. builds that have issues from the project linked to them (either as 'Fixed' or 'Related'). See the Bamboo documentation for instructions on linking issues to builds.
- The latest status of the build plans for the related builds, i.e. the build plan contains a build that has an issue from the project linked to it.

To view the Bamboo build information related to a project,

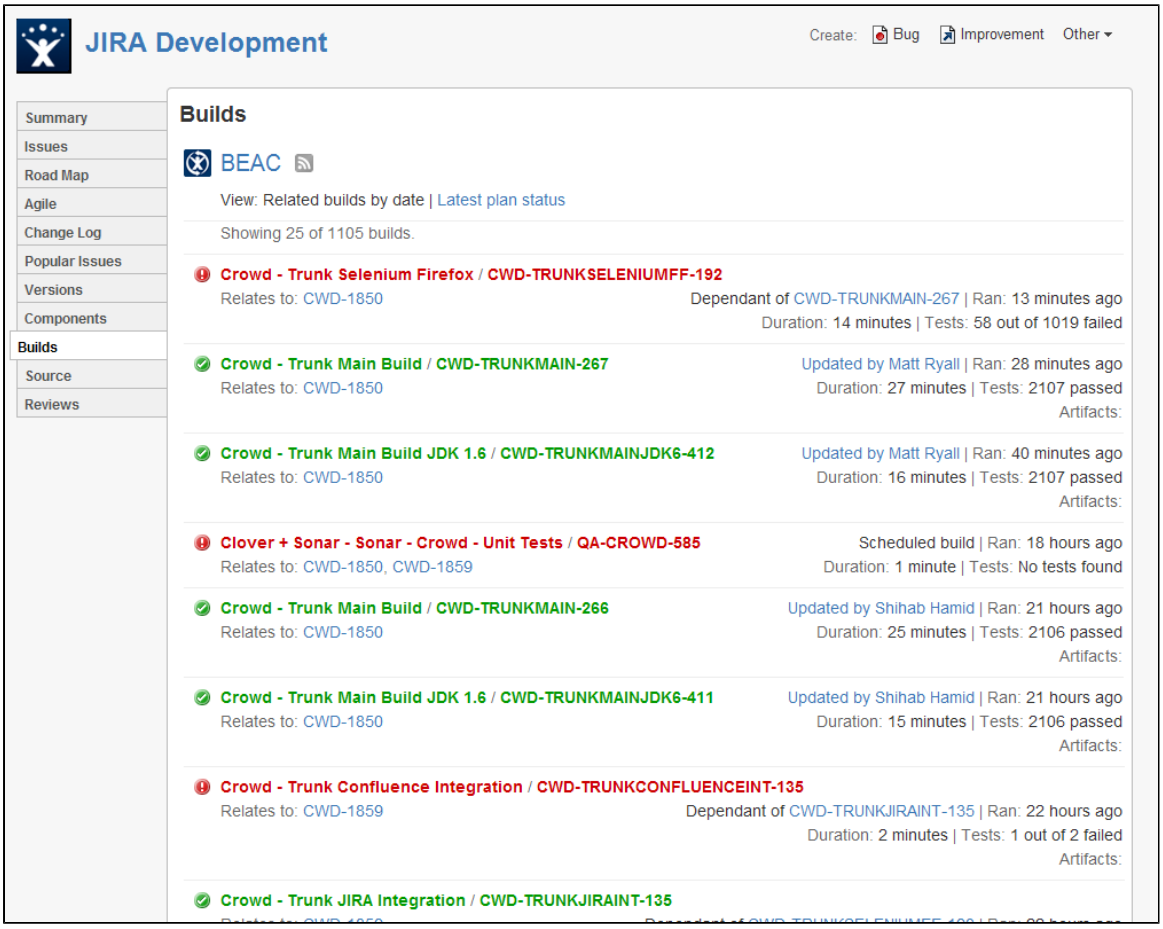
1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Builds'** tab on the left of the page. You will be able to view the following information:
   - Builds related to the Project (displayed by default)
   - Status of Build Plans related to the Project (click the **'Latest plan status'** link at the top of the '**Builds**' page)

If you cannot see the **Builds** tab, your administrator may need to add the **'View Version Control'** permission to your project.

Builds related to the Project

By default, the **Builds** tab will display the list of related builds, ordered by build date in descending order.

Screenshot: Viewing the builds related to a project



Setting up an RSS feed to track Builds related to the Project

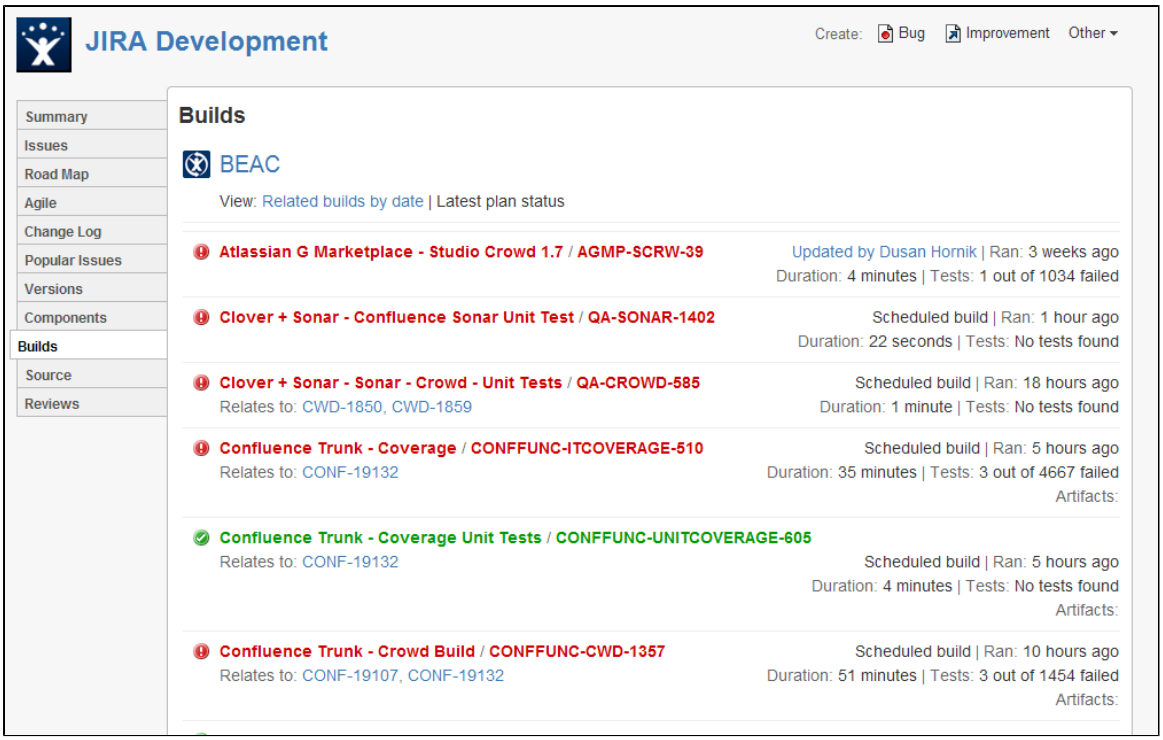
You can set up an RSS feed to track this information by clicking on the RSS icon in the top left section of the page. Each entry in the list will display information about the related build, including:

- the build name and name of the build plan
- when the build was last run
- summary information, such as related builds, duration of the build, tests passed
- build labels (if any)
- links to build artifacts (if any)

Status of Build Plans related to the Project

The build plans listed will show the status of the build plan, (i.e. status of the latest build), including information about the latest build in the plan (similar to the diagram above). Build plans will be sorted by the plan name.

Screenshot: Viewing the status of builds related to a project



Related Topics

- Viewing the Bamboo Builds related to an Issue
- Browsing a Version's Bamboo Builds

Browsing a Project's FishEye Changesets

JIRA's **Changeset** report allows you to view recent changeset activity for a project (that is, where a JIRA issue key belonging to the project was referenced in the commit message), if you are using a source-code repository together with Atlassian FishEye. You can:

- View all 'Recent Changesets' for all repository changesets across the entire project.
- View 'Activity Statistics' on LOC (lines-of-code), files or commits for the project, issue or author.
- Search the FishEye repository linked to the JIRA project currently being browsed.

> To be able to view the changeset report, your JIRA administrator must have configured the FishEye plugin appropriately. You will also need the 'View Version Control' permission in the appropriate projects.

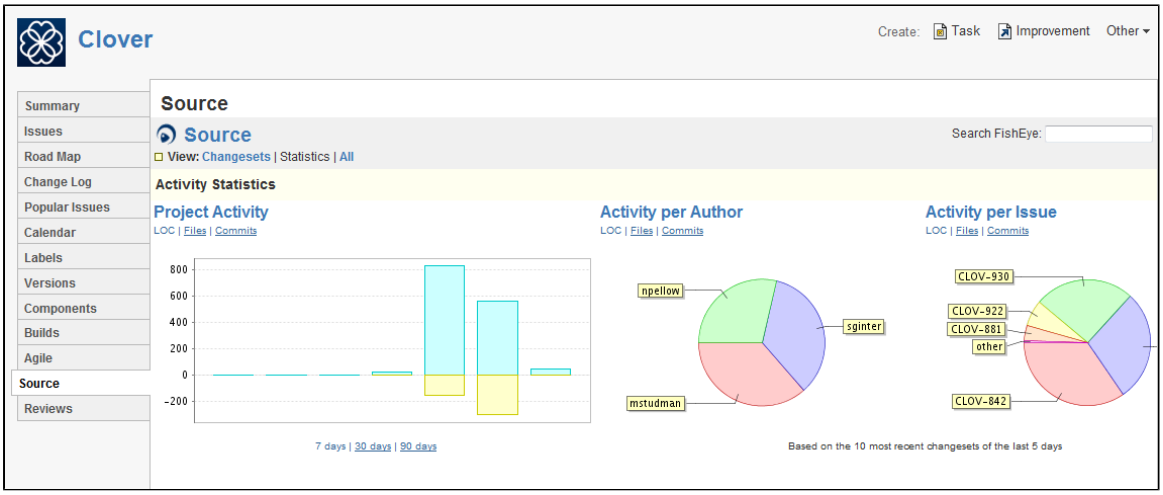
To view the changeset activity for a project,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Source'** tab on the left of the page. The recent changesets for your project will display (see screenshot below). By default, you will see a listing of the most recent changesets for a project:

Screenshot: Viewing the recent changesets for a project

You can also view the Activity Statistics on LOC, Files or Commits for the project, by clicking the '**Statistics**' link. The Activity Statistics for the project will display:

Screenshot: Viewing the activity statistics for a project



If you wish to see the above two sets of information together, you can click on the '**All**' link to view it all on one page.

Related Topics

- Viewing an Issue's FishEye Changesets

Browsing a Project's Crucible Reviews

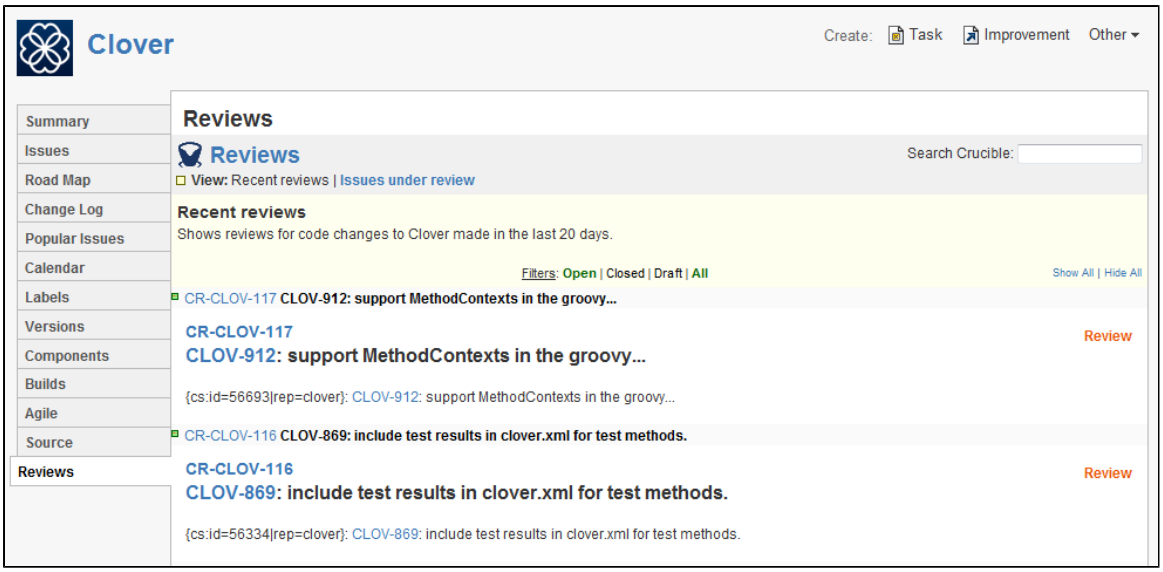
JIRA's **Reviews** report allows you to view recent code reviews activity for a project (that is, where a JIRA issue key belonging to the project was referenced in the review's description), if you are using a source-code repository together with Atlassian Crucible.

> To be able to view the **Reviews** report, your JIRA administrator must have configured the FishEye plugin appropriately. You will also need the 'View Version Control' permission in the appropriate projects.

To view the Reviews for a project,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Reviews'** tab on the left of the page. The recent changesets for your project will display (see screenshot below). By default, you will see a listing of the most recent changesets for a project:

Screenshot: Viewing the recent code reviews for a project



Viewing a Project's Burndown Chart

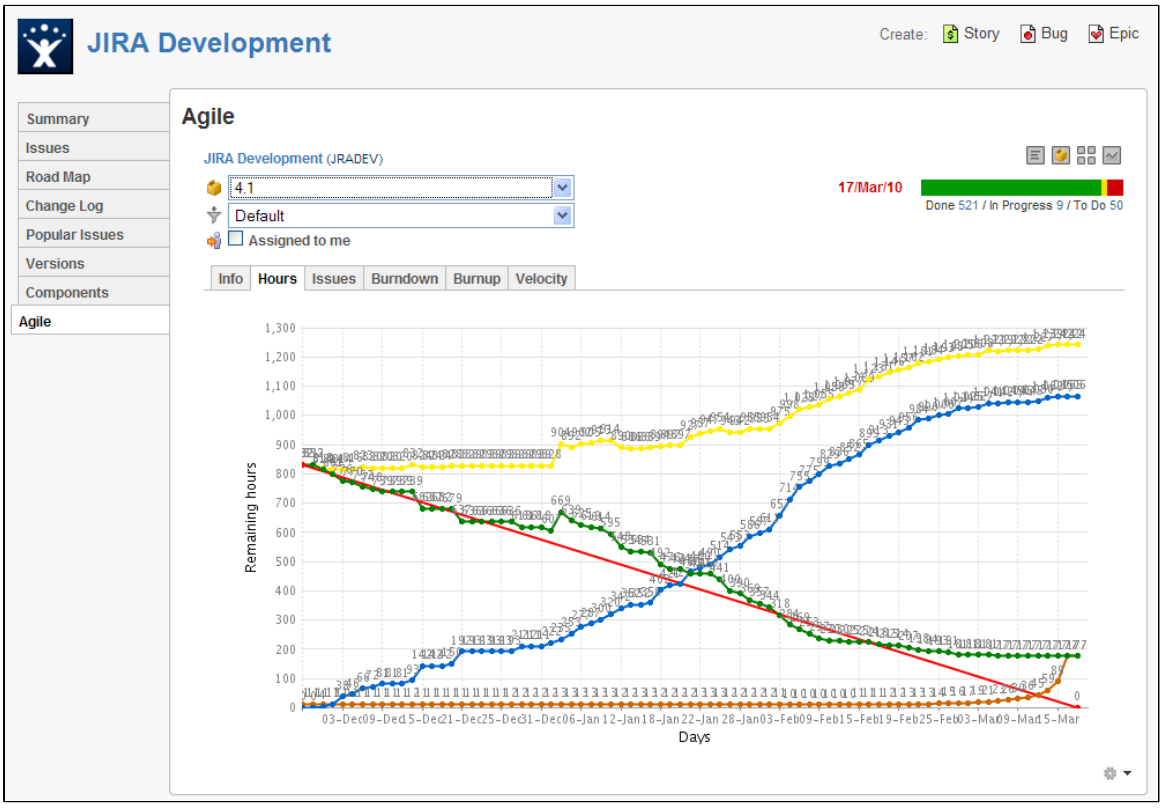
JIRA's **Agile** report allows you to view information about a project's Backlog and its various 'Burndown' Charts, if you are using the Atlassian GreenHopper plugin.

> To be able to view the **Agile** report, your JIRA administrator must have configured the GreenHopper plugin appropriately.

To view information about a project's Backlog and Burndown Charts,

1. On the top navigation bar, click the white triangle next to 'Projects'. The projects dropdown will display.
   Tip: You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project that you wish to browse. If the project is not displayed in the dropdown, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA site, and select your project from there.
3. Click the **'Agile'** tab on the left of the page. The backlog for your project will display:

Screenshot: Viewing the Hour Burndown Chart for a project



On this page, you can:

- Use the '**Version**' drop-down to display the backlog for a different project version.
- Use the '**Context**' drop-down to select a different GreenHopper context.
- Select the '**Info**' tab to display more information about the backlog for the selected project version.
- Select one of the chart tabs (**Hours**, **Issues**, **Burndown**, **Burnup** or **Velocity**) to view the GreenHopper chart for your selected project version and context.

  Velocity Charts are also known more generically as 'Value Charts'.

Customising the Dashboard

On this page:

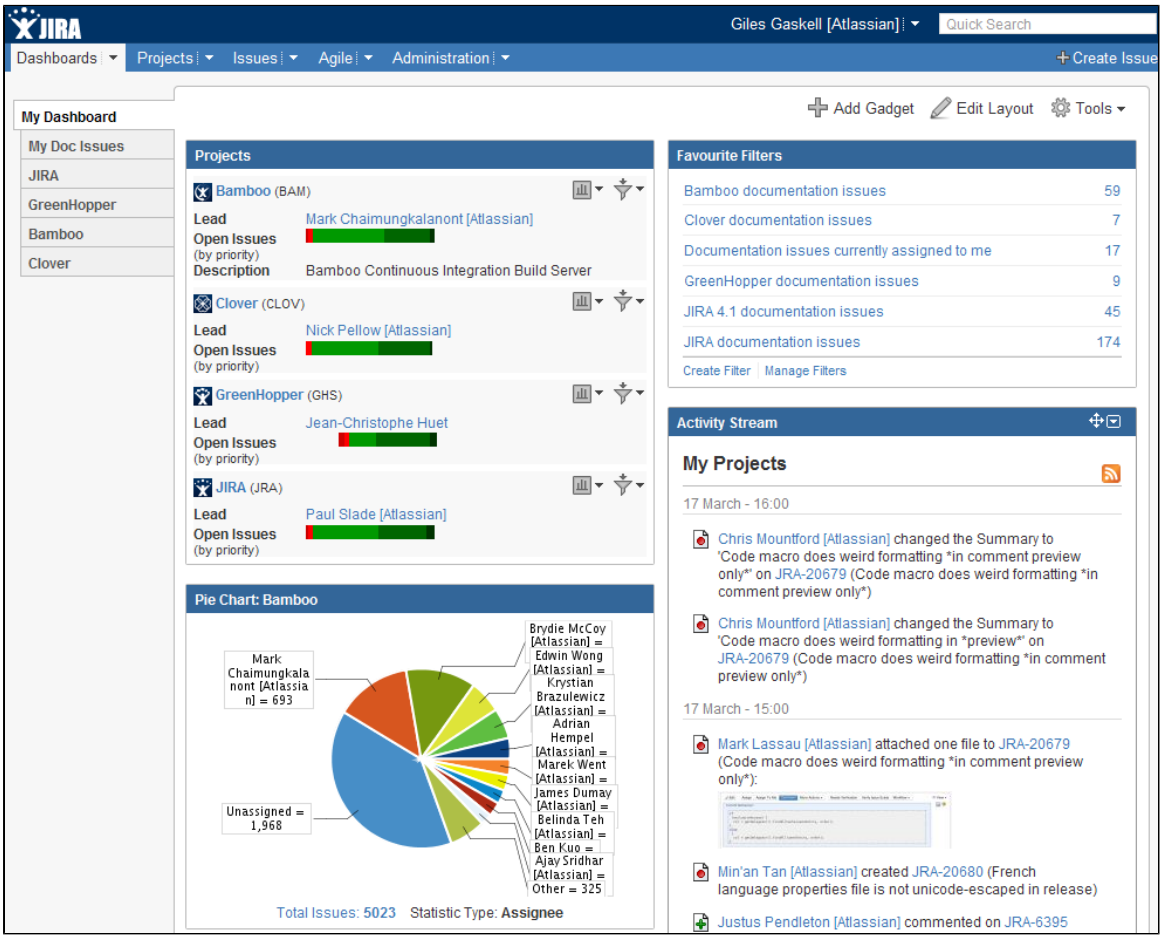
- About Dashboards and Gadgets
  - Available Gadgets
- Creating a Dashboard

About Dashboards and Gadgets

The **JIRA Dashboards** is the first screen you see when you log in to JIRA. It can be configured to display many different types of information, depending on your areas of interest.

If you are anywhere else in JIRA, you can access your JIRA Dashboards view by clicking the '**Dashboards**' link in the top left corner of the JIRA interface.

The information boxes on the dashboard are called **Gadgets**:



If your user account has only one dashboard, the tabs on the left of the browser window will not be available and the dashboard will occupy the full window width.

You can easily customise your dashboard by choosing a different layout, adding more gadgets, dragging the gadgets into different positions, and changing the look of individual gadgets.

You can also create more pages for your dashboard, share your pages with other people and choose your favourites pages, as described in Managing Multiple Dashboard Pages. Each page can be configured independently, as per the instructions below.

Available Gadgets

Gadget	Description
Activity Stream Gadget	The **Activity Stream** gadget displays a summary of your recent activity.
Administration Gadget	The **Administration** gadget displays quick links to administrative functions.
Assigned To Me Gadget	The **Assigned To Me** gadget displays all open issues in all projects assigned to the current user viewing the dashboard.
Average Age Gadget	The **Average Age** gadget displays a bar chart showing the average number of days that issues have been unresolved.
Bamboo Charts Gadget *	The **Bamboo Charts** gadget displays various charts and plan statistics from a particular Bamboo server.
Bamboo Plan Summary Chart Gadget *	The **Bamboo Plan Summary** gadget displays a graphical summary of a build plan.
Bamboo Plans Gadget *	The **Bamboo Plans** gadget displays a list of all plans on a Bamboo server, and each plan's current status.
Bugzilla ID Search Gadget	The **Bugzilla ID Search** gadget allows the user to search all JIRA issues for references to Bugzilla IDs.
Calendar Gadget *	The **Issue Calendar** gadget shows issues and versions in a calendar format based on their due date. Calendars can be based on an issue filter or on a project.
Clover Coverage Gadget *	The **Clover Coverage** gadget displays the Clover coverage of plans from a particular Bamboo server.
Created vs Resolved Gadget	The **Created vs Resolved** gadget displays a difference chart showing the issues created vs resolved over a given period.
Crucible Charts Gadget *	The **Crucible Charts** gadget displays various charts showing statistical summaries of code reviews.
Favourite Filters Gadget	The **Favourite Filters** gadget displays a list of all the issue filters that have currently been added by you as a favourite filter.
Filter Results Gadget	The **Filter Results** gadget displays the results of a specified issue filter.
FishEye Charts Gadget *	The **FishEye Charts** gadget displays two charts showing showing statistics about a given sourcecode repository.
FishEye Recent Changesets Gadget *	The **FishEye Recent Changesets** gadget displays a number of recent changesets from a FishEye repository.
In Progress Gadget	The **In Progress** gadget displays all issues that are currently in progress and assigned to the current user viewing the dashboard.
Introduction Gadget	The **Introduction** gadget displays a configurable introduction message on the dashboard.

Issue Statistics Gadget	The **Issue Statistics** gadget displays the collection of issues returned from a specified filter, broken down by a specified field.
JIRA: News Gadget	The **JIRA:News** gadget displays recent Atlassian news about JIRA.
Pie Chart Gadget	The **Pie Chart** gadget displays issues from a project or issue filter, grouped by a statistic type, in pie-chart format. The issues can be grouped by any statistic type (e.g. Status, Priority, Assignee, etc).
Projects Gadget	The **Projects** gadget provides information and various filters related to a specified project(s).
Quick Links Gadget	The **Quick Links** gadget displays a number of useful links to issues associated with the current user.
Recently Created Issues Gadget	The **Recently Created Issues** gadget displays a bar chart showing the rate at which issues are being created, as well as how many of those created issues are resolved.
Resolution Time Gadget	The **Resolution Time** gadget displays a bar chart showing the average resolution time (in days) of resolved issues.
Road Map Gadget	The **Road Map** gadget shows versions which are due for release within a specified period of time, and a summary of progress made towards completing the issues in those versions.
Text Gadget *	The **Text** gadget displays a configurable HTML text on the dashboard.
Time Since Issues Gadget	The **Time Since Issues** gadget displays a bar chart showing the number of issues that something has happened to within a given time period. The 'something has happened' is based on a date field that you choose, such as 'Created', 'Updated', 'Due', 'Resolved' or a custom field.
Two Dimensional Filter Statistics Gadget	The **Two Dimensional Filter Statistics** gadget displays statistical data based on a specified filter in a configurable table format.
Voted Gadget	The **Voted Issues** gadget shows issues for which you have voted.
Watched Gadget	The **Watched Issues** gadget shows issues which you are watching.

See the big list of all Atlassian gadgets for more ideas.

* *This gadget will only be available if it has been installed by your JIRA administrator.*

The Firebug add-on for Firefox can significantly degrade the performance of web pages. If JIRA is running too slowly (the JIRA dashboard, in particular) then we recommend that you disable Firebug. Read this FAQ for instructions.

Creating a Dashboard

The dashboard that you see when you first start using JIRA is a "default" dashboard that has been configured by your JIRA administrator. You cannot edit the default dashboard; but you can easily create your own dashboard, which you can then customise as you wish.

To create your own dashboard:

1. At the top right of the Dashboard, click the '**Tools**' menu.
2. Select either '**Create Dashboard**' to create a blank dashboard, or '**Copy Dashboard**' to create a copy of the dashboard you are currently viewing.

You can now customise your dashboard as follows:

- Choosing a Dashboard Layout
- Adding a Gadget

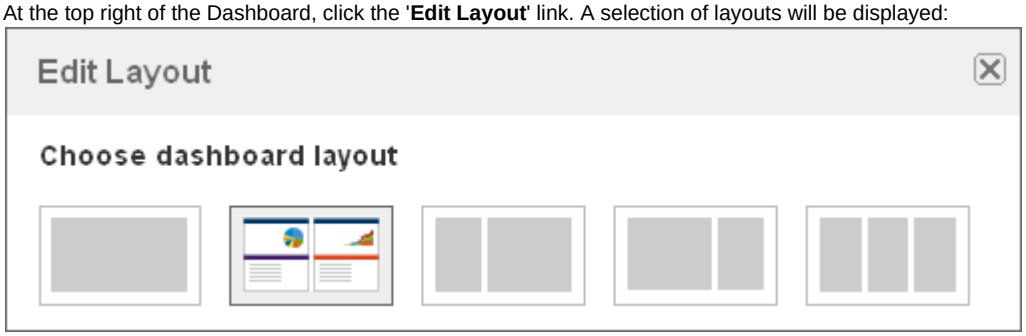
- Moving a Gadget
- Removing a Gadget

If you are using multiple dashboard pages, you can only configure dashboard pages that you own.

Choosing a Dashboard Layout

To choose a different layout for your dashboard page (e.g. three columns instead of two):

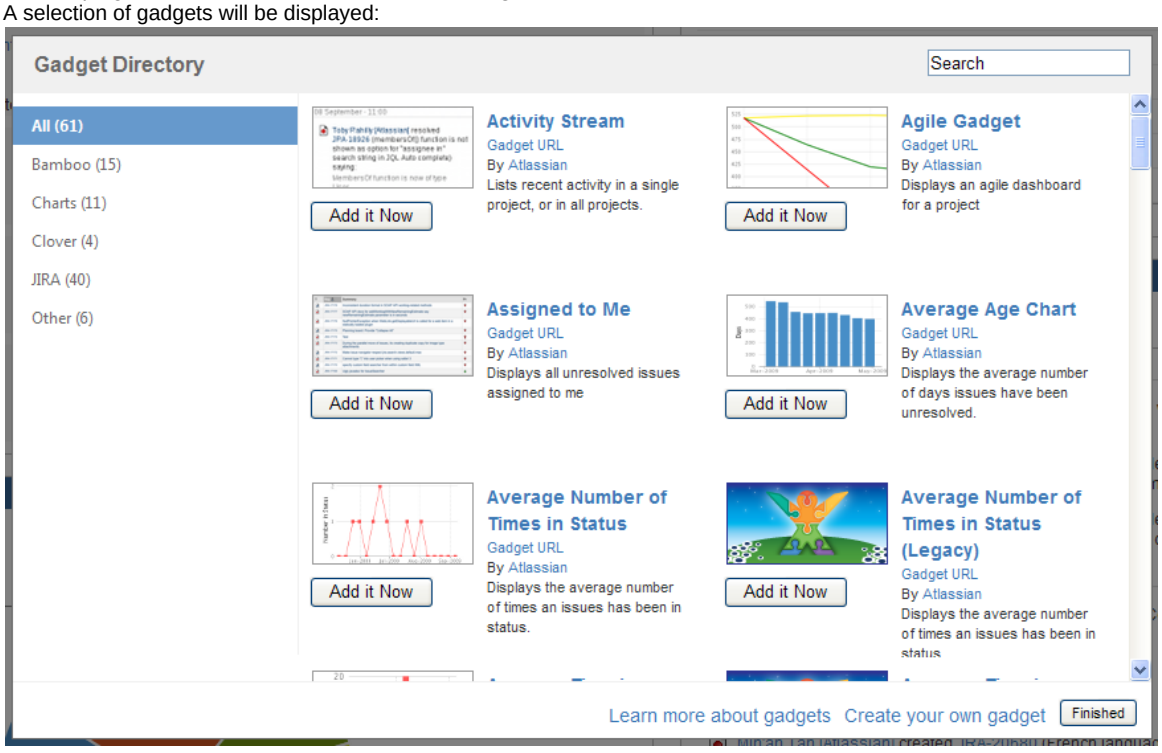
1. At the top right of the Dashboard, click the '**Edit Layout**' link. A selection of layouts will be displayed:



2. Click your preferred layout.

Adding a Gadget

1. At the top right of the Dashboard, click the '**Add Gadget**' link.
2. A selection of gadgets will be displayed:



Select a category on the left to restrict the list of gadgets on the right to that category.

3. Click the '**Add it now**' button beneath your chosen gadget.
4. Click the '**Finished**' button to return to your Dashboard.
5. If the gadget you have selected requires configuration, you will be presented with the gadget's configuration page. Configure appropriately and click '**Save**'.

Moving a Gadget

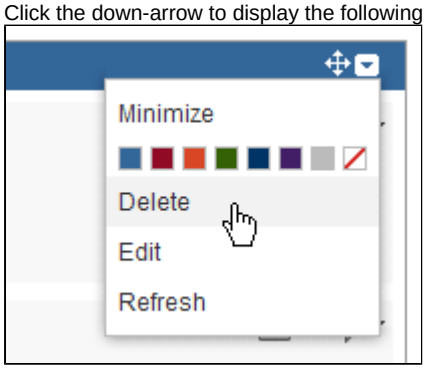
To move a gadget to a different position on your dashboard:

- Click the gadget and drag it into its new position.

Removing a Gadget

To remove a gadget from your dashboard:

1. Hold your mouse over the top right corner of the gadget, until a down-arrow appears.
2. Click the down-arrow to display the following menu:



3. Click '**Delete**'.

RELATED TOPICS

The big list of Atlassian gadgets
The JIRA Wallboards plugin

Managing Multiple Dashboard Pages

JIRA allows you to configure more than one dashboard page. Each dashboard page can be configured independently, allowing you to neatly organise related information by context. You can also share your dashboard pages with other users, as well as adding dashboards shared by other users as favourites.

Each dashboard page appears as a separate "tab" on the dashboard. You can view a dashboard page by simply clicking its name in the tab list.

On this page:

- Managing your Dashboard
- Creating new dashboard pages
- Displaying a dashboard page on your dashboard ('Favourite Dashboards')
- Sharing Dashboard Pages
- Finding an existing Dashboard Page
- Editing an existing Dashboard Page's details
- Copying an existing Dashboard Page
- Deleting an existing Dashboard Page
- Managing Other User's Shared Dashboards

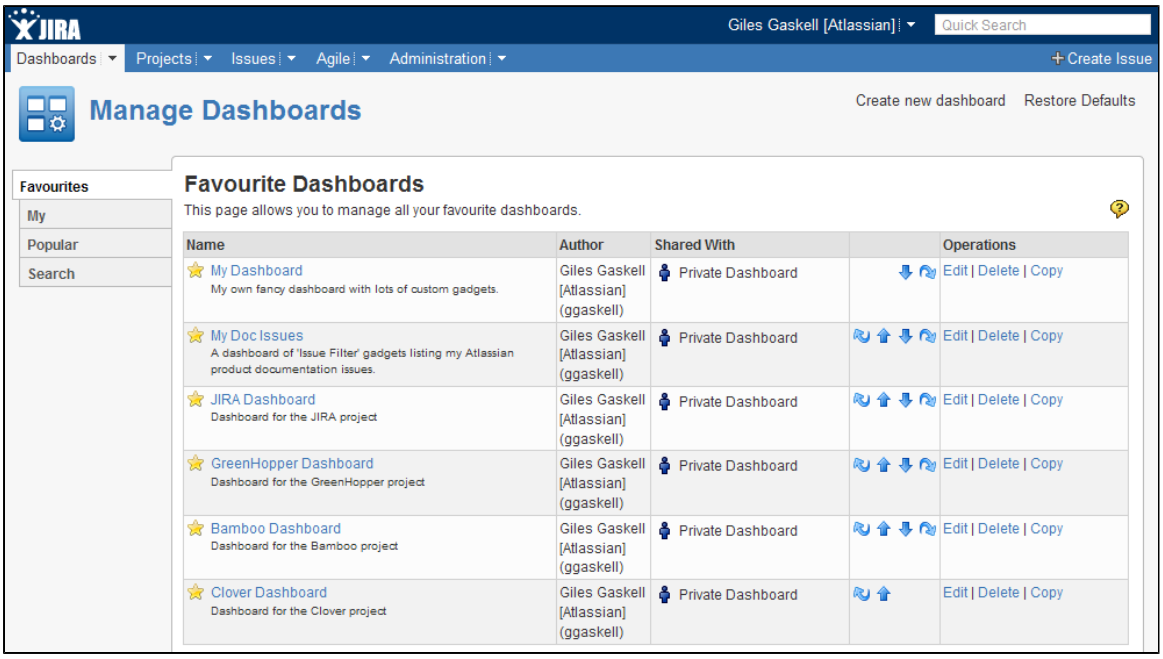
Managing your Dashboard

The '**Manage Dashboards**' page allows you to view and configure dashboard pages that you have created, as well as work with dashboard pages that other users have shared with you.

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. From this page, you can:
    - Create a new dashboard page.

- Add a dashboard page as a favourite.
- Share a dashboard page that you have created, with other users.
- Search for dashboard pages that has been created by you or shared with you by other users.
- Configure an existing dashboard or edit an existing dashboard's details of a dashboard that you have created.
- Copy a dashboard page that has been created by you or shared with you by other users.
- Delete a dashboard page that you have created.

Click the above links for further details on each function.



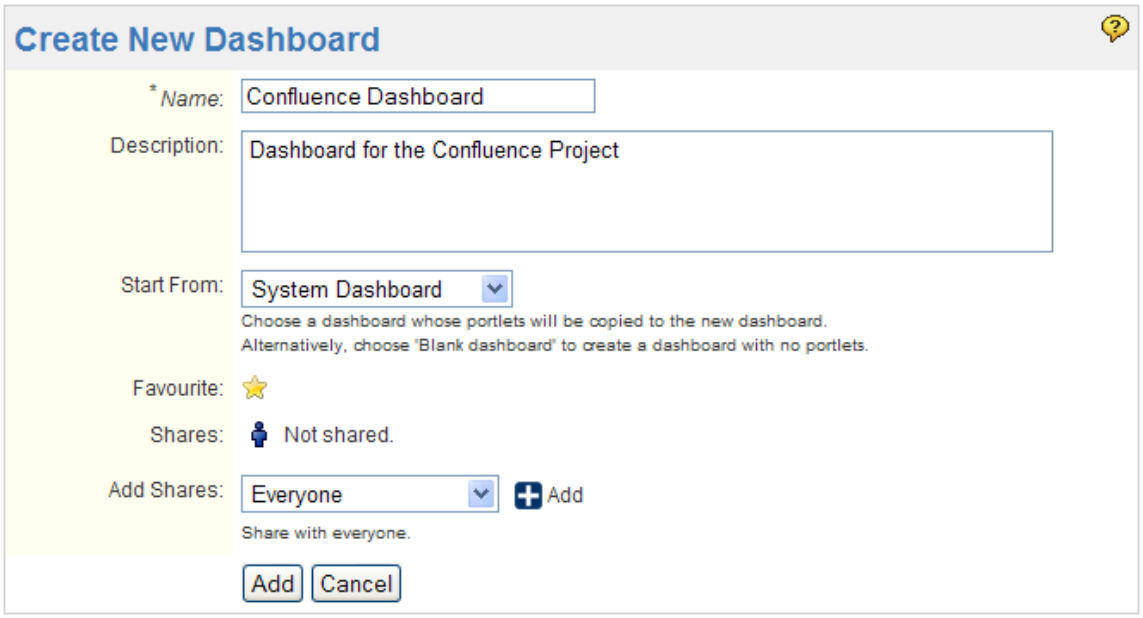
You can also re-order your dashboard pages on this page, by using the arrow icons:

Move a dashboard up	Click the up arrow ( ) for the dashboard that you wish to move.
Move a dashboard down	Click the down arrow ( ) for the dashboard that you wish to move.
Move a dashboard to the top of the list	Click the curly up arrow ( ) for the dashboard that you wish to move.
Move a dashboard to the bottom of the list	Click the curly down arrow ( ) for the dashboard that you wish to move.

Creating new dashboard pages

To create a new dashboard page please follow these steps:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu. The 'Manage Dashboards' page will display. This page lists all currently configured dashboard pages.
2. Click the '**Create new dashboard**' link. The 'Create new dashboard' page will display.



Provide a name for the new dashboard page and optionally enter a short description. You can also choose an existing page as a starting point for the new page. This means that the configuration of the existing page will be duplicated for the newly created page. Alternatively, if you would like to create a page with no gadgets, leave the 'Blank dashboard' option selected.

3. Your new dashboard page will be added as a 'favourite' dashboard page by default upon creation, which means that it will display as a tab on your JIRA dashboard. If you do not wish to display this dashboard page as a tab on your JIRA dashboard, deselect the star icon. You can add the dashboard page as a favourite after it has been created. Read more about adding an existing dashboard page as a favourite.
4. The sharing of your new dashboard page depends on your sharing preference in your user profile. If you have not specified a personal preference, then the global default for sharing will apply (i.e. 'Private', unless changed by your JIRA Administrator under 'User Defaults' in the Administration menu). If you wish to change the sharing of your dashboard page, refer to the instructions on sharing dashboard pages below.

> Please note, you need the 'Create Shared Object' global permission to be able to share your dashboard page. If you cannot see any dashboard sharing functionality, contact your JIRA Administrator to be granted this permission.

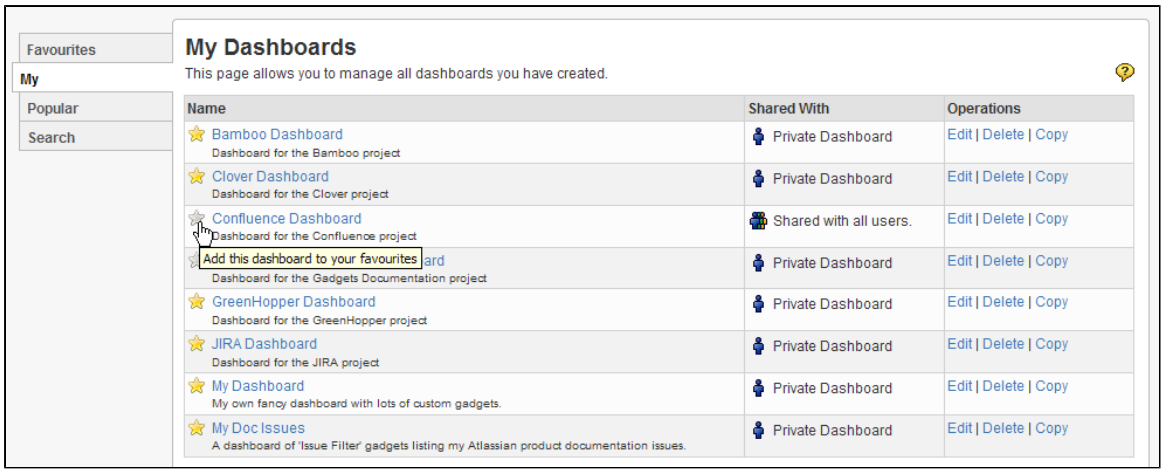
5. Click the '**Add**' button. Your new page will be listed under the 'My' tab of the 'Manage Dashboards' page. If you selected the new dashboard page as a favourite, it will also appear under the 'Favourite' tab and will be displayed as a tab on your JIRA dashboard.
6. You can now customise your new dashboard page, and add gadgets to it, as described in Customising the Dashboard.

Displaying a dashboard page on your dashboard ('Favourite Dashboards')

Dashboard pages that you have created, or that have been shared by other people, can be added as a 'favourite'. This means that the dashboard page will appear as a tab on the left side of your browser window, when viewing your JIRA dashboards. There is no restriction on the number of dashboards that you can add as a 'favourite' and each of these will appear on an individual tab when viewing your JIRA dashboards.

To add an existing dashboard page to your dashboard:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. Locate the dashboard page that you wish to display on your dashboard. If you created the dashboard, it will be listed under the 'My' tab, otherwise you can searchfor dashboards shared by other users via the 'Search' tab.
    - Your favourite dashboards are shown with a yellow star.
    - Dashboards that are not currently your favourites are shown with a grey star.
3. Click the grey star icon next to the name of the desired dashboard page to add it as a favourite. The dashboard page will be displayed on your main dashboard.



Please note, if you have added another user's shared dashboard as a favourite and a gadget(s) is not displaying correctly, the gadget(s) may be using an issue filter that is not shared with you. You will need to contact the author of the issue filter to change the filter sharing.

To remove a dashboard page from your dashboard:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. Locate the dashboard page that you wish to remove from your dashboard under the 'Favourites' tab.
3. Click the star icon next to the name of the dashboard page. The dashboard page will be removed from your main dashboard.

Please note, if you do not have any dashboard pages added as favourites, the default dashboard will be displayed on your dashboard with an error message. You can choose to keep the default dashboard displayed on your dashboard, but you will need to add it as a favourite to stop the error message from showing. You may need to search for the 'System Default' dashboard to add it as a favourite.

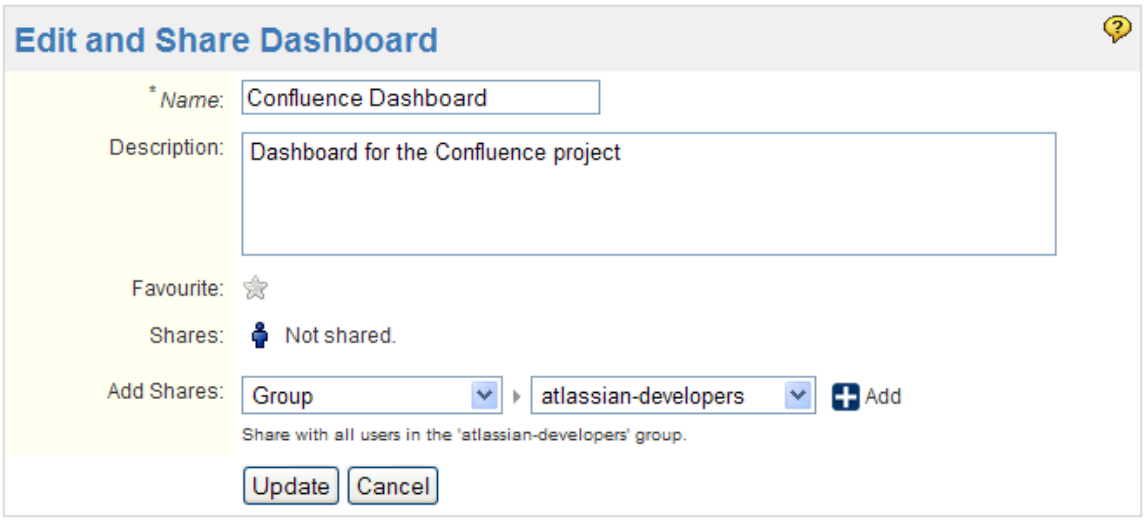
Sharing Dashboard Pages

JIRA also allows you to share any dashboard pages that you have configured. Dashboard pages can be shared with other users via user groups, projects and project roles. Dashboard pages can also be shared globally. Sharing a dashboard page allows other users to display it on their JIRA dashboard, by selecting it as a favourite.

Please note, you may need to review the sharing permissions for any issue filters used in portlets on your shared dashboard. If another user adds your dashboard as a favourite, but cannot access a filter for a portlet, then the portlet will display with an error message.

To share an existing dashboard page to the dashboard, please follow these steps:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. Locate the dashboard page that you wish to display on your dashboard under the 'My' tab and click the '**Edit**' link for the dashboard in the '**Operations**' column.
3. The 'Edit Dashboard' page will display. Select the group, project or project role that you want to share the dashboard with, or share it with all users, if you wish. Click the '**Add**' link to add the share. You can add further share permissions if you wish.
4. Click the '**Update**' button to save your changes.



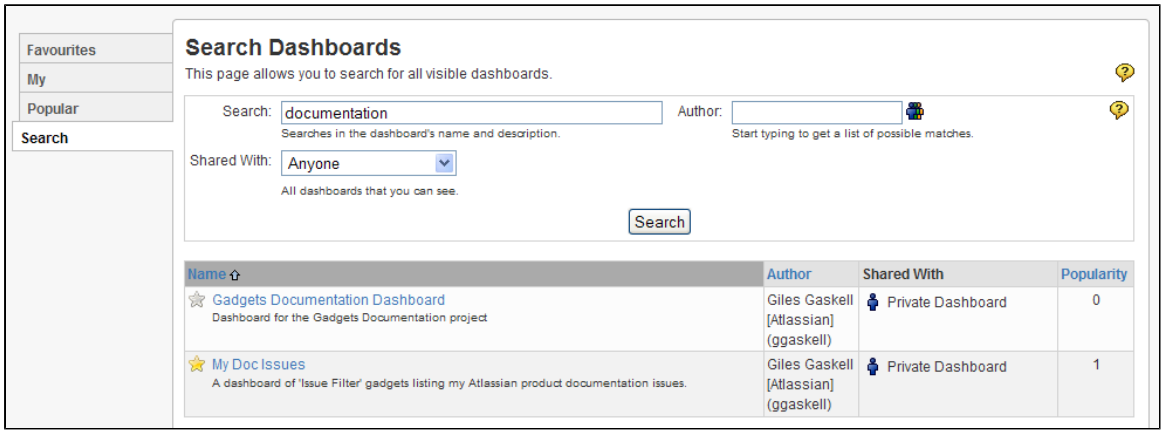
Please note, you be able to share your dashboard only with the groups that you are member of.

Finding an existing Dashboard Page

Dashboard pages that you have created or have been shared by other users, can be found via the dashboard Search function of the 'Manage Dashboards' page. If it is a popular dashboard (i.e. added as a favourite by many users), you can also locate it on the 'Popular' tab of the 'Manage Dashboards' page which lists the top twenty most popular dashboards.

To search for an existing dashboard page, please follow the steps below:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. Click the '**Search**' tab. The dashboard Search will display. Enter your search criteria and click '**Search**' to run the search.
3. Your search results will be displayed on the same page. You can sort the search results by any of the columns, by clicking the column headers. Click the name of any dashboard page to temporarily display it on your dashboard (i.e. it will be removed from your dashboard when you navigate away). To keep the dashboard page as a tab on your dashboard, click the '**add it as a favourite**' link.

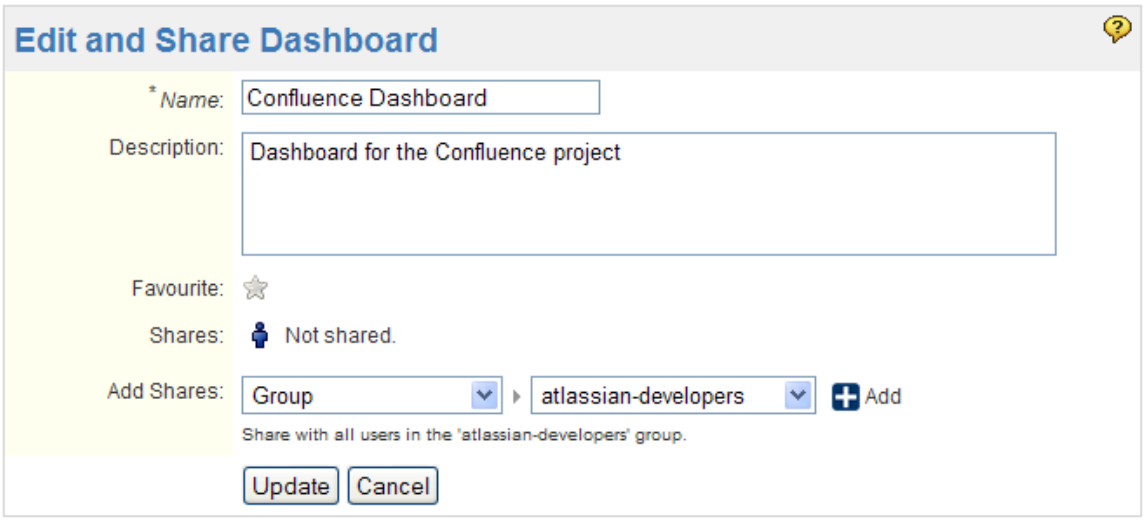


Editing an existing Dashboard Page's details

You can always update the details, i.e. Name, Description, Sharing, Favourite, of an existing dashboard page after its creation. Please note that you can only update the details of dashboard pages which you have created.

To update the details of one of your existing dashboard pages, please follow the steps below:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. Locate the dashboard page that you wish to update and click the '**Edit**' link for the dashboard in the 'Operations' column.
3. The 'Edit Dashboard' page will display. Update the details of the dashboard page as desired. If you wish to change the sharing or favourite settings for the dashboard page, refer to the relevant instructions above.



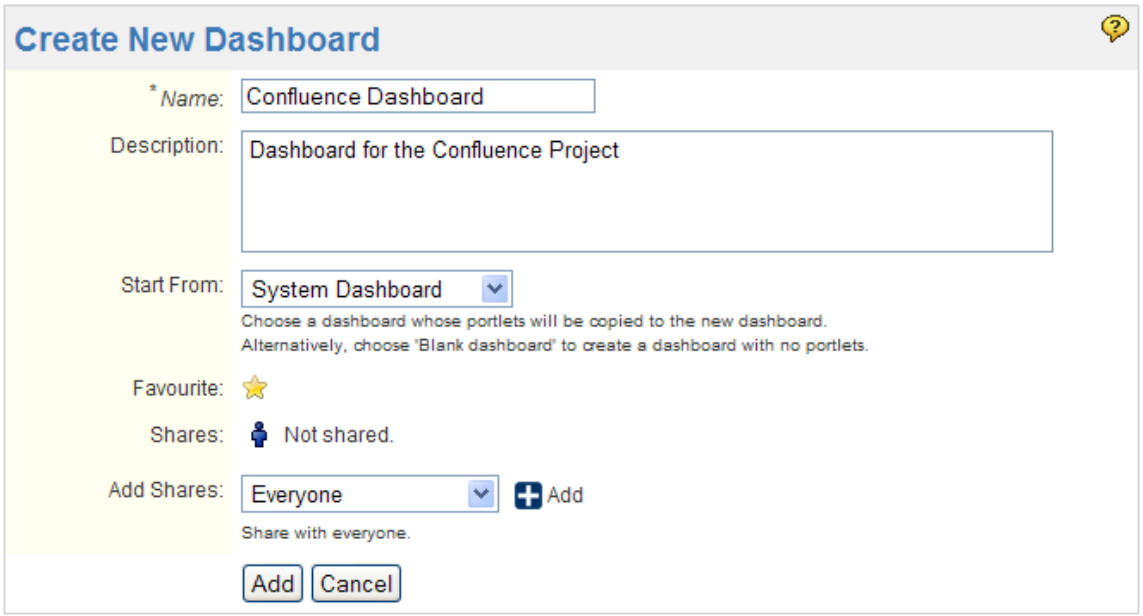
4. Click the '**Update**' button to save your changes.

Copying an existing Dashboard Page

You can make a copy of an existing dashboard page (created by you or shared with you), which creates a new dashboard page with the same gadget configuration as the existing dashboard page.

To update the details of one of your existing dashboard pages, please follow the steps below:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. Locate the dashboard page that you wish to copy and click the '**Copy**' link for the dashboard in the 'Operations' column.
3. The 'Create New Dashboard' page will display. Update the details of the dashboard page as desired. If you wish to change the sharing or favourite settings for the dashboard page, refer to the relevant instructions above.



4. Click the '**Add**' button to save your changes.

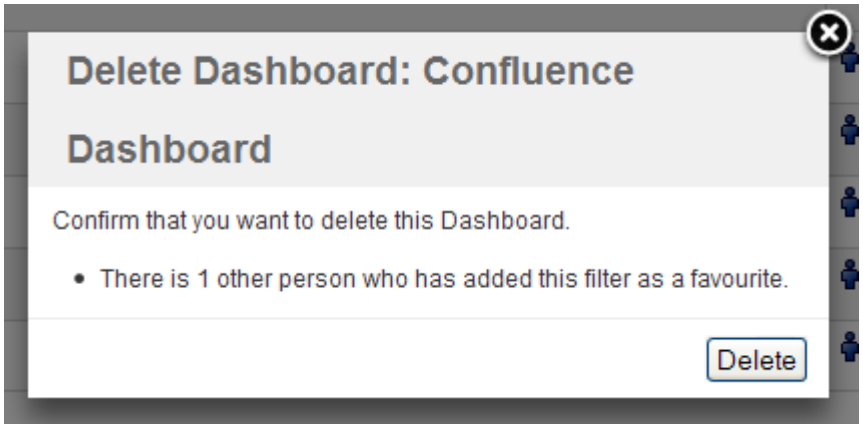
Deleting an existing Dashboard Page

Please note that you can only delete dashboard pages that you created.

To delete a dashboard pages, please follow the steps below:

1. At the top left of the dashboard, click the down-arrow on the '**Dashboards**' tab and select '**Manage Dashboard**' from the drop-down menu.
2. The 'Manage Dashboards' page will display. Locate the dashboard page that you wish to copy and click the '**Delete**' link for the

dashboard in the 'Operations' column.
3. A confirmation message box will appear. This message will also inform you if (and how many) other users have selected this dashboard as a favourite. If you wish to continue with the deletion, click the '**Delete**' button. Otherwise, click the '**x**' in the top right of the message box to cancel this action.



Be aware that deleting a dashboard which other users have marked as a favourite will prevent these users from accessing that dashboard in future.

Managing Other User's Shared Dashboards

A **shared dashboard** is a dashboard whose creator has shared that dashboard with other users. Refer to Sharing Dashboard Pages above for details. When a shared dashboard is created by a user, that user:

- Initially 'owns' the shared dashboard.
- Being the owner, can edit and modify the shared dashboard.

If you have the **'JIRA Administrators'** global permission, you can manage shared dashboards that were created by other users.

To access the 'Shared Dashboards' feature:

1. Ensure that you are logged in as a user with the **JIRA Administrators** global permission.
2. On the top navigation bar, click the '**Dashboards**' dropdown and select '**Shared Dashboards**' from the list.
3. See Managing Shared Dashboards in the JIRA Administrator's Guide.

Changing the Look and Behaviour of a Gadget

On this page:

- Hiding or Changing the Colour of the Gadget's Frame
- Minimising and Expanding a Gadget
- Opening the Maximised or Canvas View of a Gadget
- Editing a Gadget's Settings

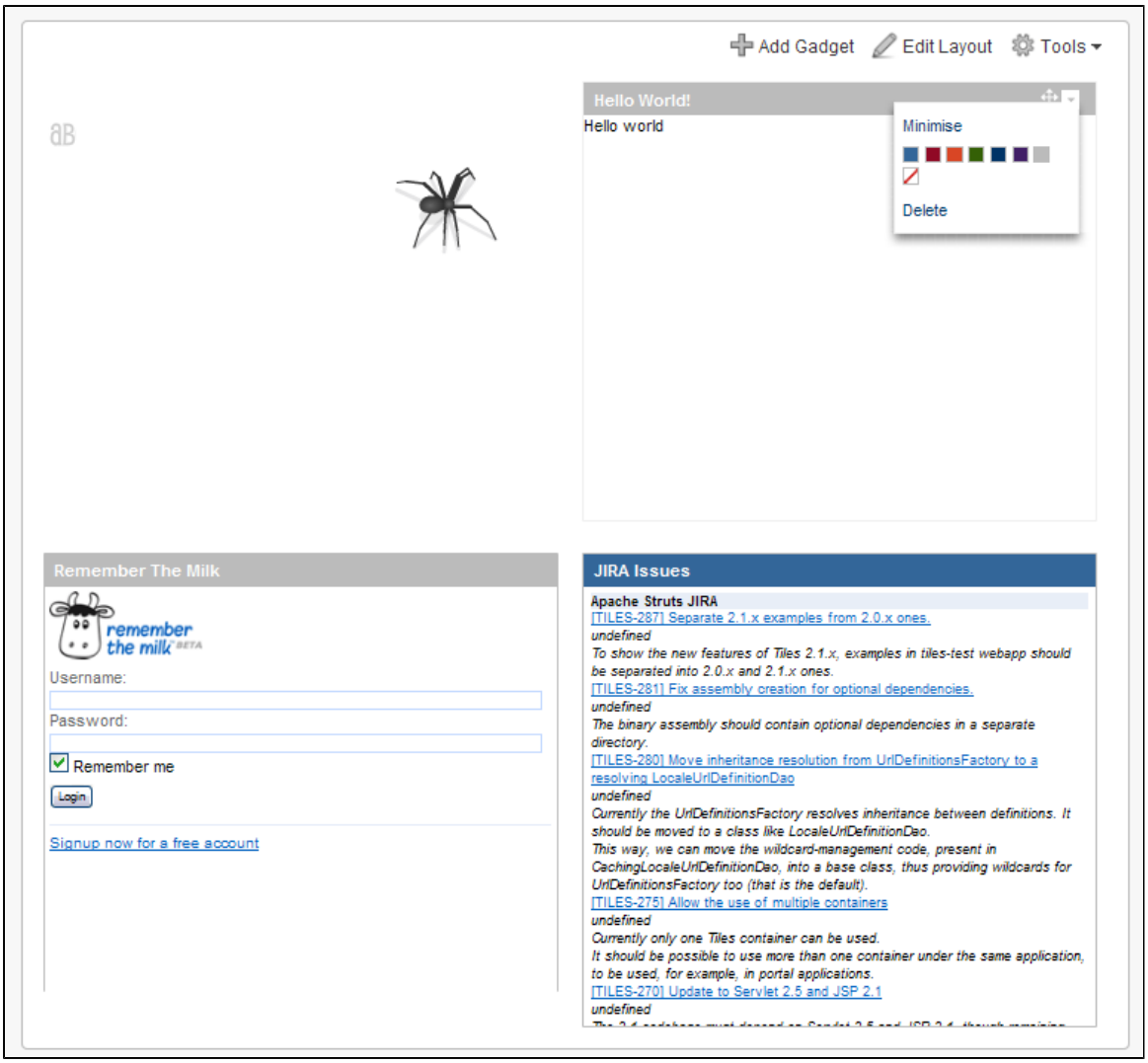
Hiding or Changing the Colour of the Gadget's Frame

You can change the colour of the frame surrounding a gadget on your dashboard. You can even hide the gadget's frame altogether, so that it only shows when you move your mouse pointer over the gadget. In the screenshot below, the top two gadgets have hidden frames. The frame for the top gadget on the left is not visible. The frame for the top gadget on the right is currently visible because the mouse pointer is hovering over the gadget.

To hide or change the colour of a gadget's frame,

1. Go to the dashboard by clicking the '**Dashboard**' link or the '**Home**' link at the top left of the screen.
2. The dashboard will appear, looking something like the screenshot below. Move your mouse pointer over the gadget you want to change. If the gadget's frame is hidden, the frame will appear now.
3. Click the dropdown menu icon at top right of the gadget frame.
4. The dropdown menu will appear, as shown in the screenshot below. Click the colour you want for your gadget's frame. To hide the gadget's frame, select the white colour box with the red line through it.

Screenshot: Hiding or changing the colour of a gadget's frame



Minimising and Expanding a Gadget

You can shrink (minimise) a gadget on your dashboard so that it displays only the top bar of the gadget frame. In the screenshot below, the top left gadget ('**Spider**') has been minimised.

- If you minimise a gadget that has a hidden frame, the gadget will not be visible on the dashboard until you move your mouse pointer over the gadget. See the section above on hiding or changing the colour of the gadget frame.
- You can minimise/expand a gadget even if you do not have update permissions on the dashboard.
- The minimise/expand setting is stored in a cookie, and is not saved to the dashboard server.

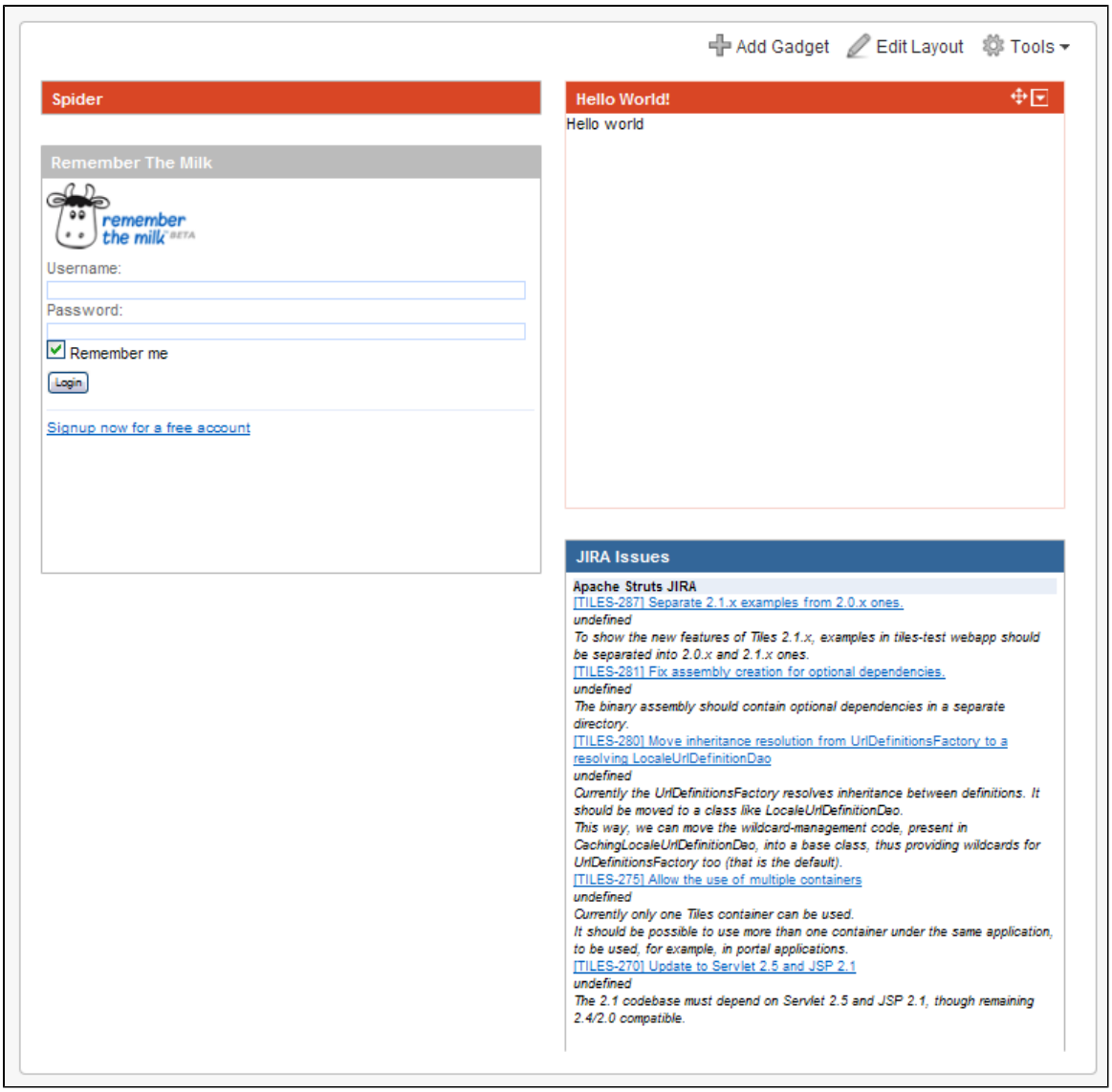
To minimise a gadget,

1. Move your mouse pointer over the gadget you want to change.
2. The gadget menu icons will appear. Click the dropdown menu icon at top right of the gadget frame.
3. The dropdown menu will appear, as shown in the screenshot above. Click '**Minimise**'.

To expand a gadget that has been minimised,

1. Move your mouse pointer over the gadget you want to change.
2. The gadget menu icons will appear. Click the dropdown menu icon at top right of the gadget frame.
3. The dropdown menu will appear. Click '**Expand**'.

Screenshot: A minimised gadget



Opening the Maximised or Canvas View of a Gadget

Some gadgets allow you to expand themselves so that they take up the entire space allowed by the dashboard. This is also known as 'canvas view'.

- The maximised or canvas view of a gadget often provides additional functionality, i.e. more than is available in the standard view of the gadget.
- This is not the same as minimising and then expanding a gadget (see above).
- Only some gadgets provide the maximised or canvas view.
- You can open the canvas view of a gadget even if you do not have update permissions on the dashboard.
- The maximised/canvas view setting is stored in a cookie, and is not saved to the dashboard server.

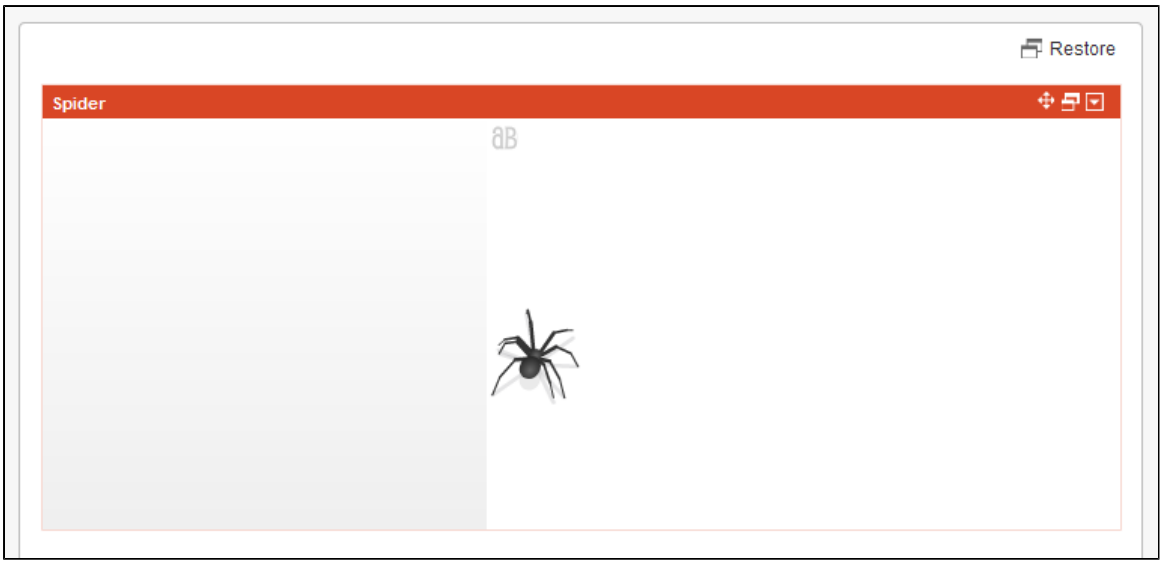
To open the maximised or canvas view of a gadget,

1. Move your mouse pointer over the gadget you want to change.
2. The gadget menu icons will appear. Click the maximise icon at top right of the gadget frame. This icon will appear only if the gadget provides a maximised or canvas view.
3. The gadget's maximised view will open, as shown in the screenshot below.

To close the canvas view and return to your dashboard,

1. Click the '**Restore**' option at the top right of the screen, or the '**Restore**' icon at top right of the gadget frame.

Screenshot: The maximised or canvas view of a gadget



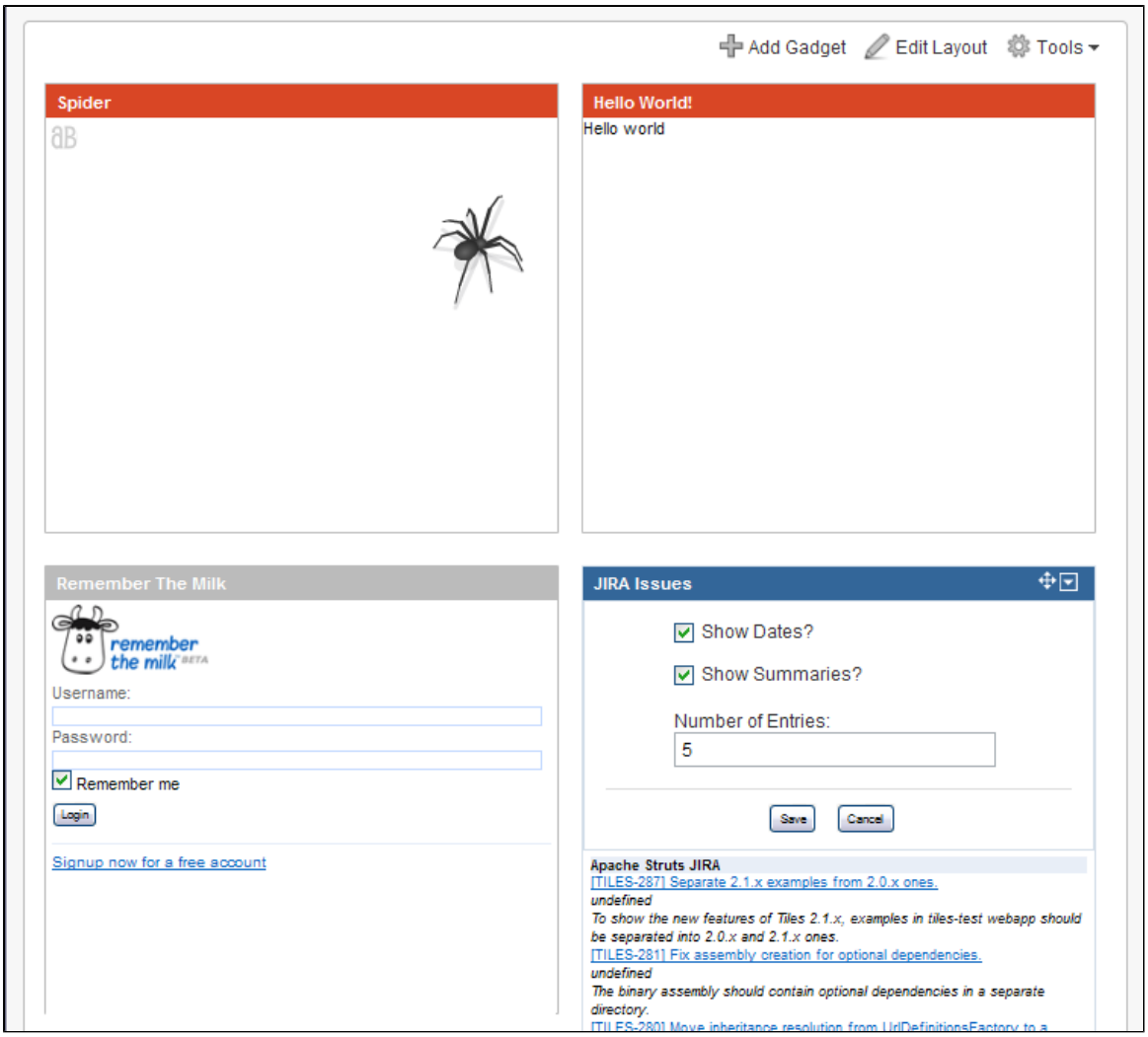
Editing a Gadget's Settings

Some gadgets provide specific properties or settings that you can edit. These settings will be different for each gadget. For example, a gadget may allow you to customise its welcome message, or to define the server where the gadget will find its information.

To edit a gadget's settings,

1. Move your mouse pointer over the gadget you want to change.
2. The gadget menu icons will appear. Click the dropdown menu icon at top right of the gadget frame.
3. The dropdown menu will appear. Click '**Edit**'.
4. A panel will open, showing the settings offered by the selected gadget. In the screenshot below, the bottom two gadgets have their settings panels open.
5. Adjust the settings as required then click '**Save**'.

Screenshot: Editing a gadget's settings



Adding the Activity Stream Gadget

The **Activity Stream** gadget displays a summary of recent activity in particular projects (and/or by particular people) in which you are interested. This can include:

- Issues in your local JIRA system.
- Issues in another JIRA system (provided your administrator has configured a two-way Application Link).
- Activity from another Atlassian application, such as:
  - document updates (from Confluence)
  - code commits (from FishEye)
  - code reviews (from Crucible)
  - builds (from Bamboo)

    Note that this requires your administrator to configure a two-way Application Link.
- Activity from remote applications (note that your administrator will need to set this up via the REST API or the provider plugin API, or locally via Java).

The **Activity Stream** gadget also provides an RSS feed, allowing you to create very specific RSS feeds of only the information that is most relevant to you. Simply add the **Activity Stream** gadget to your dashboard, specify the people/projects of interest (see instructions below), then click the RSS icon:

What does it look like?

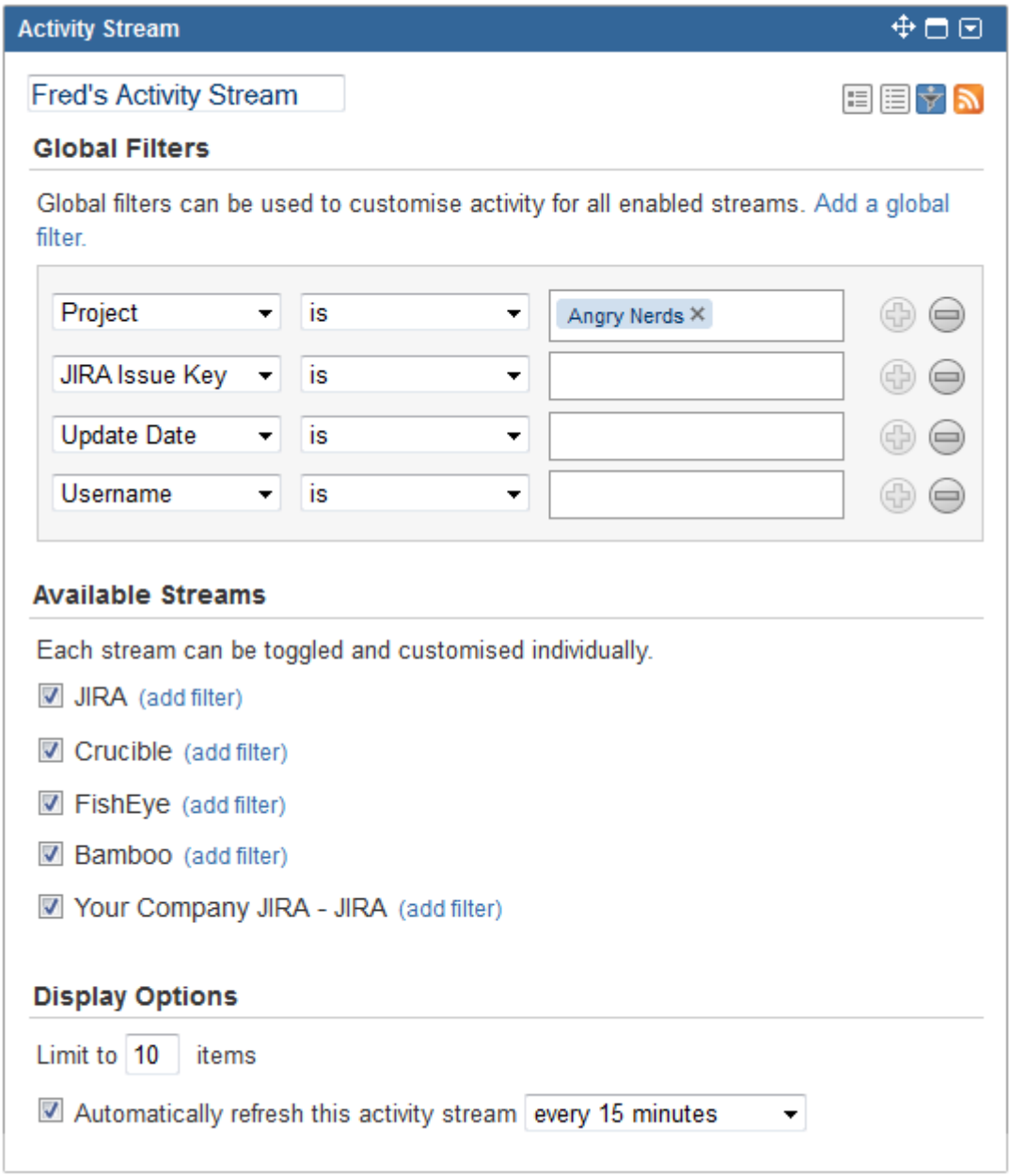
The **Activity Stream** gadget should appear as follows on the dashboard:

Note that you can vote, watch or comment directly on JIRA issues in the activity stream (if you have the appropriate permission), using the links provided.

Adding the 'Activity Stream' gadget to your Dashboard

1. Go to your JIRA dashboard and click **Add Gadget**.

2. The **Gadget Directory** will appear. Locate the **Activity Stream** gadget and click the **Add it Now** button. Then click the **Finished** button at the bottom of the Gadget Directory.
3. The Activity Stream gadget will appear on your dashboard as follows, ready for you to configure:



4. **Title** — type a heading for this gadget.
5. **Apply filters** — by default, the gadget will display all activity for all projects. If you wish to refine this, select the **Apply filters** check-box, then select the '+' signs to filter the activity by:
    - **Project**
    - **JIRA Issue Key**
    - **Update Date**
    - **Username** — the user(s) whose activity you wish to monitor. You can specify multiple usernames delimited by spaces, e.g. "jsmith tjones dbrown".
6. **Available Streams** — select the applications whose activity you wish to monitor. This can include Atlassian applications (e.g. JRA, Confluence) as well as remote applications.
7. **Limit to _ items** — type the number of activities that you want the gadget to display.
8. **Automatically refresh this activity stream** — select this check-box if you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

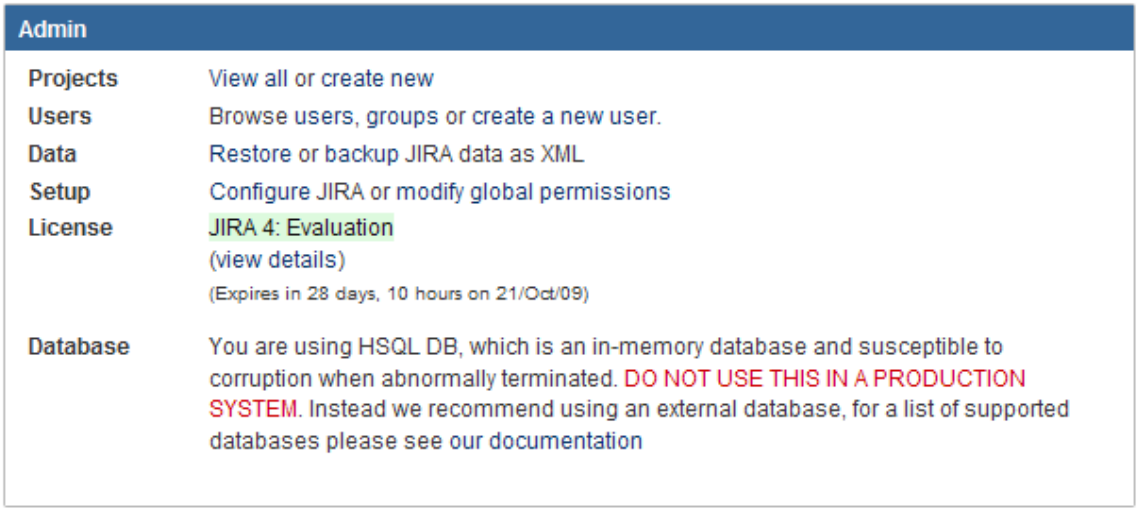
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Administration Gadget

The **Administration** gadget displays quick links to administrative functions conveniently on the dashboard. This gadget is for people who have the **'JIRA Administrators'** or the **'JIRA System Administrators'** global permission.

What does it look like?

The **'Administration'** gadget should appear as follows on the dashboard:



Note: if you only have the **'JIRA Administrators'** permission (and not the **'JIRA System Administrators'** permission), the following words will not be links: 'Restore', 'Backup' and 'License: view details'.

Adding the 'Administration' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Admin**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Assigned To Me Gadget

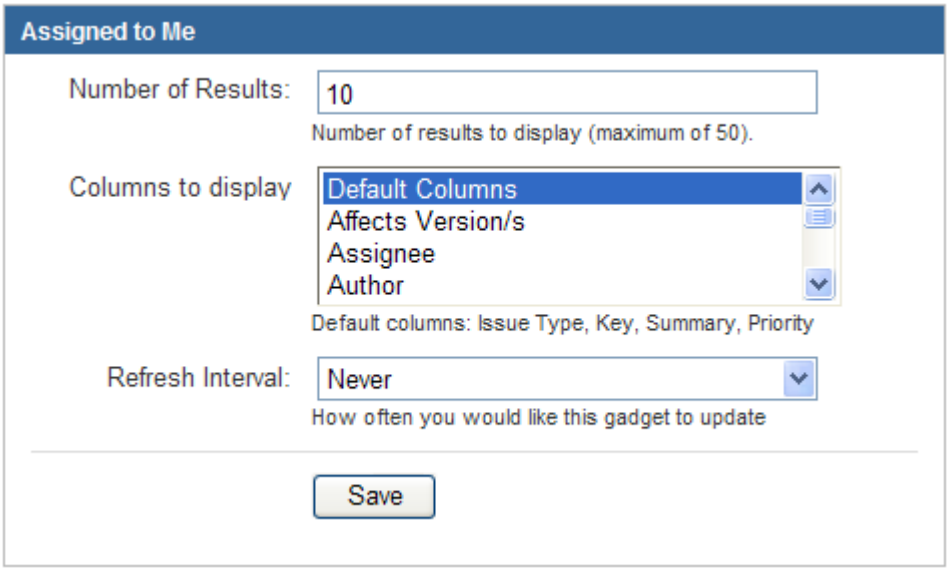
The **Assigned To Me** gadget displays all open issues in all projects assigned to the current user viewing the dashboard.

What does it look like?

The **Assigned to Me** gadget should appear as follows on the dashboard:

Adding the 'Assigned To Me' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Assigned To Me**' gadget and click the '**Add it Now**' button. Then click the '**Finished** ' button at the bottom of the Gadget Directory.
3. The 'Assigned To Me' gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Number of Results**' — type the number of issues you would like the gadget to display per page (maximum 50).
b. '**Columns to display**' — select the column(s) representing the issue fields you would like the gadget to display. The default columns are: Issue Type, Key, Summary, Priority.
c. '**Refresh Interval**' — select how often you want the gadget to update the list of issues (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

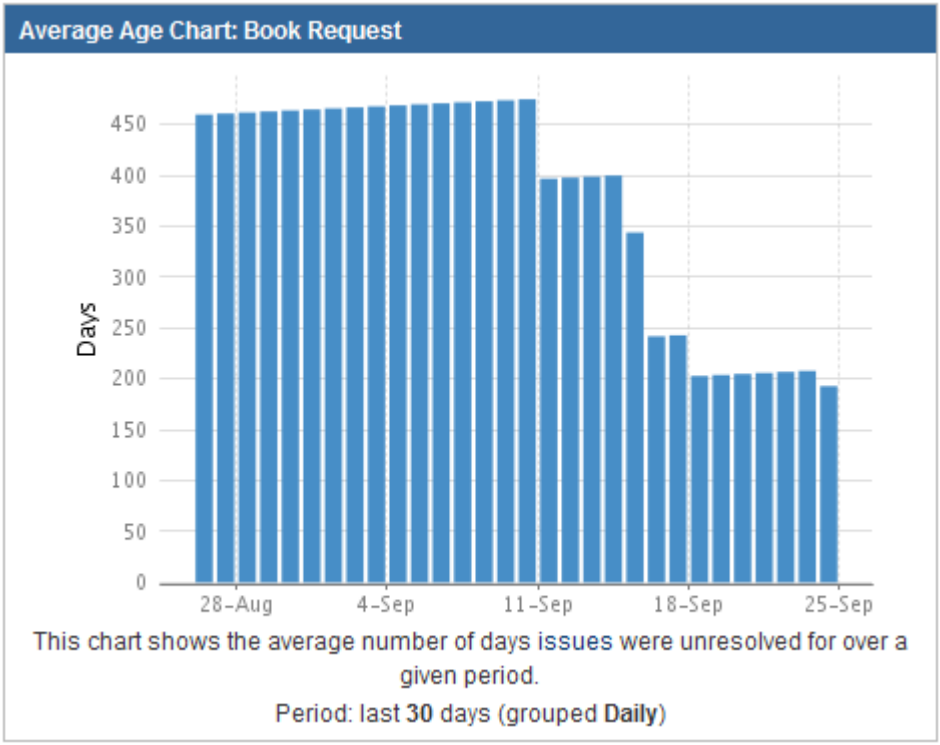
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Average Age Gadget

The **'Average Age'** gadget displays a bar chart showing the average age (in days) of unresolved issues at given points in time. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years). For the purposes of this gadget an issue is defined as unresolved if it has no value in the system Resolution field. The age of an issue is the difference between the current date and the created date of the issue.

What does it look like?

The **'Average Age'** gadget will appear as follows on the dashboard:

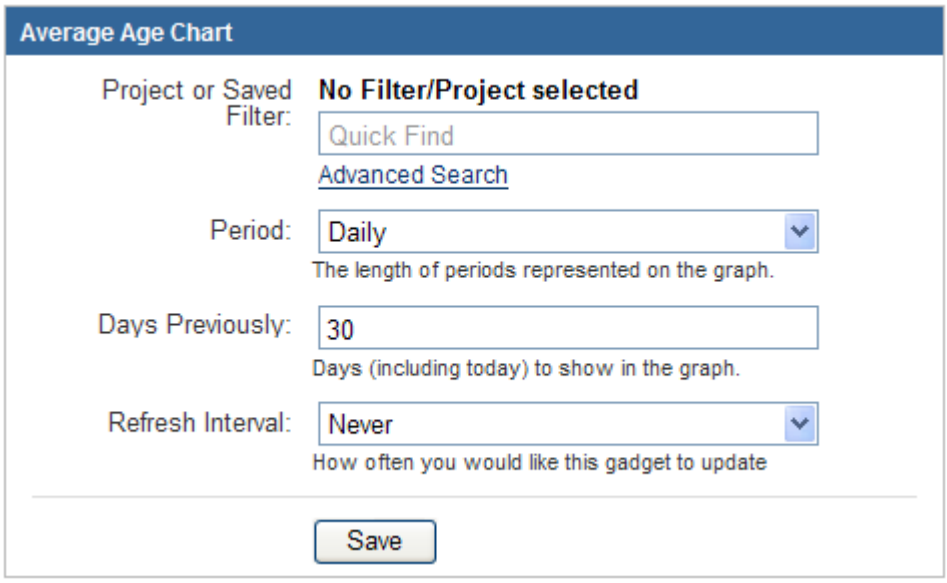


A report showing this information is also available.

Adding the 'Average Age' gadget to your Dashboard

To add the **'Average Age'** gadget to your dashboard:

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Average Age**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Average Age gadget will appear on your dashboard as follows, ready for you to configure:



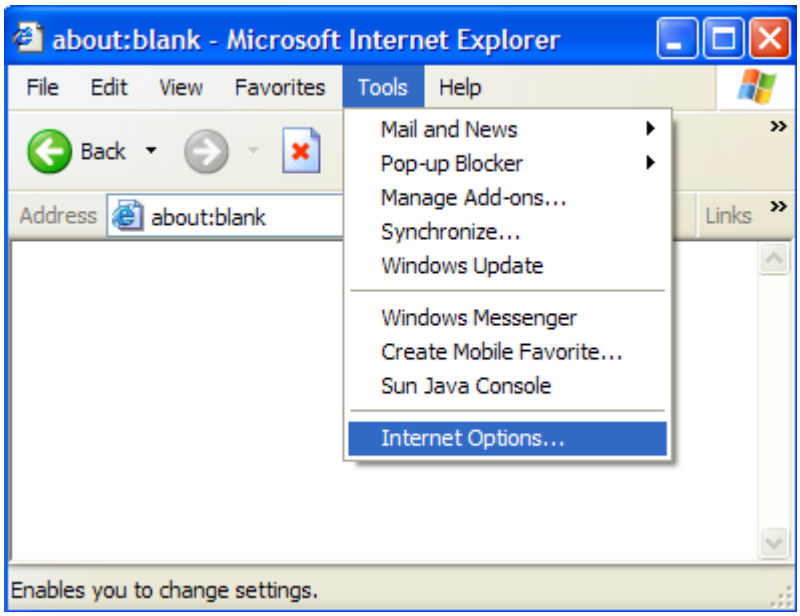
   a. '**Project or Saved Filter**' — start typing the name of the project (or saved filter) on whose issues the chart will be based. Alternatively, if you're unsure of the name of the project or filter you're looking for, click '**Advanced Search**' to search for a project (or saved filter) whose name contains particular text; or a saved filter that was created by a particular user and/or is shared with particular users.
   b. **'Period'** — select the timeframe on which the chart will be based:
      - **'Hourly'**
      - **'Daily'**
      - **'Weekly'**
      - **'Quarterly'**
      - **'Yearly'**
   c. **'Days Previously'** — enter the number of days' worth of data (counting backwards from today) to be included in the chart.
   d. '**Refresh Interval**' — select how often you want the gadget to update the chart (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

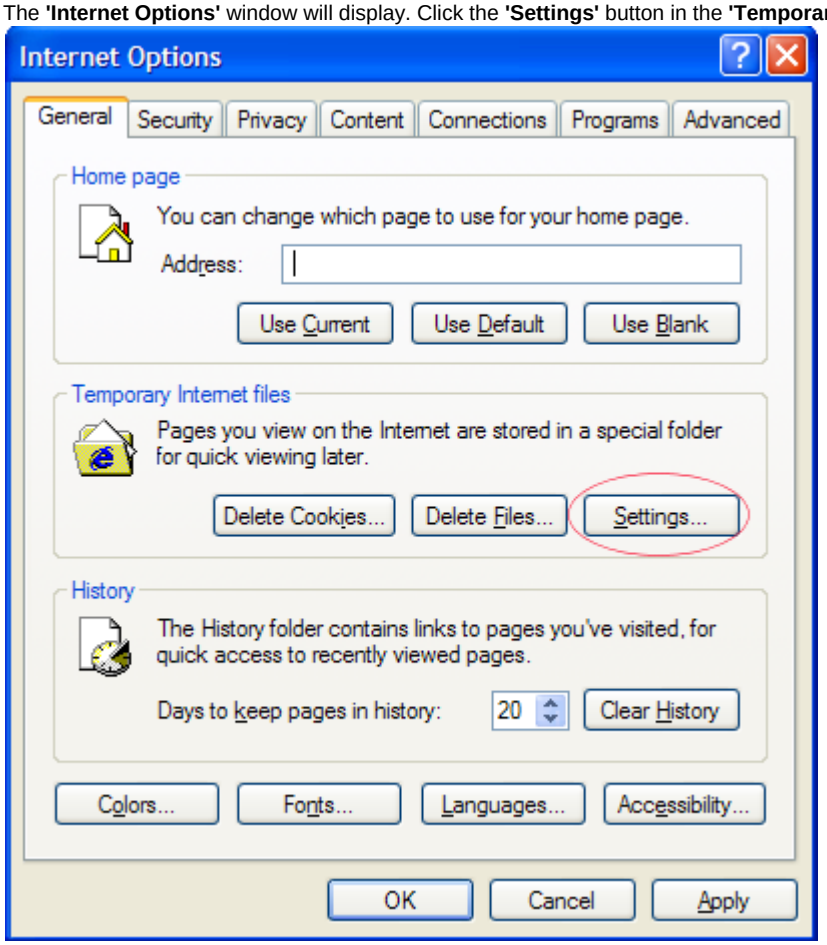
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

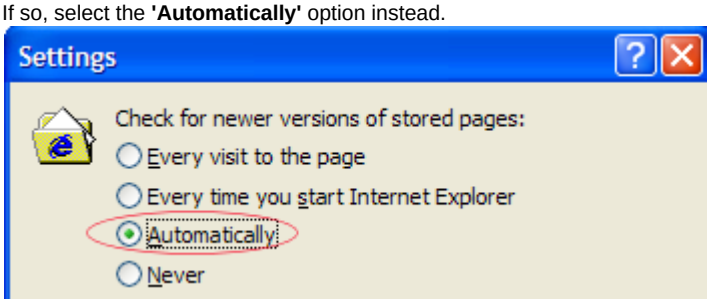
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



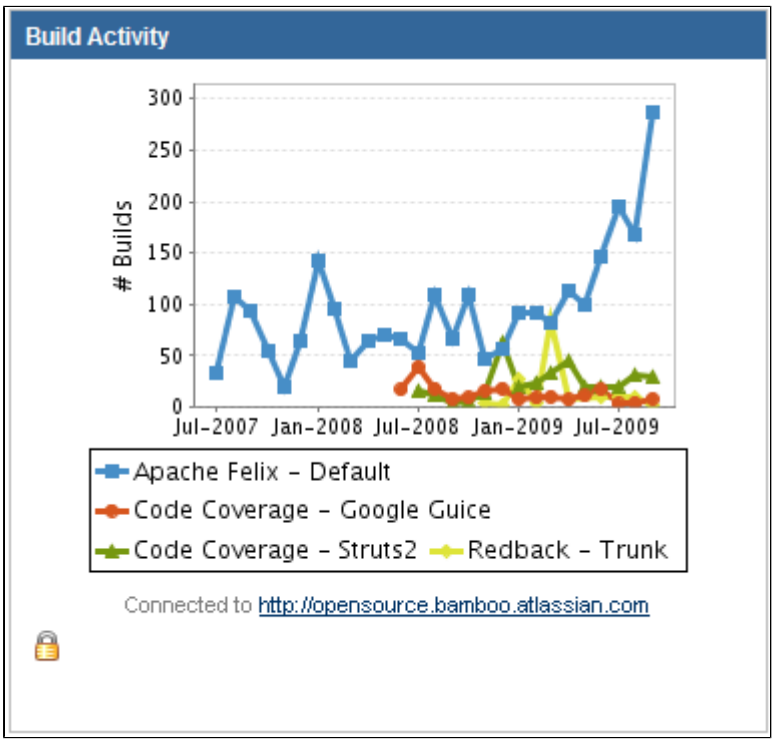
Adding the Bamboo Charts Gadget

The **Bamboo Charts** gadget displays various charts and plan statistics from a particular Bamboo server.

What does it look like?

The **Bamboo Charts** gadget should appear as follows on the dashboard:

Screenshot: 'Bamboo Charts' gadget



The Bamboo Charts gadget will only be available to add to your dashboard if your JIRA administrator has configured the Bamboo plugin on your JIRA server. Also, if you have added multiple Bamboo servers in JIRA there will be one Bamboo Charts gadget available per server, e.g. 'Bamboo Charts Gadget from http://172.20.5.83:8085', 'Bamboo Charts Gadget from http://172.19.6.93:8085', etc.

Adding the 'Bamboo Charts' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Bamboo Charts**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory. The 'Bamboo Charts' gadget will appear on your dashboard as follows, ready for you to configure.
3. Click the arrow in the top right corner of the gadget to open the configuration menu and click '**Edit**'. Configure the Bamboo information to be displayed on your gadget as follows:
    - **'Select Report Type'** — Select the Bamboo report that you would like to display as a chart.
    - **'Select Plans'** — Select the plans that you would like included in the chart.
    - **'Group By'** — Select the time interval to group by in your chart.
    - **'Show Builds From'** — Select how many days worth of builds you would like to include.
    - **'Refresh Interval'** — Select how often you would like the information on the gadget to update.
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Authorising JIRA to Display Bamboo Gadgets
When you add this gadget to your JIRA dashboard, you may see a message similar to this:

> *The website (container) you have placed this gadget on is unauthorised. Please contact your system administator to have it approved.*

To fix this problem, you will need to configure your Bamboo site to allow JIRA to draw information from it via gadgets on the JIRA dashboard. To do this, your JIRA administrator first needs to define your JIRA site as an OAuth consumer in Bamboo. You will then be required to perform a once-off authentication before your gadget will display correctly.

Adding the Bamboo Plan Summary Chart Gadget

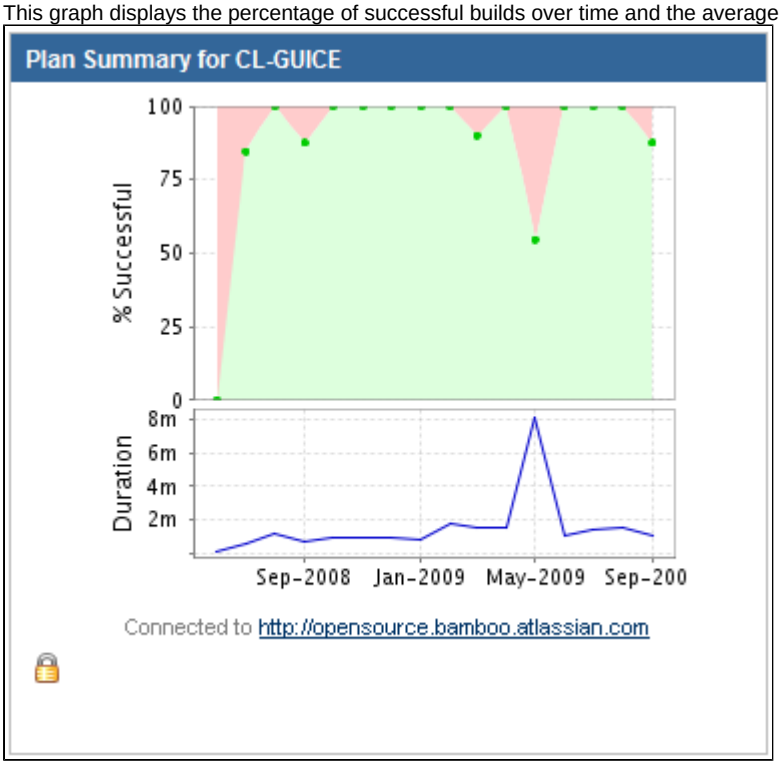
The **Bamboo Plan Summary Chart** gadget displays a graphical summary of a Bamboo build plan from a particular Bamboo server.

What does it look like?

There are two graph types available with the **Bamboo Plan Summary Chart** gadget:

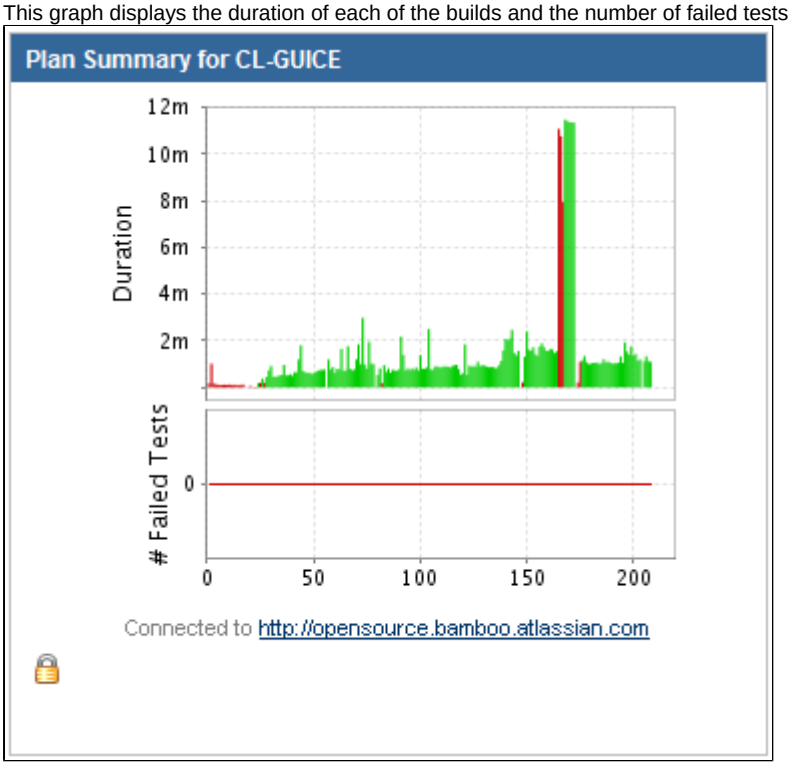
1. Group By Time Period

This graph displays the percentage of successful builds over time and the average duration of the builds in each time period:



2. Group By Build

This graph displays the duration of each of the builds and the number of failed tests per build.:



The Bamboo Plan Summary gadget will only be available to add to your dashboard if your JIRA administrator has configured the Bamboo plugin on your JIRA server. Also, if you have added multiple Bamboo servers in JIRA there will be one Bamboo Status gadget available per server, e.g. 'Bamboo Status Gadget from http://172.20.5.83:8085', 'Bamboo Status Gadget from http://172.19.6.93:8085', etc.

Adding the 'Bamboo Plan Summary Chart' gadget to your Dashboard

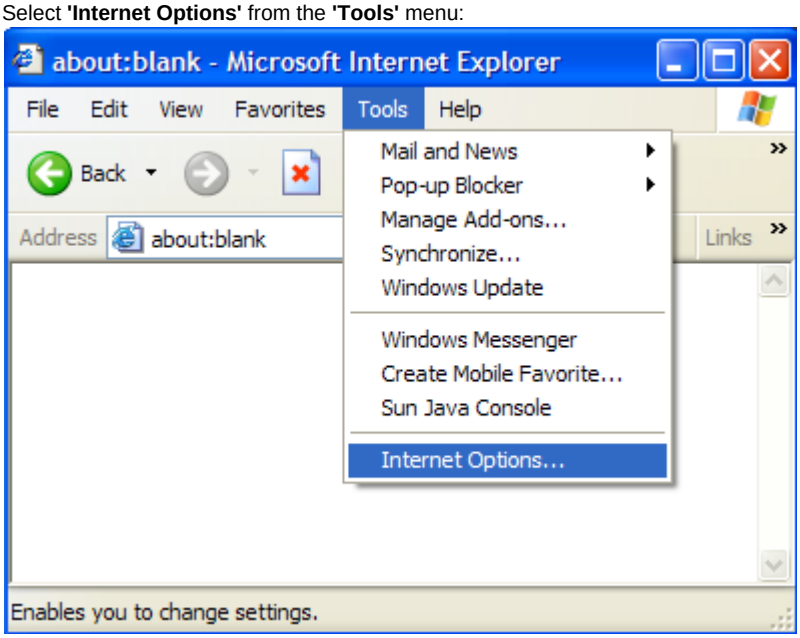
1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Bamboo Plan Summary Chart**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory. The 'Bamboo Plan Summary Chart' gadget will appear on your dashboard as follows, ready for you to configure.
3. Click the arrow in the top right corner of the gadget to open the configuration menu and click '**Edit**'. Configure the Bamboo information to be displayed on your gadget as follows:
    - **'Select Plan'** — Select the Bamboo plan for which you would like to show a summary.
    - **'Select Chart Type'** — Select the chart which you would like displayed for the plan, i.e. 'Success Rate & Duration' by desired interval *(group by time period)* or 'Duration and Failed Tests' by build number *(group by build)*.
    - **'Show Builds From'** — Select how many days worth of builds you would like to include.
    - **'Refresh Interval'** — Select how often you would like the information on the gadget to update.
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

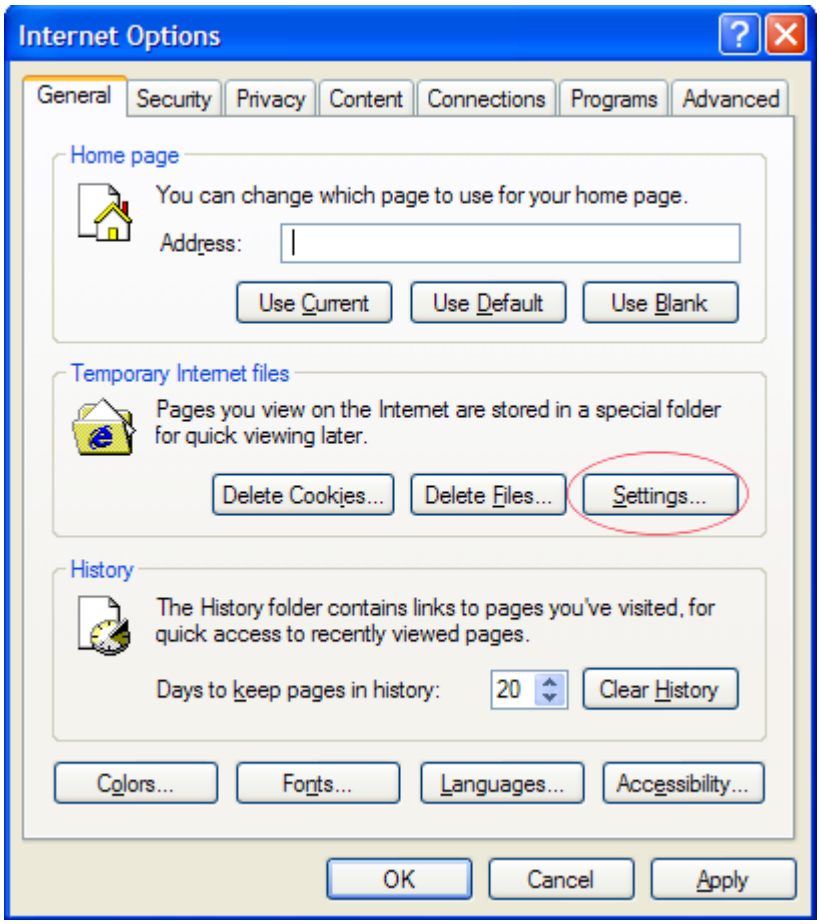
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

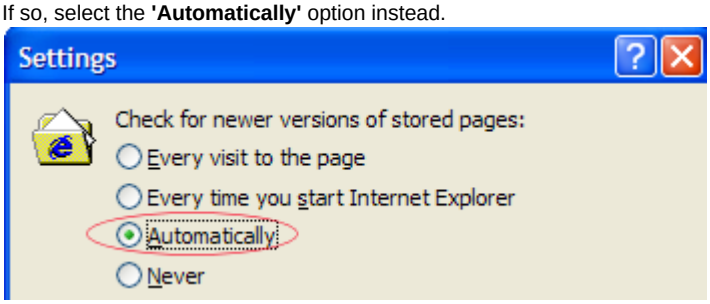
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Authorising JIRA to Display Bamboo Gadgets
When you add this gadget to your JIRA dashboard, you may see a message similar to this:

> *The website (container) you have placed this gadget on is unauthorised. Please contact your system administator to have it approved.*

To fix this problem, you will need to configure your Bamboo site to allow JIRA to draw information from it via gadgets on the JIRA dashboard. To do this, your JIRA administrator first needs to define your JIRA site as an OAuth consumer in Bamboo. You will then be required to perform a once-off authentication before your gadget will display correctly.

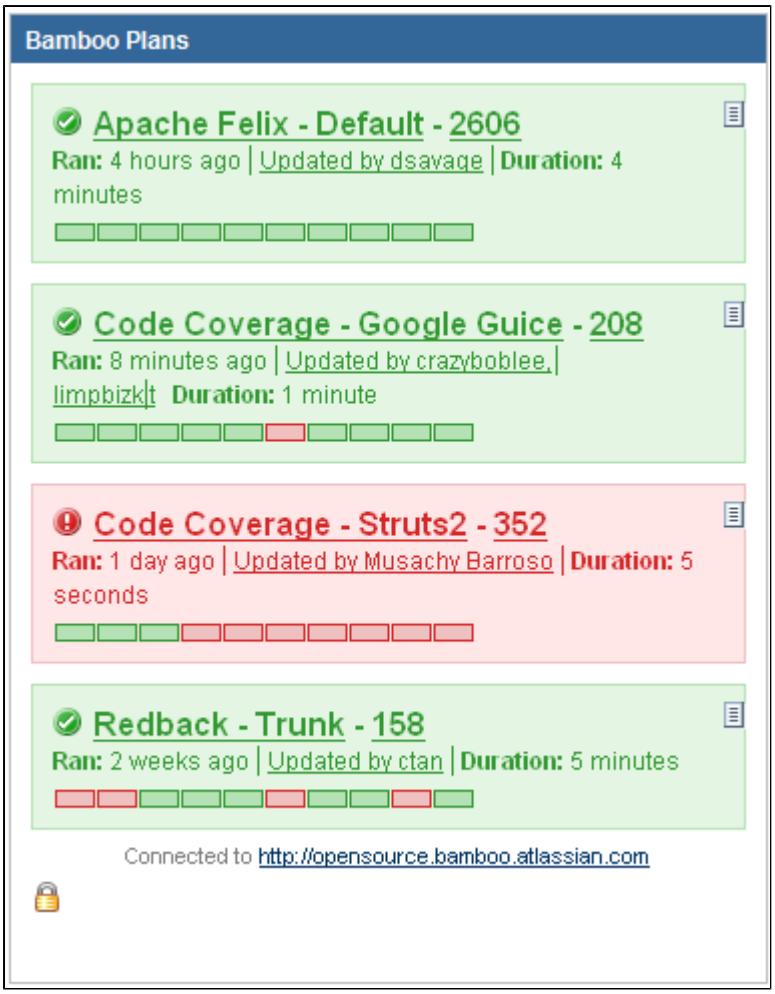
Adding the Bamboo Plans Gadget

The **Bamboo Plans** gadget displays a list of all plans on a particular Bamboo server and each plan's current status.

What does it look like?

The **Bamboo Plans** gadget should appear as follows on the dashboard:

Screenshot: 'Bamboo Plans' gadget



The Bamboo Plans gadget will only be available to add to your dashboard if your JIRA administrator has configured the Bamboo plugin on your JIRA server. Also, if you have added multiple Bamboo servers in JIRA there will be one Bamboo Status gadget available per server, e.g. 'Bamboo Plans Gadget from http://172.20.5.83:8085', 'Bamboo Plans Gadget from http://172.19.6.93:8085', etc.

Adding the 'Bamboo Plans ' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Bamboo Plans**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory. The 'Bamboo Plans' gadget will appear on your dashboard as follows, ready for you to configure.
3. Click the arrow in the top right corner of the gadget to open the configuration menu and click '**Edit**'. Configure the Bamboo information to be displayed on your gadget as follows:
    - **'Select Plans'** — Select the Bamboo plan which you would displayed on your gadget.
    - **'Refresh Interval'** — Select how often you would like the information on the gadget to update.
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Authorising JIRA to Display Bamboo Gadgets
When you add this gadget to your JIRA dashboard, you may see a message similar to this:

> *The website (container) you have placed this gadget on is unauthorised. Please contact your system administator to have it approved.*

To fix this problem, you will need to configure your Bamboo site to allow JIRA to draw information from it via gadgets on the JIRA dashboard. To do this, your JIRA administrator first needs to define your JIRA site as an OAuth consumer in Bamboo. You will then be required to perform a once-off authentication before your gadget will display correctly.

Adding the Bugzilla ID Search Gadget

The **Bugzilla Issue ID Search** gadget allows you to search all JIRA issues for references to Bugzilla issue IDs. If the specified ID is not found within JIRA, the gadget redirects to the Bugzilla issue (if a Bugzilla server URL has been specified). This allows JIRA to become the one interface for all JIRA and Bugzilla issues.

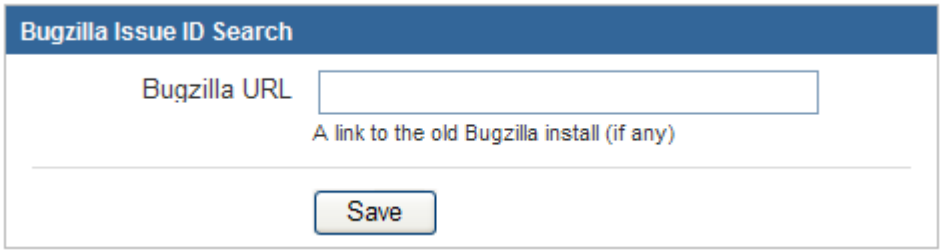
> Please note that this gadget does not work if the Bugzilla issues were imported using the JIRA Importers Plugin (which replaced the built-in JIRA importer at the release of JIRA 4.4). Instead, please use JIRA's Simple/Advanced Search to find your Bugzilla issue IDs.

What does it look like?

The **Bugzilla Issue ID Search** gadget should appear as follows on the dashboard:

Adding the 'Bugzilla Issue ID Search' gadget to Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Bugzilla Issue ID Search**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The 'Bugzilla Issue ID Search' gadget will appear on your dashboard as follows, ready for you to configure:



4. Optionally enter the URL of the Bugzilla server you wish to search.
5. Click the **'Save'** button.

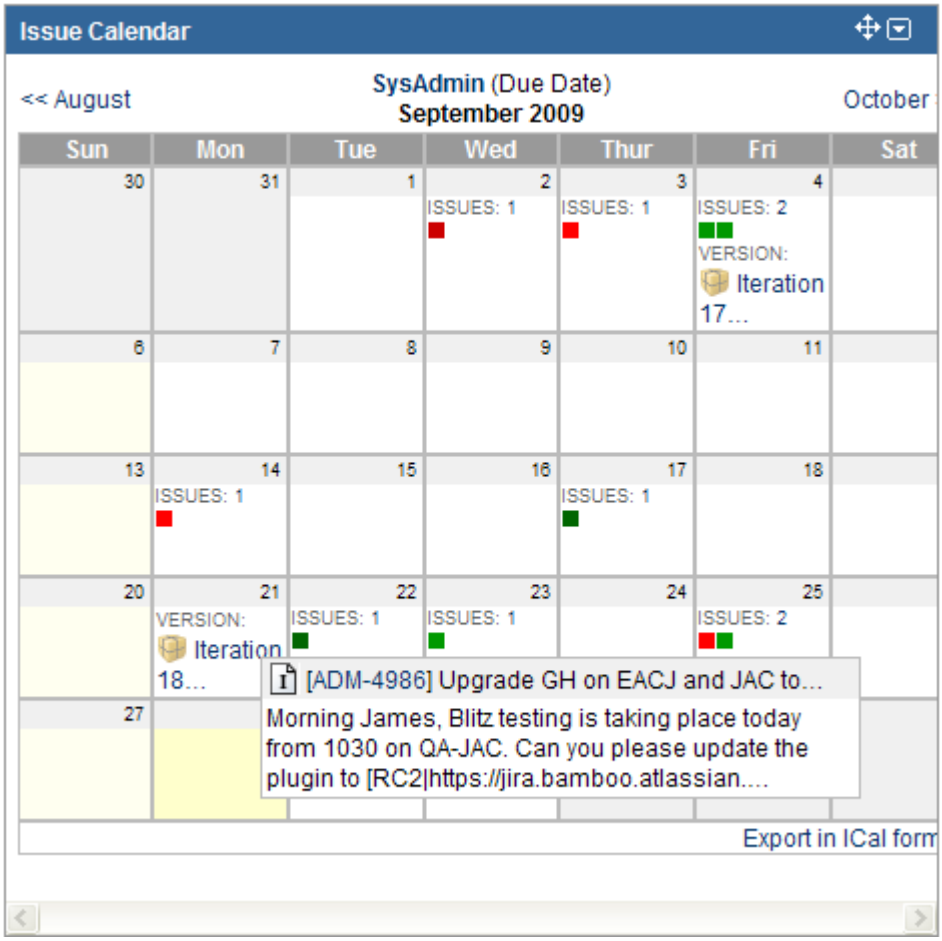
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Calendar Gadget

The **JIRA Issues Calendar** gadget shows issues and versions in a calendar format based on their due date. Calendars can be based on an issue filter or on a project.

What does it look like?

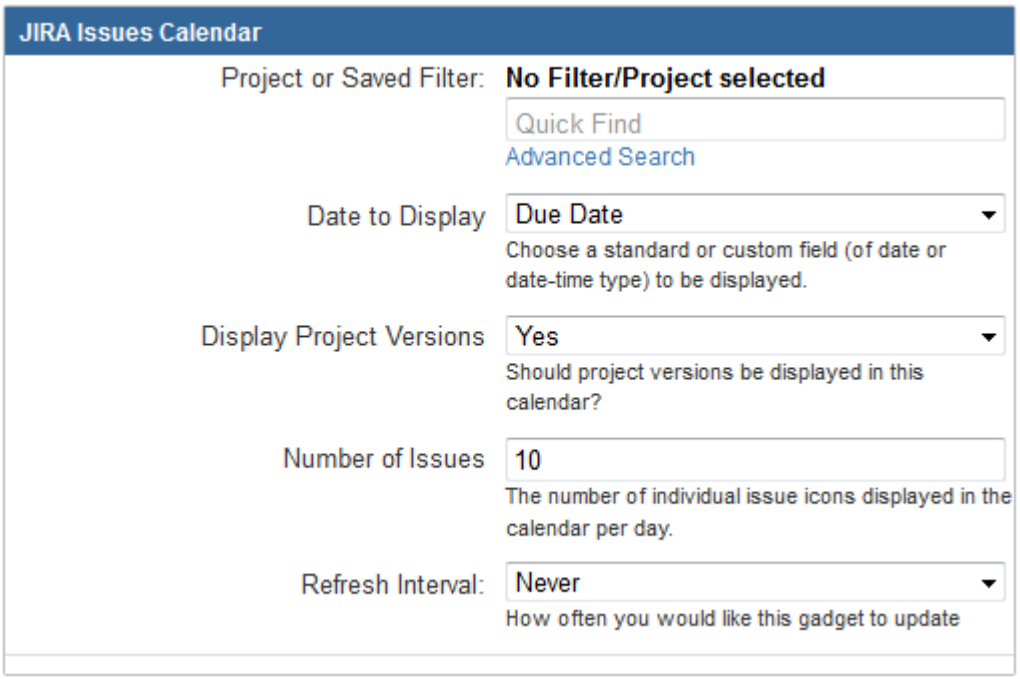
The **JIRA Issues Calendar** gadget should appear as follows on the dashboard:



The Issue Calendar gadget will only be available to add to your dashboard if your JIRA administrator has installed the Calendar plugin.

Adding the 'Calendar' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**JIRA Issues Calendar**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The JIRA Issues Calendar gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Project or Filter**' — click the 'Select' link to choose the project or filter on whose issues the calendar will be based.
   b. '**Date to Display**' — select the date field (e.g. Due Date; Created Date; Updated Date) on which the calendar will be based.
   c. '**Display Project Versions**' — select whether the calendar will display the **Release Date** of each Project Version.
   d. '**Number of Issues**' — select the maximum number of issues to be displayed on the calendar for any one day.
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

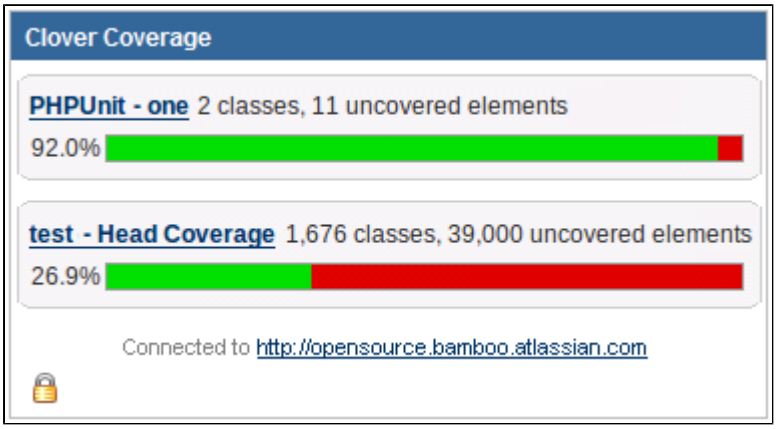
Adding the Clover Coverage Gadget

The **Clover Coverage** gadget displays the Clover coverage of plans from a particular Bamboo server.

What does it look like?

The **Clover Coverage** gadget should appear as follows on the dashboard:

Screenshot: 'Clover Coverage' gadget



The Clover Coverage gadget will only be available to add to your dashboard if your JIRA administrator has configured the Bamboo plugin on your JIRA server (the Clover gadget is exposed via the Bamboo plugin). Also, if you have added multiple Bamboo servers in JIRA there will be one Clover Coverage gadget available per server, e.g. 'Clover Coverage Gadget from http://172.20.5.83:8085', 'Clover Coverage Gadget from http://172.19.6.93:8085', etc.

Adding the 'Clover Coverage' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Clover Coverage**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory. The 'Clover Coverage' gadget will appear on your dashboard as follows,

ready for you to configure.
3. Click the arrow in the top right corner of the gadget to open the configuration menu and click '**Edit**'. Configure the information to be displayed on your gadget as follows:
    - **'Select Plans'** — Select the Bamboo plans for which you would like code coverage information displayed on your gadget.
    - **'Refresh Interval'** — Select how often you would like the information on the gadget to update.
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Authorising JIRA to Display Bamboo Gadgets
When you add this gadget to your JIRA dashboard, you may see a message similar to this:

> *The website (container) you have placed this gadget on is unauthorised. Please contact your system administator to have it approved.*

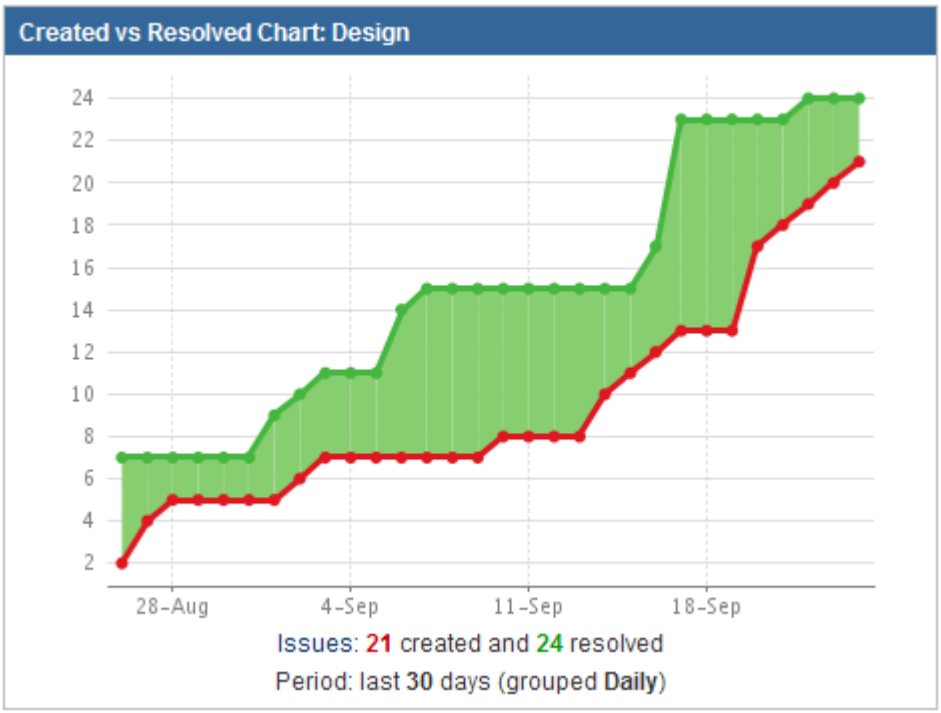
To fix this problem, you will need to configure your Bamboo site to allow JIRA to draw information from it via gadgets on the JIRA dashboard. To do this, your JIRA administrator first needs to define your JIRA site as an OAuth consumer in Bamboo. You will then be required to perform a once-off authentication before your gadget will display correctly.

Adding the Created vs Resolved Gadget

The **'Created vs Resolved'** gadget displays a difference chart showing the number of issues created vs number of issues resolved over a given period of time. The chart is based on your choice of project or issue filter, and the chart can either be cumulative or not. An issue is marked as resolved in a period if it has a resolution date in that period. The resolution date is the last date that the system Resolution field was set to any non-empty value.

What does it look like?

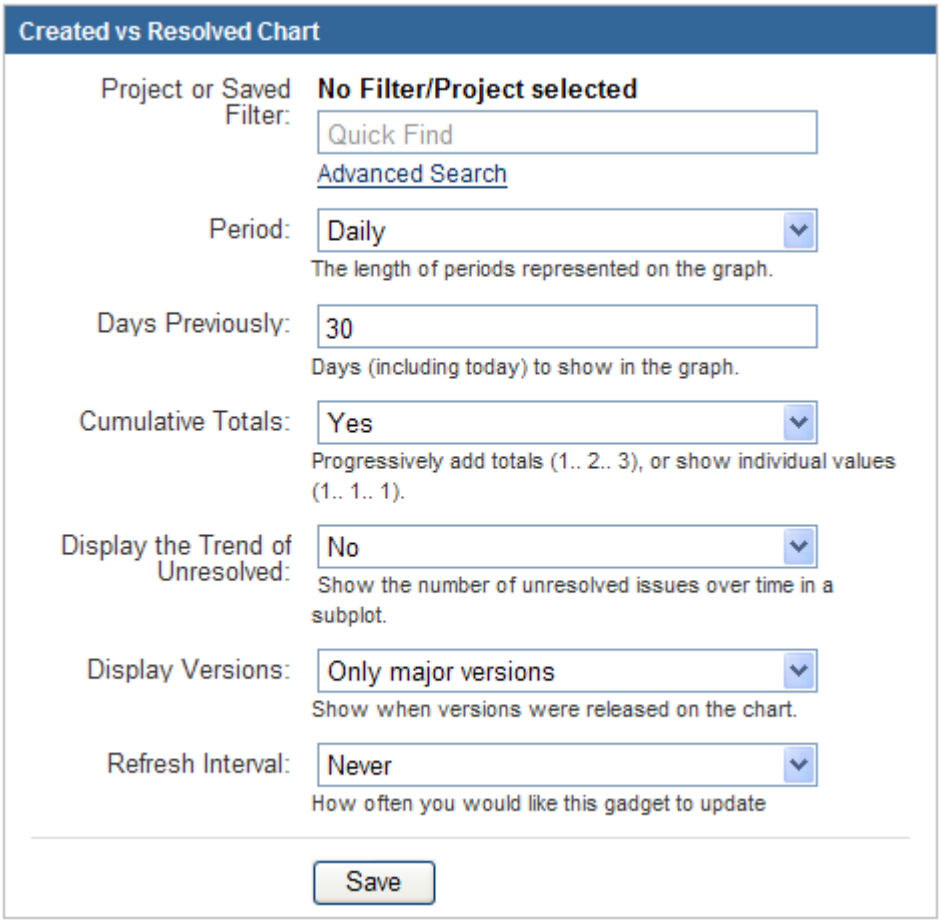
The **'Created vs Resolved'** gadget will appear as follows on the dashboard:



A report showing this information is also available.

Adding the 'Created vs Resolved Issues' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Created vs Resolved**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The 'Created vs Resolved' gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Project or Saved Filter**' — start typing the name of the project (or saved filter) on whose issues the chart will be based. Alternatively, if you're unsure of the name of the project or filter you're looking for, click 'Advanced Search' to search for a project (or saved filter) whose name contains particular text; or a saved filter that was created by a particular user and/or is shared with particular users.
b. **'Period'** — select the timeframe on which the chart will be based:
   - **'Hourly'**
   - **'Daily'**
   - **'Weekly'**
   - **'Quarterly'**
   - **'Yearly'**
c. **'Days Previously'** — enter the number of days' worth of data (counting backwards from today) to be included in the chart.
d. **'Cumulative Totals?'** — choose either:
   - **'Yes'** to progressively add data to the preceding column; or
   - **'No'** to show just a single value in each column.
e. **'Display the Trend of Unresolved?'** — choose either:
   - **'Yes'** to display an additional line graph showing the number of unresolved issues over time; or
   - **'No'** to show just the difference chart of issues created vs issues resolved.
f. **'Display Versions?'** — choose either:
   - **'All versions'** to show version release dates on the chart, for all released versions; or
   - **'Only major versions'** to show version release dates on the chart, for released versions that are named 'x.x' only; or
   - **'None'** to not show version release dates on the chart.
g. '**Refresh Interval**' — select how often you want the gadget to update the chart (never / every 15 minutes / every 30 minutes / every hour / every two hours).

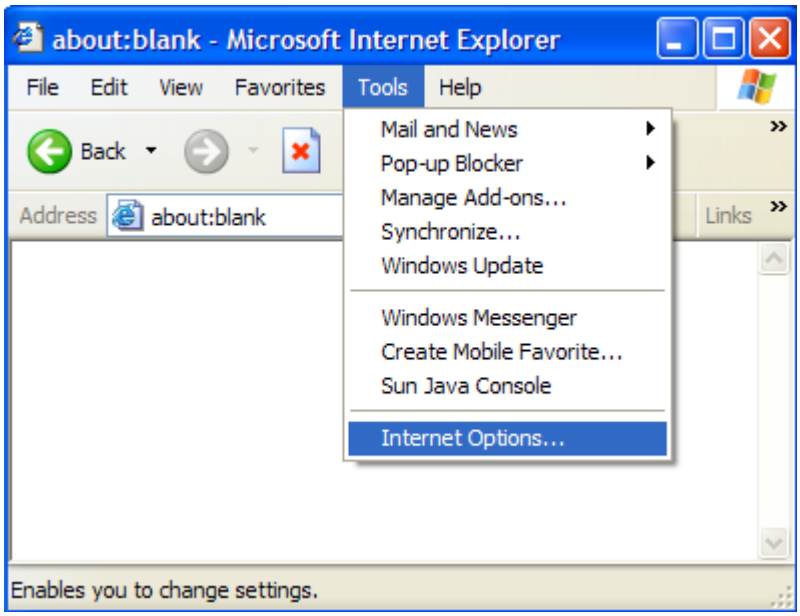
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

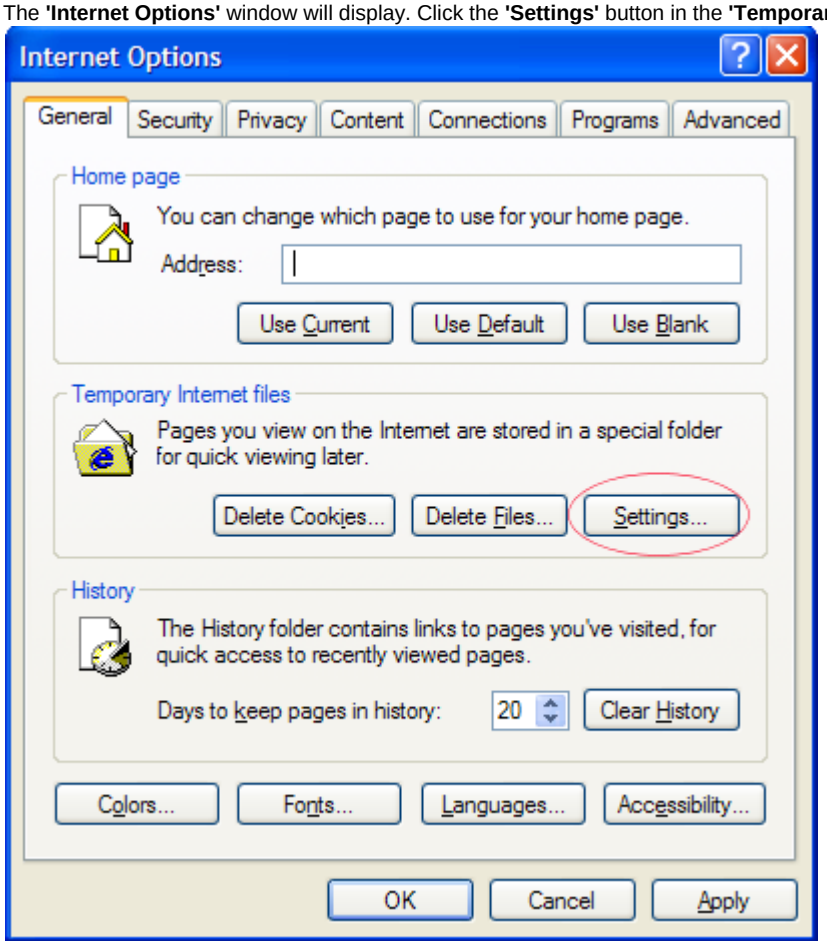
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

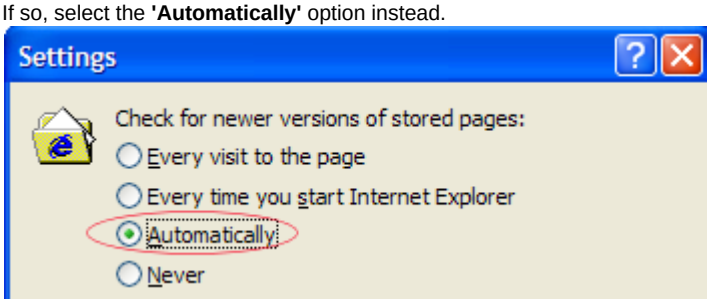
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.

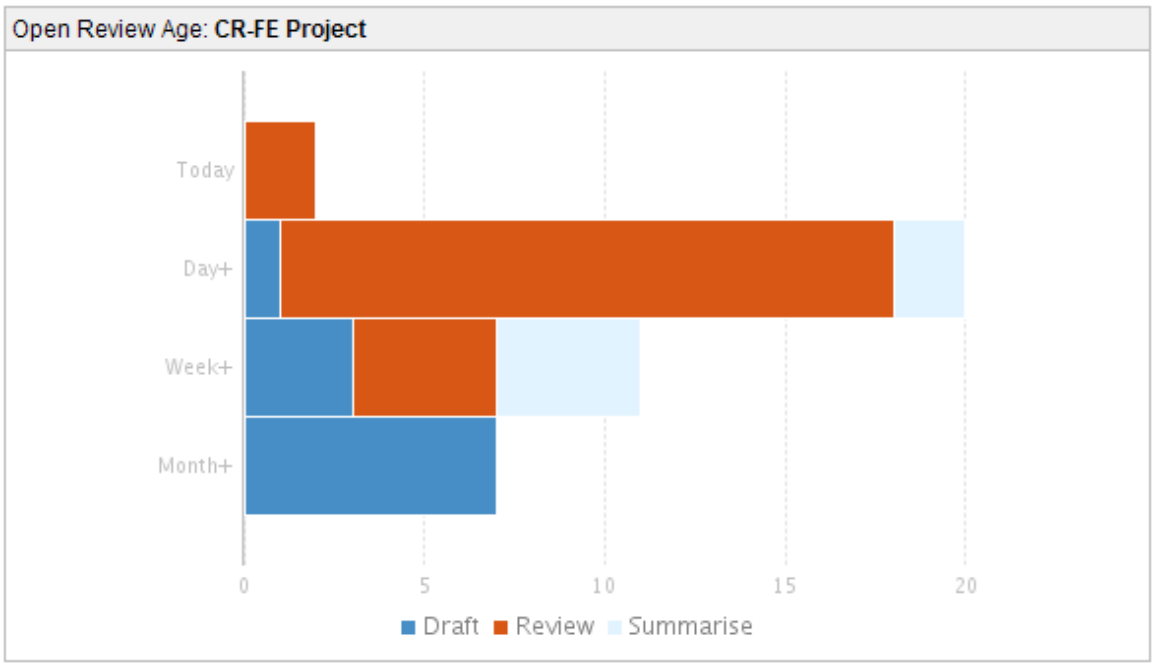


Adding the Crucible Charts Gadget

The **Crucible Charts** gadget displays various charts showing statistical summaries of your code reviews.

What does it look like?

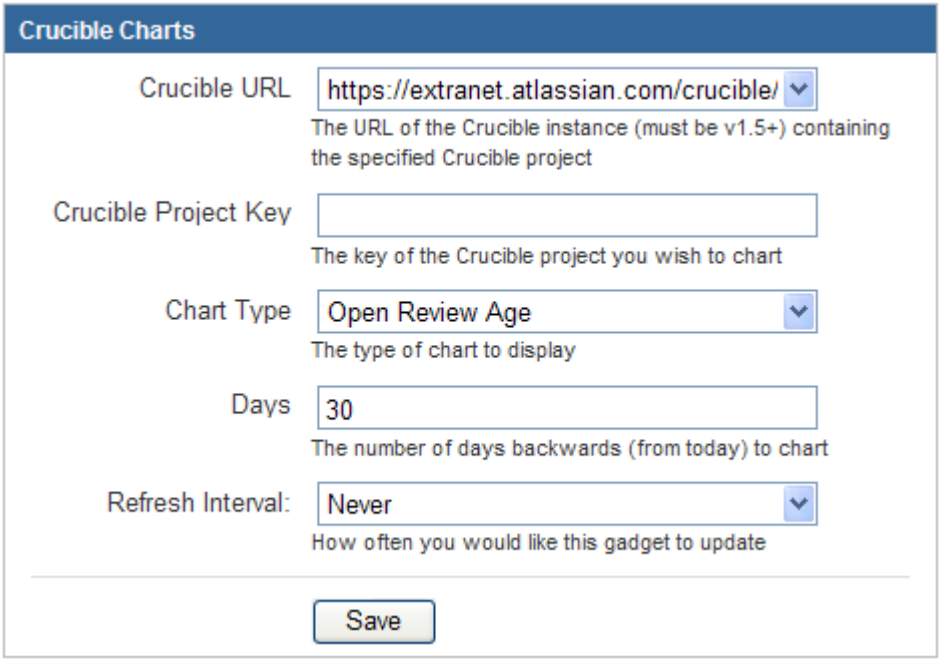
The **Crucible Charts** gadget should appear as follows on the dashboard:



The Crucible Charts gadget will only be available to add to your dashboard if your JIRA administrator has configured the FishEye plugin on your JIRA server.

Adding the 'Crucible Charts' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Crucible Charts**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The '**Crucible Charts**' gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Crucible URL**' — type the URL of your Crucible server.
   b. '**Crucible Project Key**' — type the project key of the Crucible project in which you are interested.
   c. '**Chart Type**' — select from the following:
      - '**Open Review Age**' — the age of open reviews, broken down by status.
      - '**Defect Classification**' — the number of defects raised, broken down by classification.

- '**Open Review Volume**' — the volume of open reviews over the specified time period.
- '**Comment Volume**' — the volume of comments authored over the specified time period.
- '**Defect Rank**' — the number of defects raised, broken down by rank.

d. '**Number of Days**' — type the number of days' worth of data (backwards from today) that you want the gadget to display.
e. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

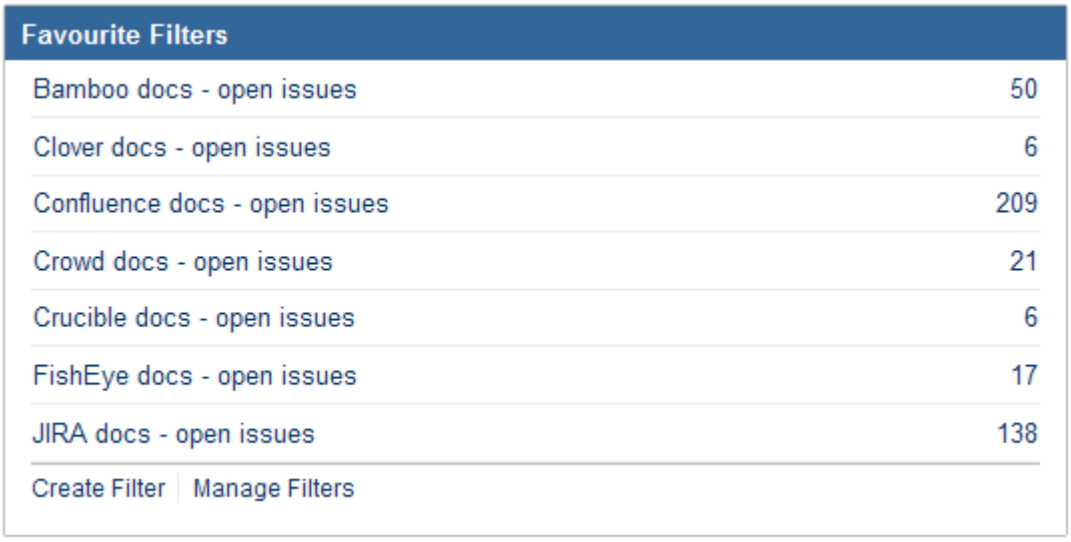
Adding the Favourite Filters Gadget

The **Favourite Filters** gadget displays a list of all the issue filters that have currently been added by you as a 'favourite' filter.

Read more about adding an issue filter as a favourite filter in the issue filters documentation.

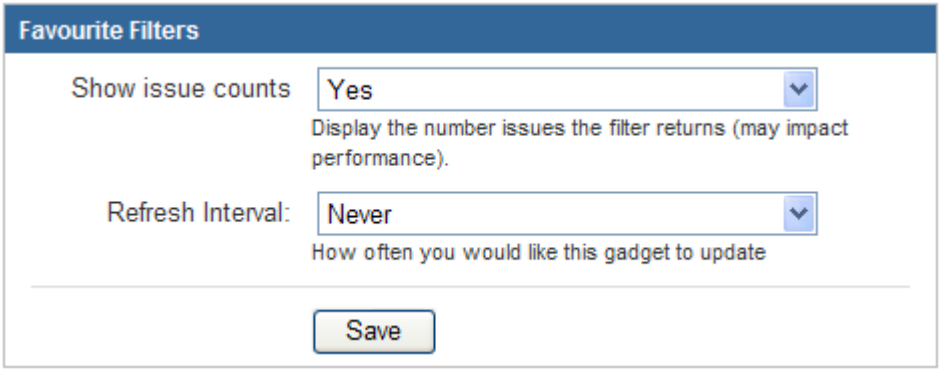
What does it look like?

The **Favourite Filters** gadget should appear as follows on the dashboard:



Adding the 'Favourite Filters' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Favourite Filters**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Favourite Filters gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Show issue counts**' — select whether, for each of your favourite filters, you wish to display the number of issues that match the filter. Note that choosing '**Yes**' may impact your dashboard's performance.
b. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

The Favourite Filters gadget is added by default to the 'System Default' dashboard.

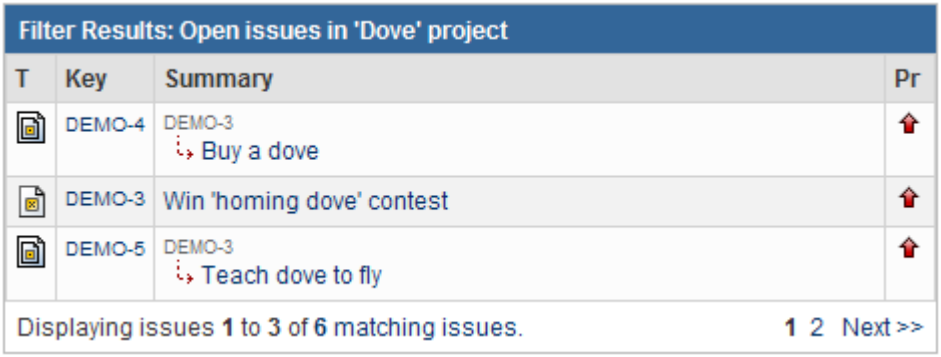
The '**Favourite Filters**' gadget has replaced the '**List All Filters**' portlet.

Adding the Filter Results Gadget

The **Filter Results** gadget displays the results of a specified issue filter on the dashboard. It can be configured to display a maximum number of issues from the collection returned from the specified filter.

What does it look like?

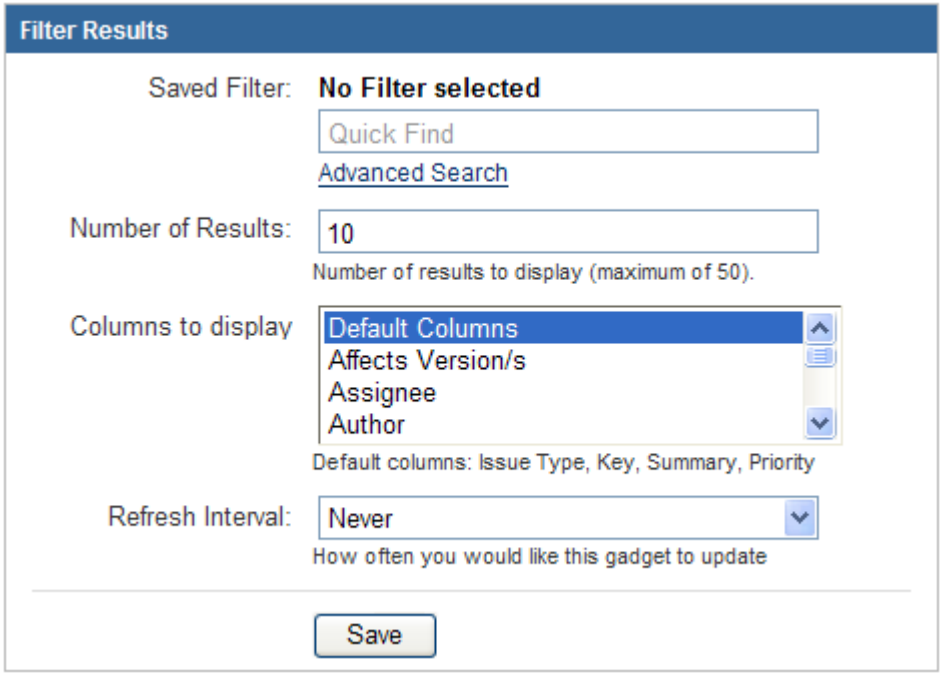
The **'Filter Results'** gadget should appear as follows on the dashboard:



You may also be interested in the Two-Dimensional Filter Statistics Gadget

Adding the 'Filter Results' Gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Filter Results**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Filter Results gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Saved Filter**' — start typing the name of the filter, or click the '**Advanced Search**' link to search for a filter/select one of your favourite filters/select a filter that you have created.
   b. '**Number of Results**' — type the maximum number of issues that you want the gadget to display per page.
   c. '**Columns to display**' — select the column(s) (i.e. issue fields) that you want the gadget to display, or choose '**Default Columns**' to display Issue Type, Key and Summary.
   d. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

The "**Filter Results**" gadget has replaced the "**Saved Filter**" portlet.

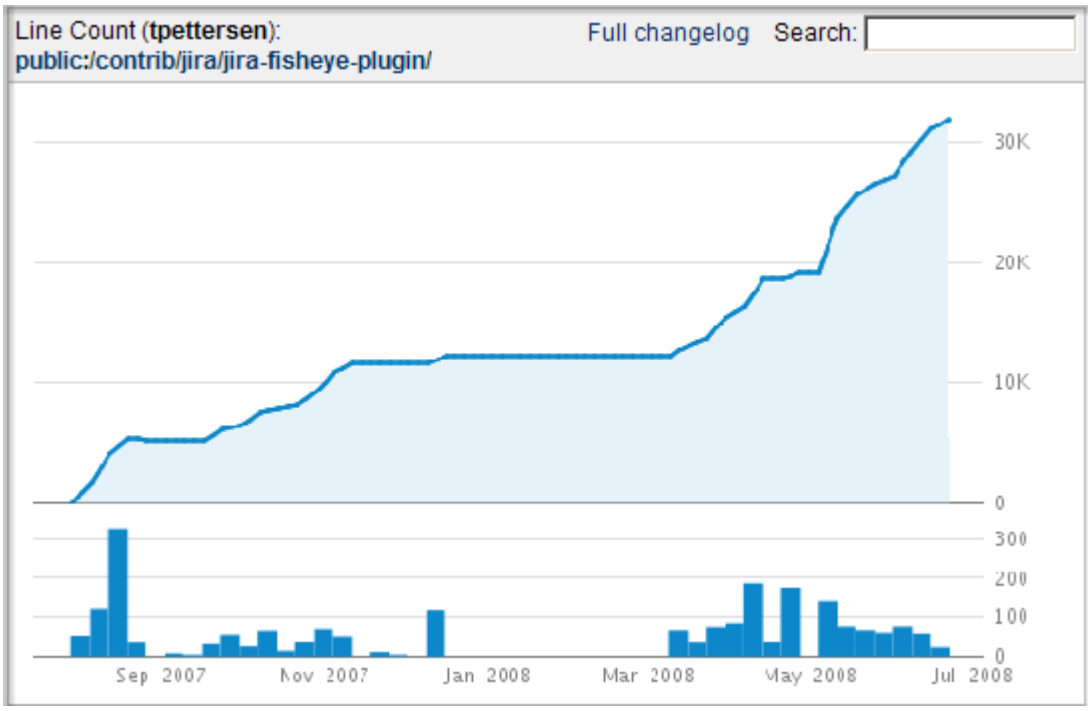
Adding the FishEye Charts Gadget

The **FishEye Charts** gadget displays two charts showing statistics about your sourcecode repository:

- Lines of code
- Commit activity

What does it look like?

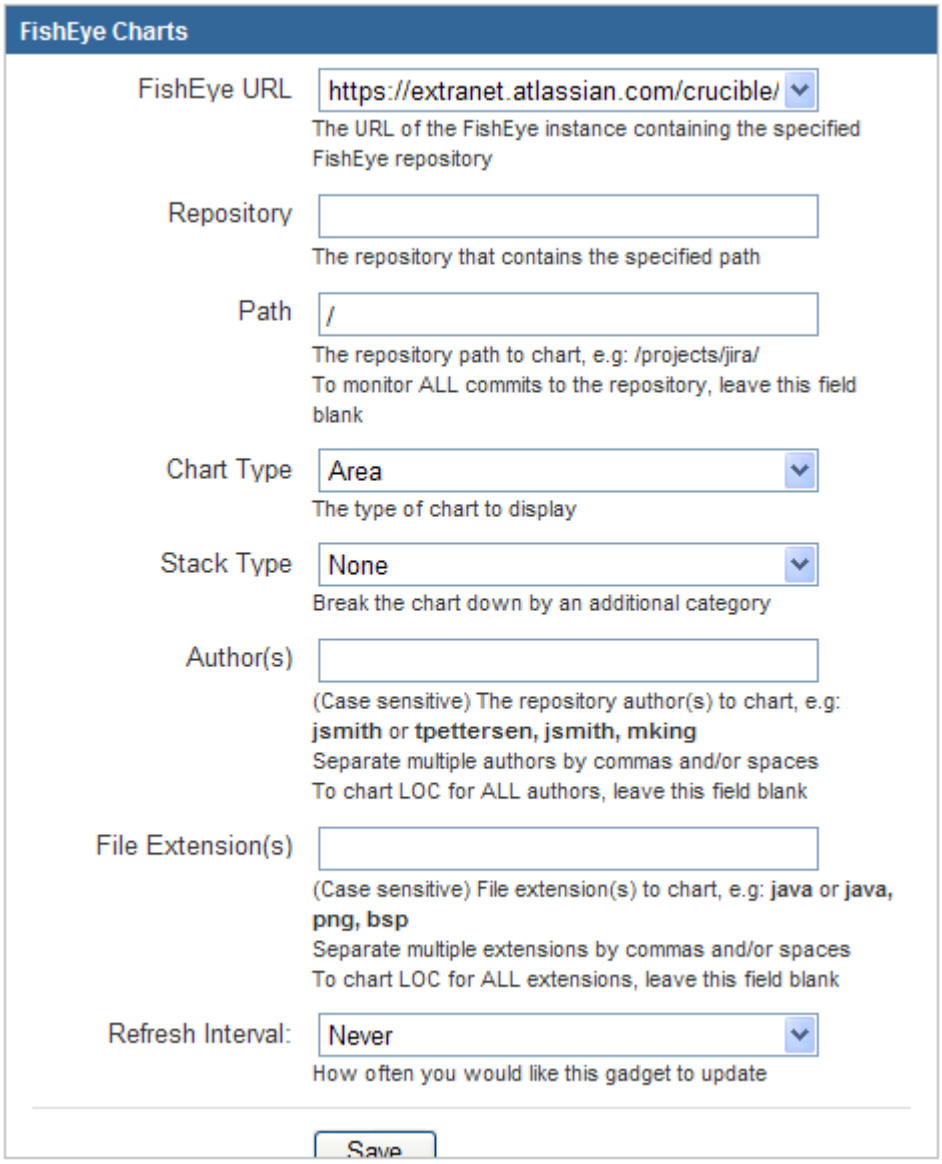
The **FishEye Charts** gadget should appear as follows on the dashboard:



The FishEye Charts gadget will only be available to add to your dashboard if your JIRA administrator has configured the FishEye plugin on your JIRA server.

Adding the 'FishEye Charts' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**FishEye Charts**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The '**FishEye Charts**' gadget will appear on your dashboard as follows, ready for you to configure:



a. '**FishEye URL**' — type the URL of your FishEye server.
b. '**Repository**' — type the name of your FishEye repository.
c. '**Path**' — optionally type the path within your repository that contains the directory in which you are interested. Leave blank to include all directories in your repository.
d. '**Chart Type**' — select from the following: '**Area**', '**Change**', '**Line**' or '**Pie**'.
e. '**Stack Type**' — allows you to break the chart down by **Subdirectory**, **File Extension** and **Author**. For example, in a pie chart with an *author* stacktype, each slice would represent the LOC (lines of code) of a different author.
f. '**Author(s)**' — optionally type the repository login name of the author(s) in whose code you are interested. Leave blank to include all authors.
g. '**File Extension(s)**' — optionally type the file extensions(s) in which you are interested. Leave blank to include all file types.
h. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

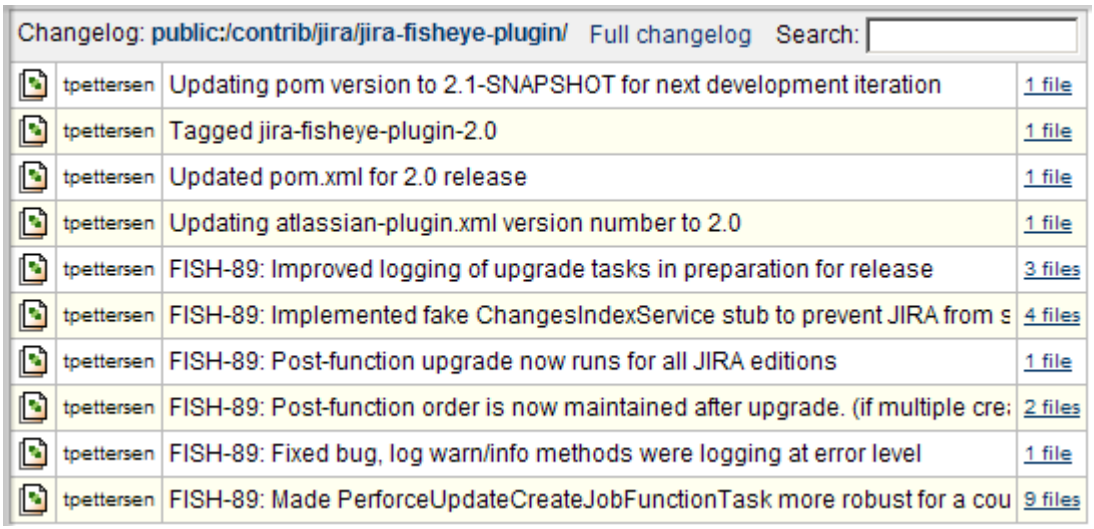
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the FishEye Recent Changesets Gadget

The **FishEye Recent Changesets** gadget displays a number of recent changesets from a FishEye repository.

What does it look like?

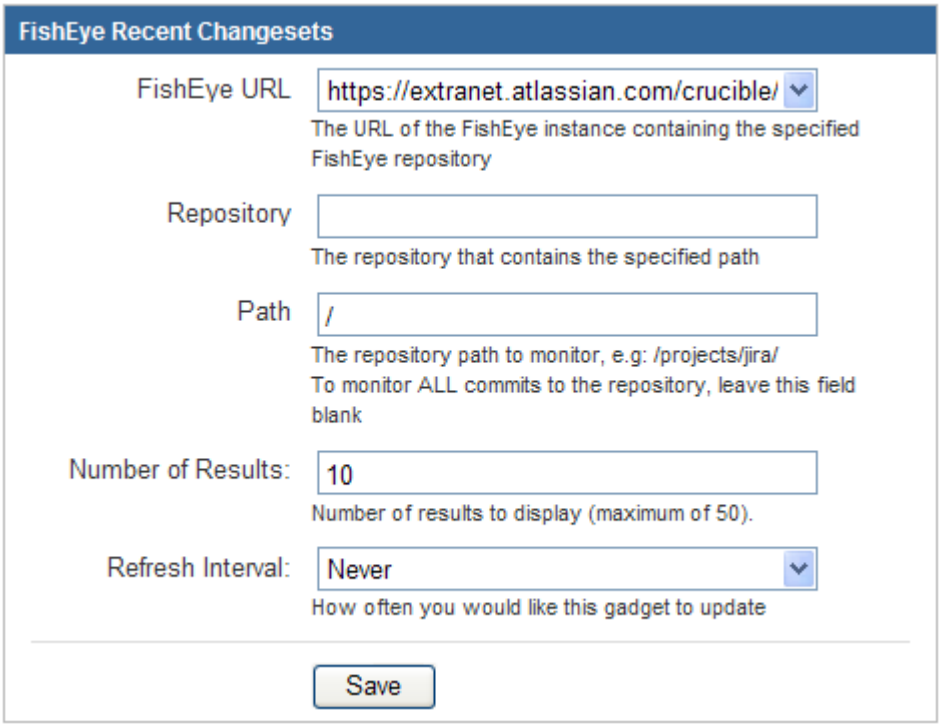
The **FishEye Recent Changesets** gadget should appear as follows on the dashboard:



The FishEye Recent Changesets gadget will only be available to add to your dashboard if your JIRA administrator has configured the FishEye plugin on your JIRA server.

Adding the 'FishEye Recent Changesets' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**FishEye Recent Changesets**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The '**FishEye Recent Changesets**' gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**FishEye URL**' – type the URL of your FishEye server.
   b. '**Repository**' – type the name of your FishEye repository.
   c. '**Path**' – optionally type the path within your repository that contains the directory in which you are interested. Leave blank to include all directories in your repository.
   d. '**Number of Results**' – type the number of commits that you want the gadget to display.
   e. '**Refresh Interval**' – select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the In Progress Gadget

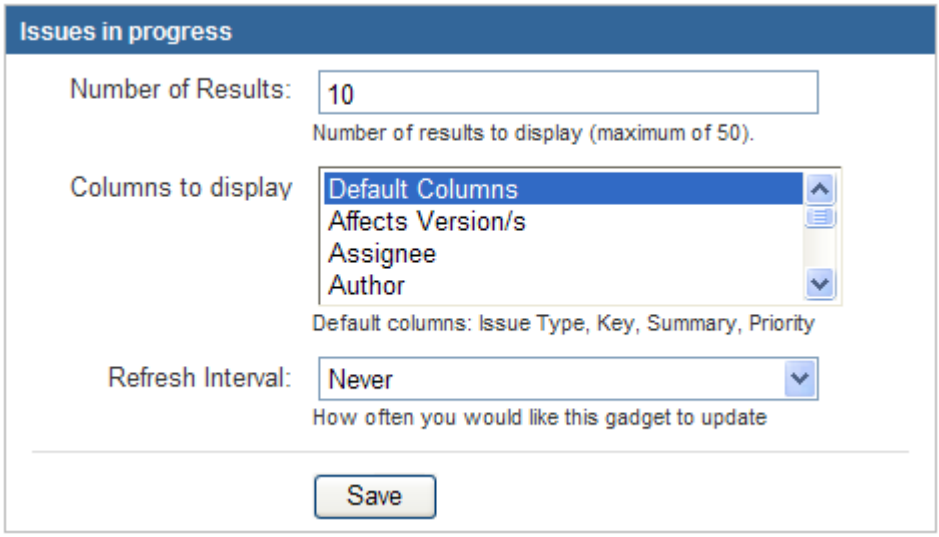
The **Issues in Progress** gadget displays all issues that are currently in progress and assigned to you.

What does it look like?

The **Issues in Progress** gadget should appear as follows on the dashboard:

Adding the 'Issues in Progress' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Issues in Progress**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Issues in Progress gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Number of Results**' — type the maximum number of issues that you want the gadget to display per page.
   b. '**Columns to display**' — select the column(s) that you want the gadget to display, or choose '**Default Columns**' to display Issue Type, Key and Summary.
   c. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

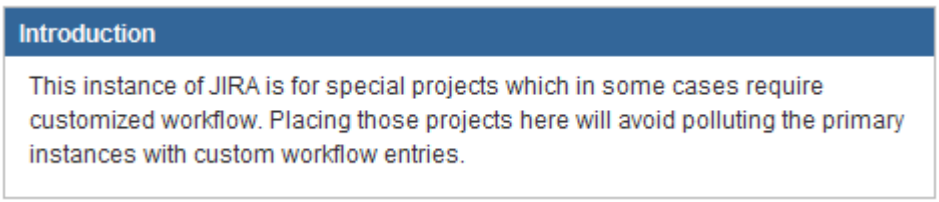
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Introduction Gadget

The **Introduction** gadget displays a configurable introduction message on the dashboard.

What does it look like?

The **Introduction** gadget should appear as follows on the dashboard:



Adding the 'Introduction' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Introduction**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.

3. The Introduction gadget will appear on your dashboard.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

> The text/html displayed in the **Introduction** gadget is configured by your JIRA administration, through the JIRA configuration page.

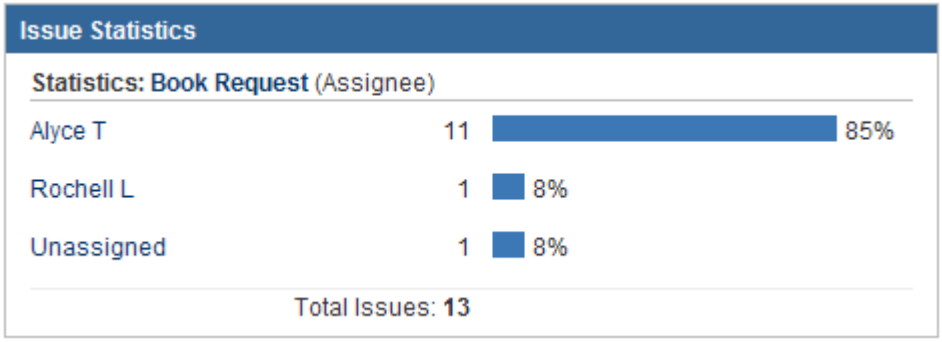
Adding the Issue Statistics Gadget

The **Issue Statistics** gadget displays the collection of issues returned from a specified project or saved filter, grouped by a specified field.

For instance, a filter can be created to return all open issues from all projects. The gadget can then be configured to display these issues broken down by a field (e.g. Assignee).

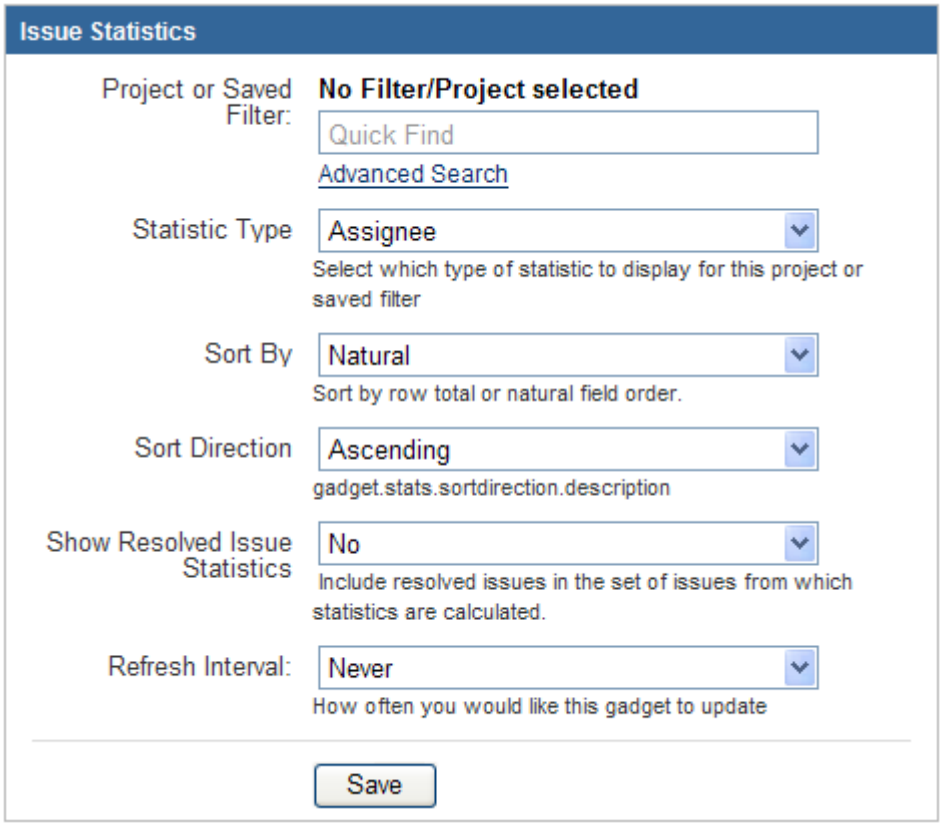
What does it look like?

The **Issue Statistics** gadget should appear as follows on the dashboard:



Adding the 'Issue Statistics' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Issue Statistics**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Issue Statistics gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Project or Saved Filter**' — start typing the name of the project or filter, or click the '**Advanced Search**' link to search for a

project or filter.
   b. '**Statistic Type**' — select the field (e.g. Assignee; Component; Priority; Resolution; etc) on which the issues will be grouped.
   c. '**Sort By**' — select how to sort the values of your selected field:
      - '**Natural**' — this will use the field's native sorting order, e.g. for the "Assignee" field, the assignee names would be sorted alphabetically.
      - '**Total**' — this will sort by the number of issues that match each value, e.g. for the "Assignee" field, the assignee names would be sorted by the number of issues assigned to each person.
   d. '**Sort Direction**' — select whether the field values should be sorted in **Ascending** or **Descending** order.
   e. '**Show Resolved Issue Statistics**' — select whether the graph will include resolved issues (i.e. issues that have a Resolution).
   f. '**Refresh Interval**' — select how often you want the gadget to update (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

> The "**Issue Statistics**" gadget has replaced the "**Filter Statistics**" portlet.

Adding the JIRA News Gadget

The **JIRA: News** gadget displays recent Atlassian news about JIRA.

What does it look like?

The **JIRA: News** gadget should appear as follows on the dashboard:

Adding the JIRA: News gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**JIRA: News**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.

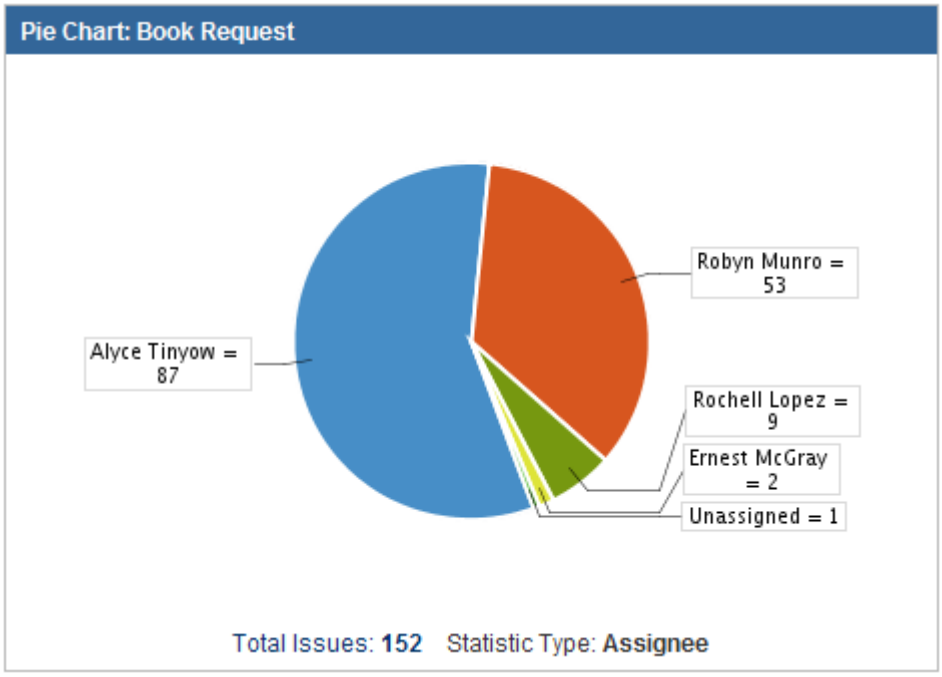
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Pie Chart Gadget

The **'Pie Chart'** gadget displays issues returned from a specified project or issue filter, grouped by a specified field. For example, an issue filter can be created to retrieve all open issues for a particular version of a particular project. The 'Pie Chart' gadget can then be used to display these issues grouped by a specified field (e.g. Assignee).

What does it look like?

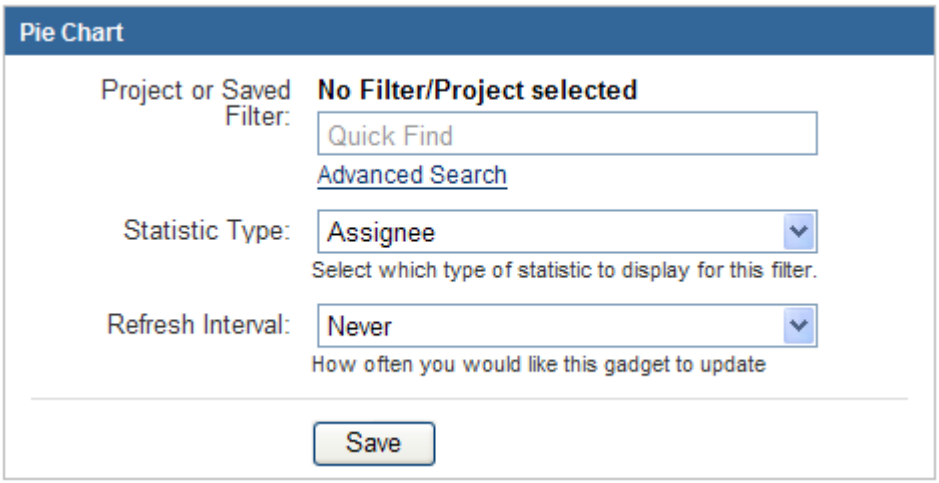
The **'Pie Chart'** gadget will appear as follows on the dashboard:



Click any section of the chart to view the matching issues.

Adding the 'Pie Chart' Gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Pie Chart**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Pie Chart gadget will appear on your dashboard as follows, ready for you to configure:



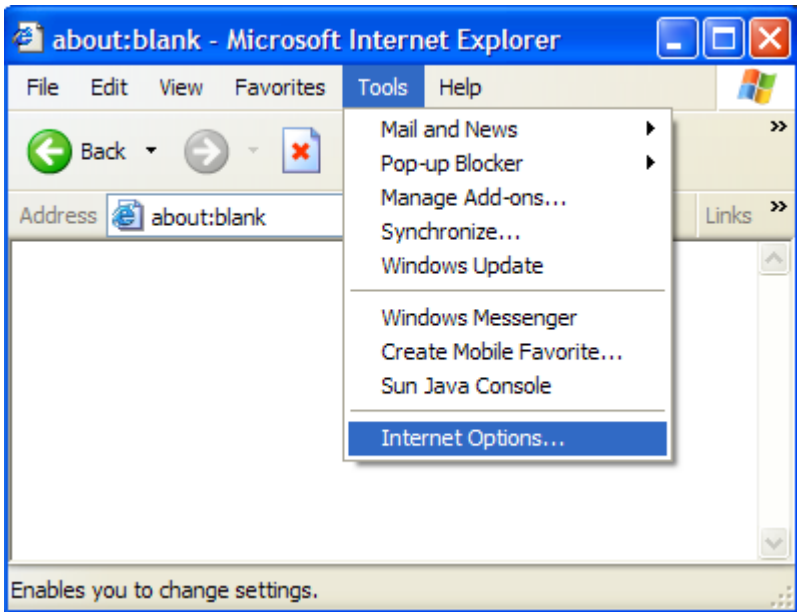
   a. '**Project or Saved Filter**' — start typing the name of the project or filter, or click the 'Advanced Search' link to search for a project or filter.
   b. '**Statistic Type**' — select the field on which the pie chart will be based.
   c. '**Refresh Interval**' — select how often you want the gadget to update the chart (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

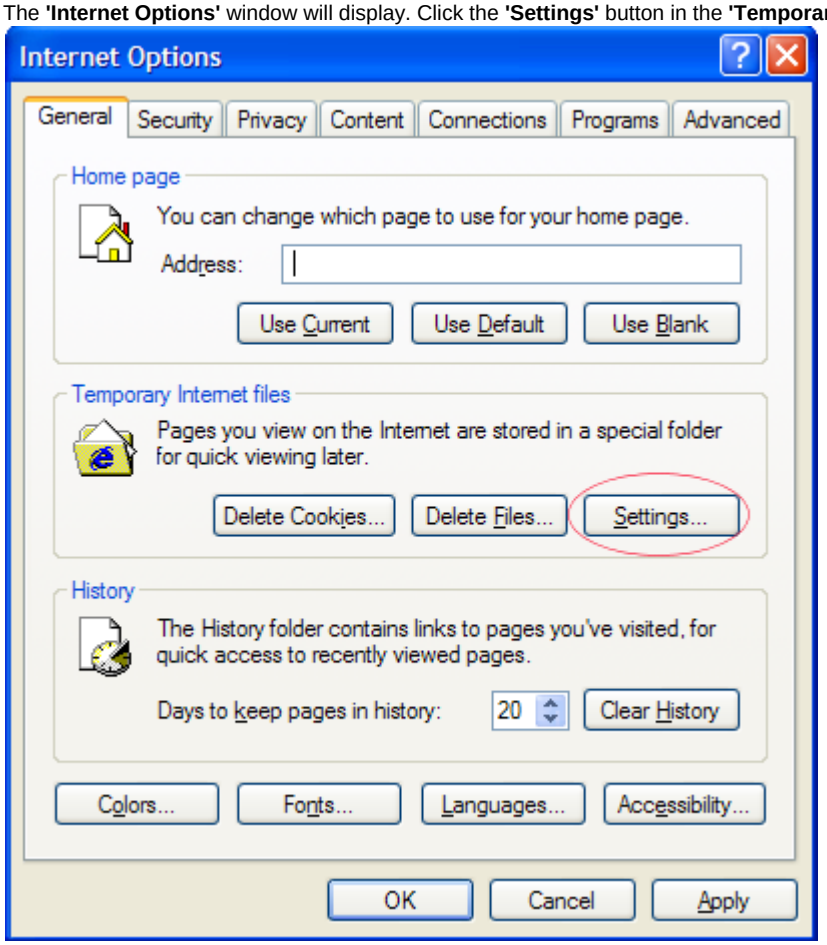
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

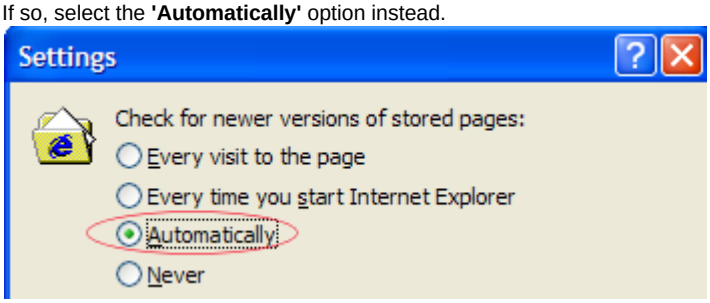
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



Adding the Projects Gadget

The **Projects** gadget provides information and various filters related to specified project(s) within JIRA.

What does it look like?

The Projects gadget should appear as follows on the dashboard:

The 'menu' icon provides links to the following, for each project:

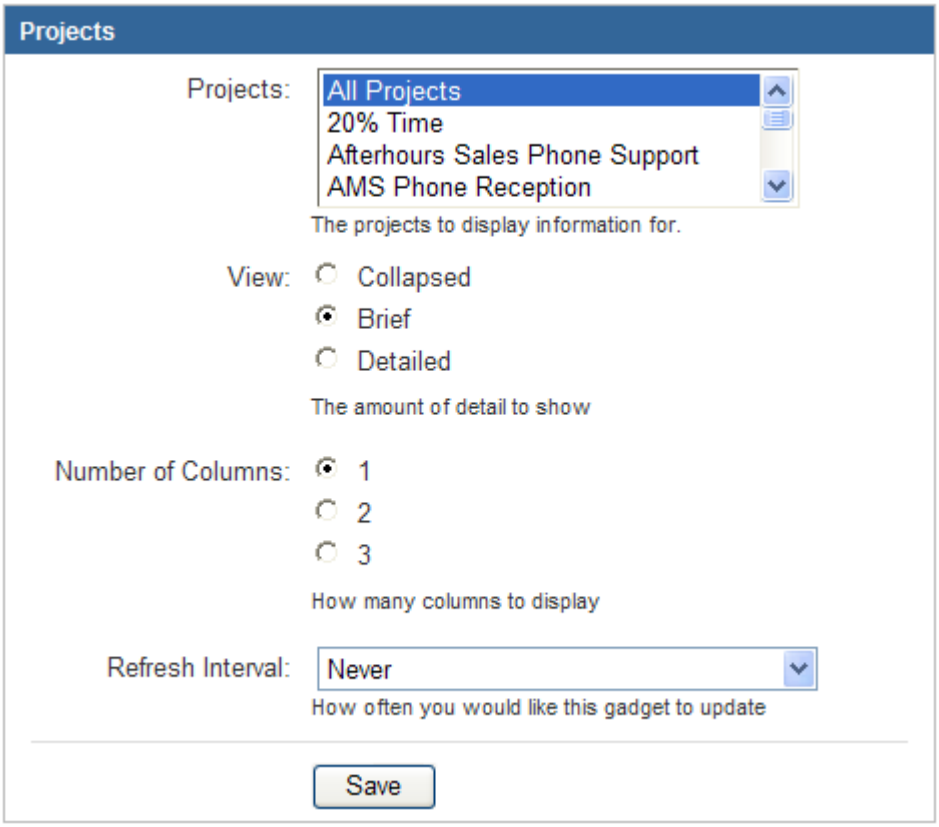
- Summary — Shows recent activity in the project, plus a list of issues that are due soon.
- Issues — Shows summaries of: all issues in a project, grouped by Status; and unresolved issues, grouped by Assignee, Priority, Version and Component.
- Road Map — Shows unresolved issues for upcoming versions of a project.
- Change Log — Shows resolved issues for previous versions of a project.
- Popular Issues — Shows a project's unresolved issues, ordered by popularity (votes).
- Versions — Shows recent versions for a given project.
- Components — Shows all components in a given project.
- Builds — Shows recent Bamboo builds for a given project.
- Source — Shows recent FishEye changesets for a given project.
- Reviews — Shows recent Crucible code for a given project.

The 'filter' icon provides links to the following issue filters in the Issue Navigator, for each project:

- All
- Resolved recently
- Outstanding
- Added recently
- Unscheduled
- Updated recently
- Assigned to me
- Most important
- Reported by me

Adding the 'Projects' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Projects**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Projects gadget will appear on your dashboard as follows, ready for you to configure:



- '**Projects**' and '**Categories**' — select one or more projects (or 'All Projects') to display in the gadget. (Note: '**Categories**' will only be shown if some have been defined in your JIRA system.)
- '**View**' — select either '**Collapsed**', '**Brief**' or '**Detailed**' to specify how much information fo display per project.
- '**Number of Columns**' — select how the gadget will be formatted (1 column, 2 columns or 3 columns).
- '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Quick Links Gadget

The **Quick Links** gadget displays a number of useful links to frequently-used searches and operations.

What does it look like?

The **Quick Links** gadget should appear as follows on the dashboard:

Adding the Quick Links gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Quick Links**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.

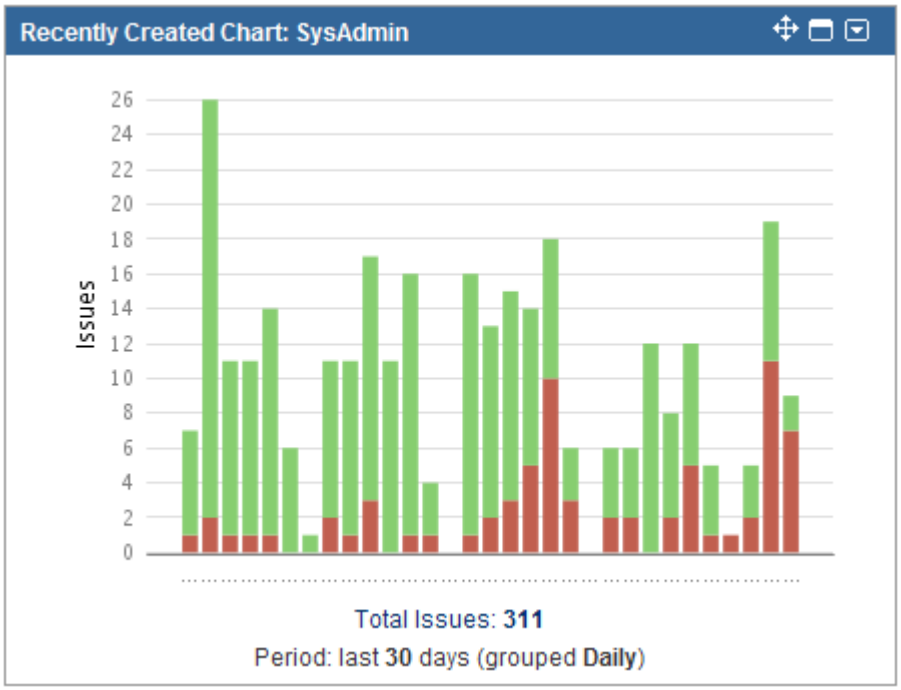
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Recently Created Issues Gadget

The **'Recently Created Issues'** gadget displays a bar chart showing the rate at which issues are being created, as well as how many of those created issues are resolved. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).

What does it look like?

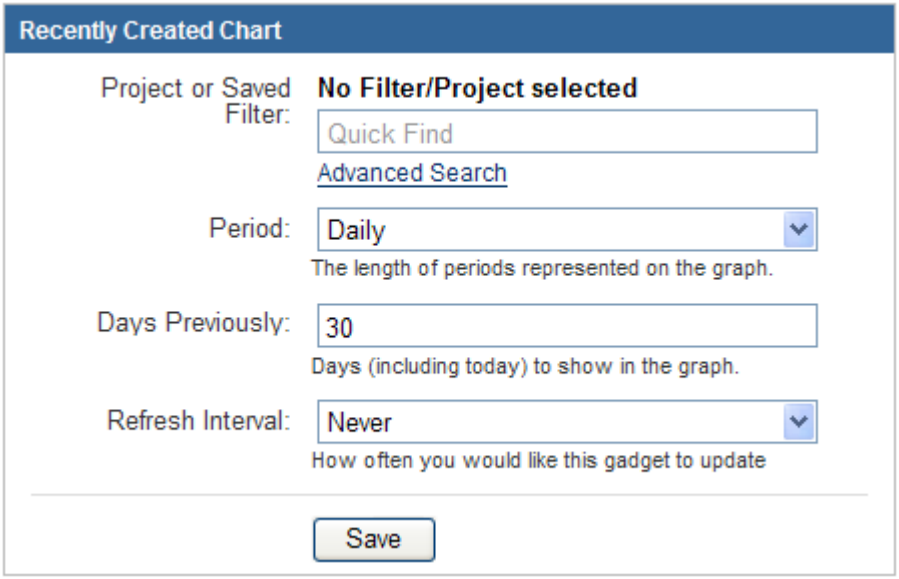
The **'Recently Created Issues'** gadget will appear as follows on the dashboard:



Click the **'more detail'** link to go to the full-size report and data table.

Adding the 'Recently Created Issues' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Recently Created Issues**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Recently Created Issues gadget will appear on your dashboard as follows, ready for you to configure:



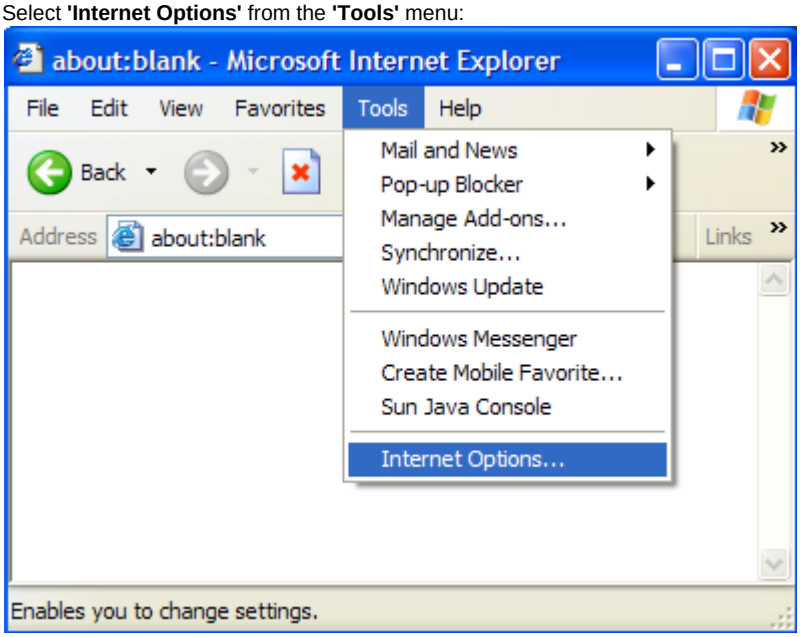
   a. '**Project or Saved Filter**' — start typing the name of the project or filter, or click the '**Advanced Search**' link to search for a project or filter.
   b. '**Title**' — type a heading for this gadget.
   c. '**Period**' — select the timeframe on which the chart will be based: **'Hourly'** / **'Daily'** / **'Weekly'** / **'Quarterly'** / **'Yearly'**
   d. '**Days Previously**' — type the number of days' worth of data (counting backwards from today) to be included in the chart.
   e. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

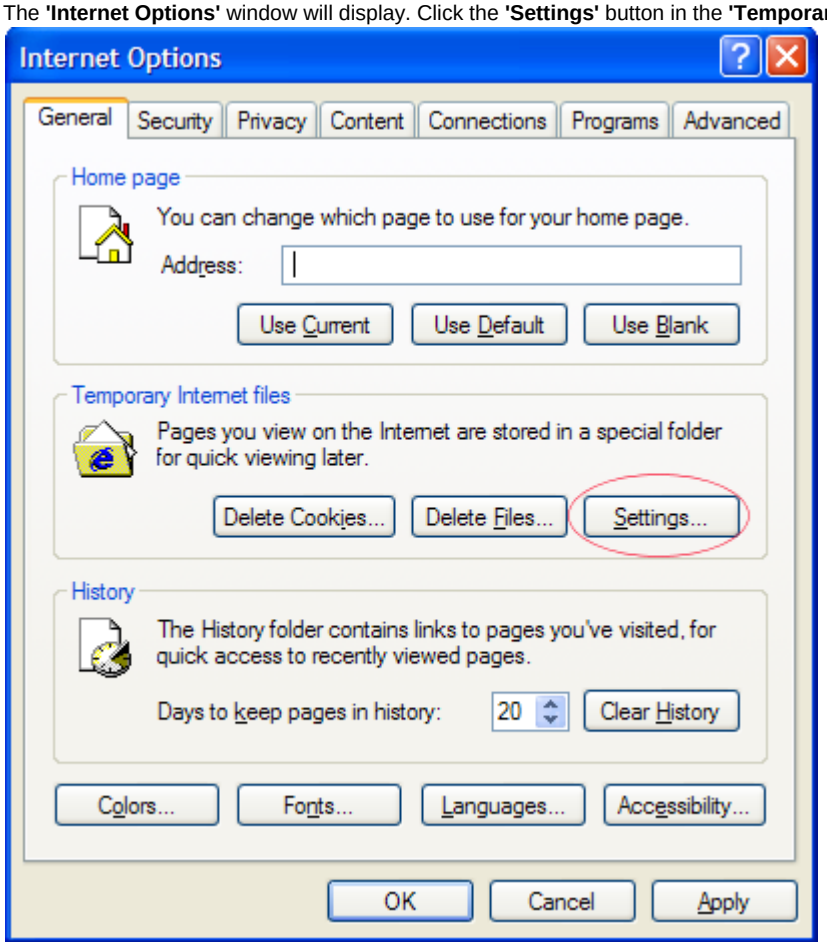
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

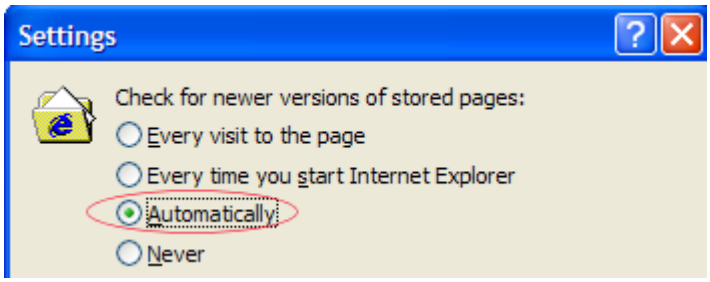
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.

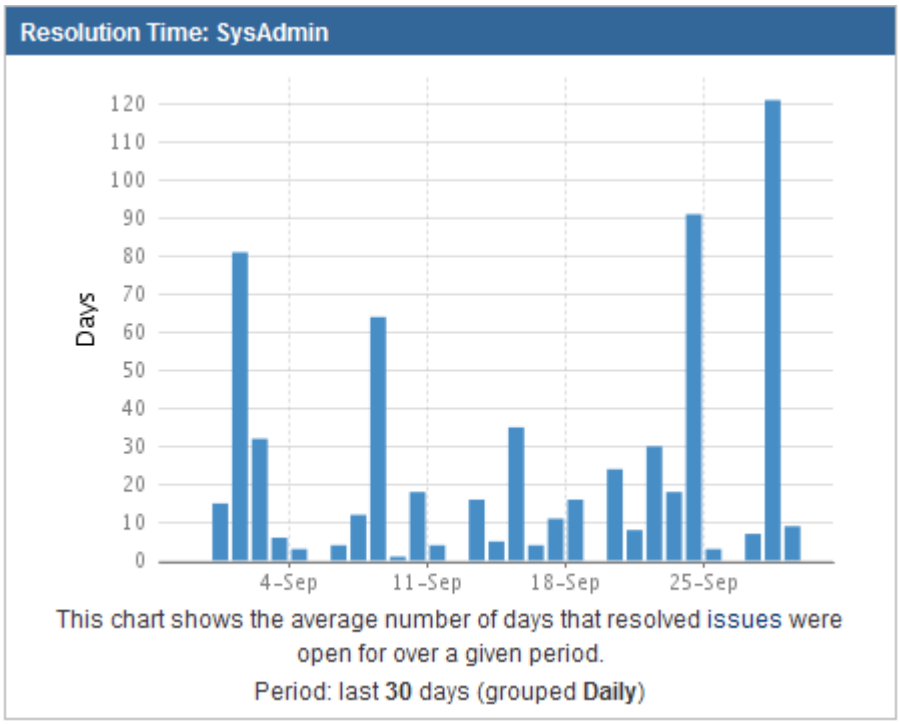


Adding the Resolution Time Gadget

The **'Resolution Time'** gadget displays a bar chart showing the average time taken to resolve issues. This is useful to show you the trends in resolution time. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years). The 'Resolution Time' is the difference between an issue's Resolution Date and Created date. If a Resolution Date is not set, the issue won't be counted in this gadget. The Resolution Date is the last date that the system Resolution field was set to any non-empty value.

What does it look like?

The **'Resolution Time'** gadget will appear as follows on the dashboard:

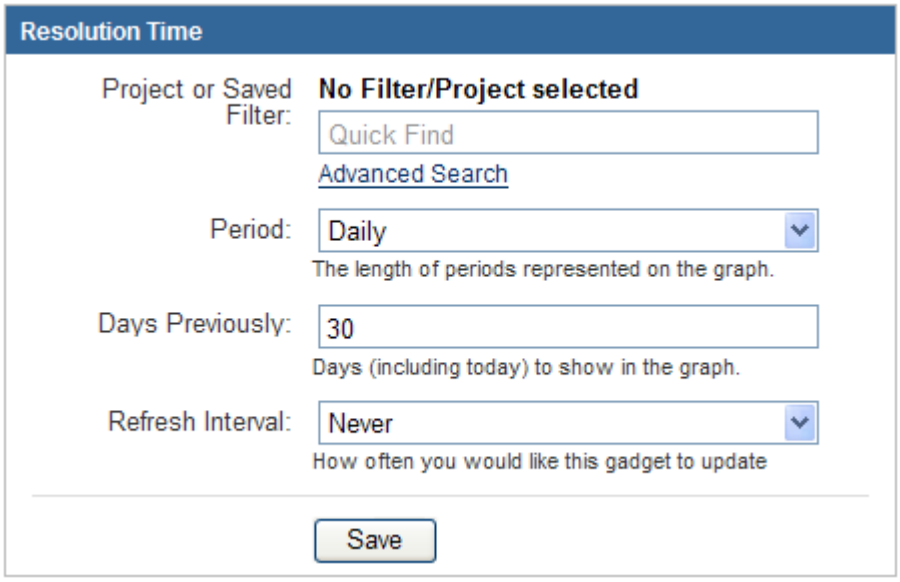


If you maximise the gadget, you can also view the data table on which the graph is based.

Adding the 'Resolution Time' Gadget to your Dashboard

To add the **'Resolution Time'** gadget to your dashboard:

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Resolution Time**' gadget and click the '**Add it Now**' button. Then click the '**Finished** ' button at the bottom of the Gadget Directory.
3. The Resolution Time gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Project or Saved Filter**' — start typing the name of the project or filter, or click the '**Advanced Search**' link to search for a project or filter.
b. **'Period'** — select the timeframe on which the chart will be based: **'Hourly'** / **'Daily'** / **'Weekly'** / **'Quarterly'** / **'Yearly'**
c. **'Days Previously'** — enter the number of days' worth of data (counting backwards from today) to be included in the chart.
d. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

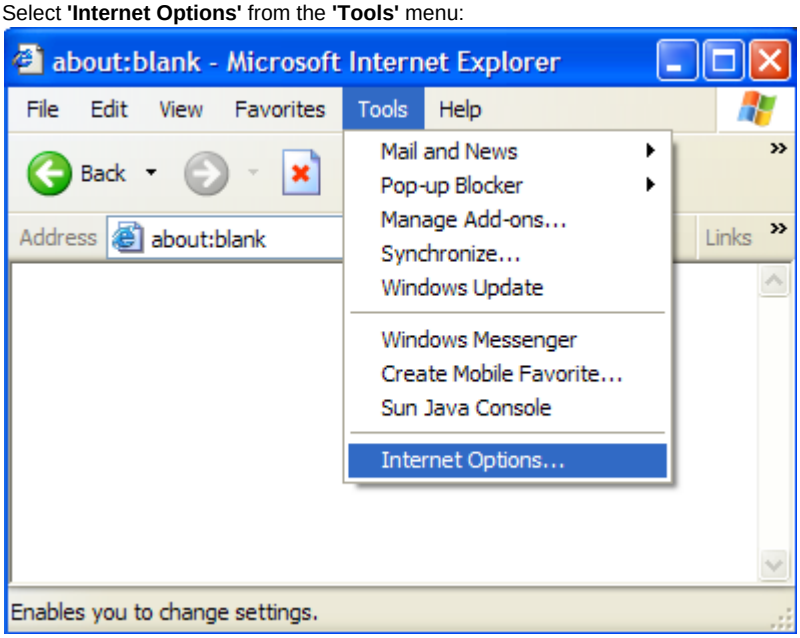
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

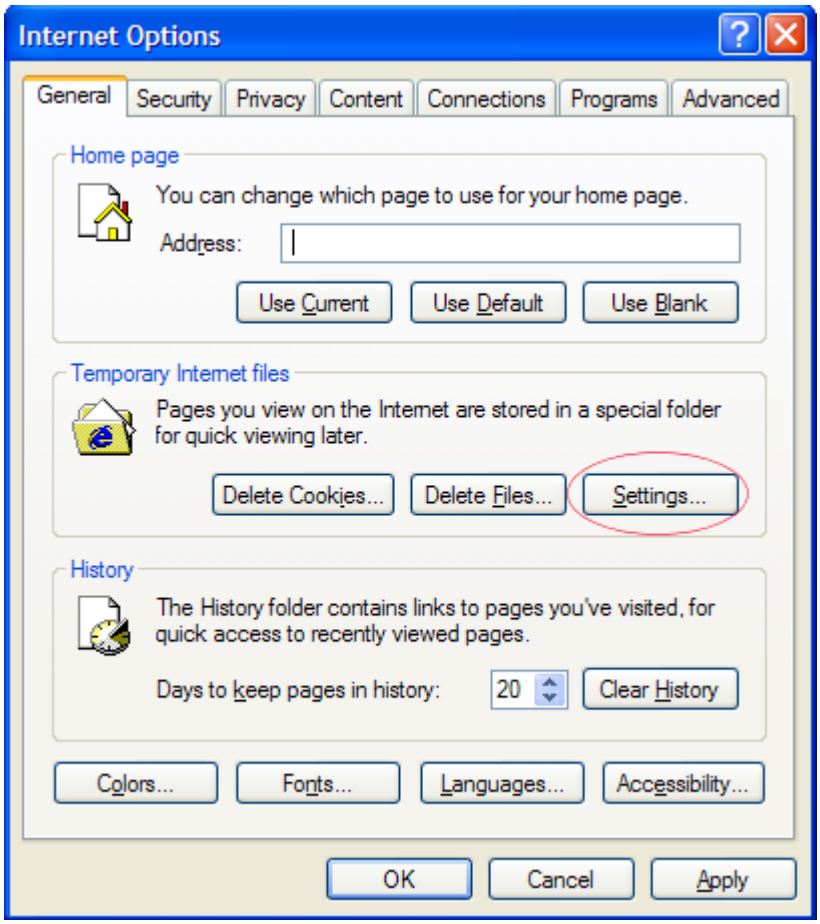
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

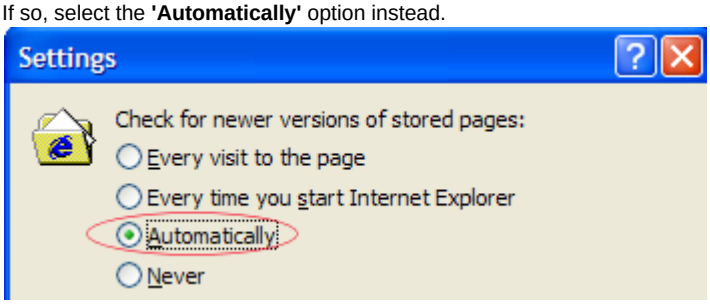
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.

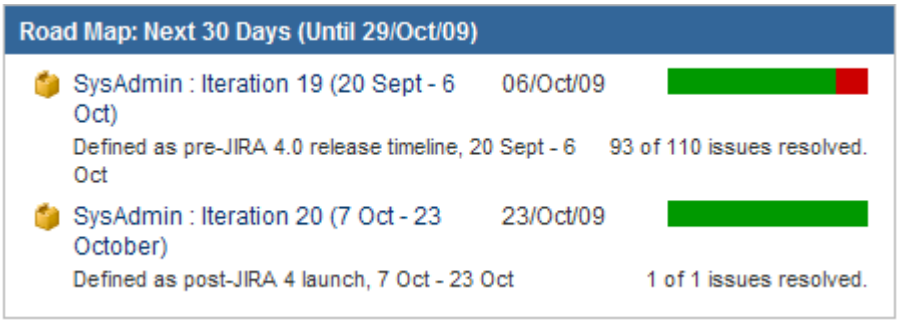


Adding the Road Map Gadget

The **Road Map** gadget shows versions which are due for release within a specified period of time, and a summary of progress made towards completing the issues in those versions.

What does it look like?

The **Road Map** gadget should appear as follows on the dashboard:



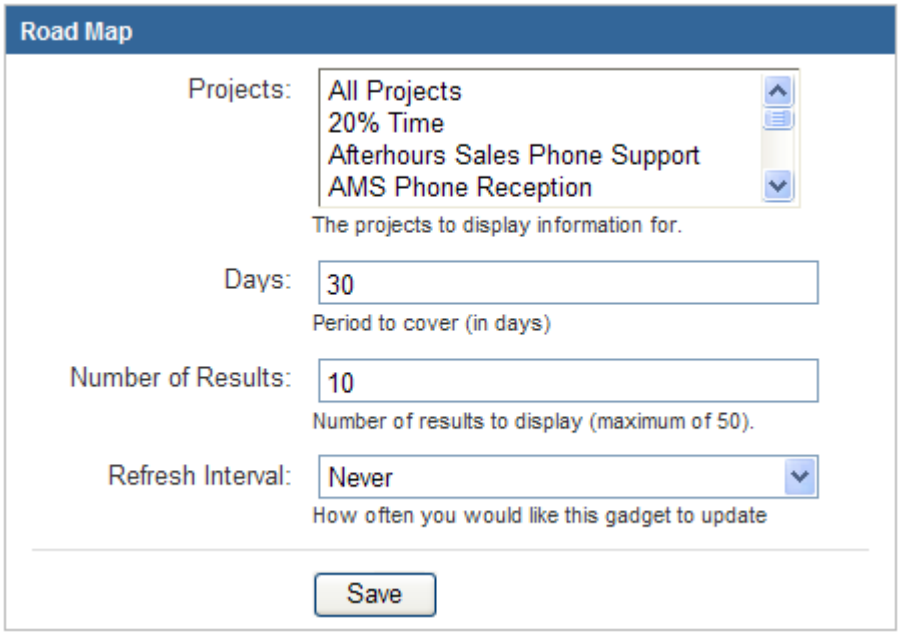
You can:

- Click the name of a project (e.g. 'Dove') to browse the project.
- Click the name of a version (e.g. 'Version 1') to browse the version.

- Click the progress bar (shown in red and/or green) to view the version's issues in the Issue Navigator.

Adding the 'Road Map' Gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Road Map**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Road Map gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Projects**' — select one or more projects (or 'All Projects') whose versions you wish to display in the gadget.
   b. '**Days**' — specify the period of time (in days) for which you wish to view versions due for release.
   c. '**Number of Results**' — type the maximum number of versions you wish the gadget to display per page.
   d. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

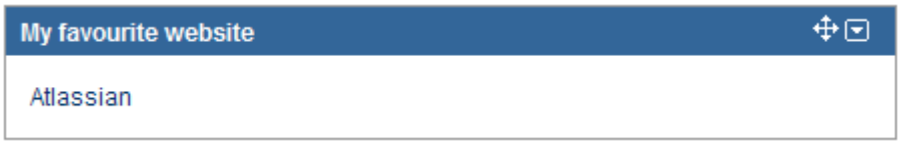
Adding the Text Gadget

The **Text** gadget displays your specified HTML text on the dashboard.

The Text gadget is disabled by default as it allows users to enter arbitary HTML. To enable the text gadget, please refer to Enabling Plugins section.

What does it look like?

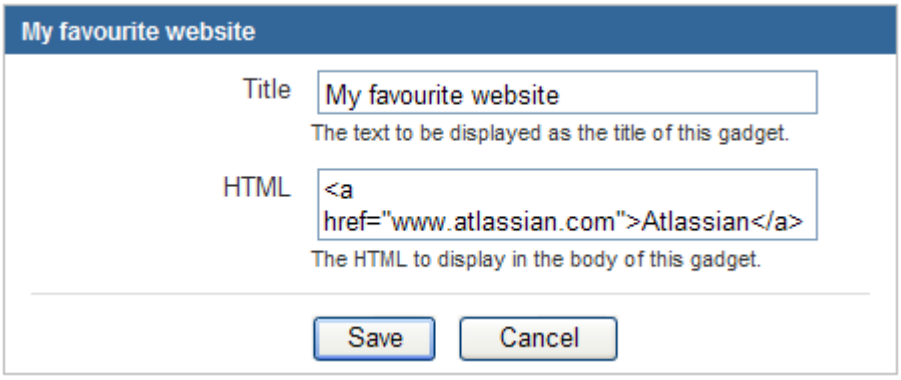
The **Text** gadget should appear as follows on the dashboard:



> This gadget is only available if your JIRA administrator has enabled the '**Text**' module in the '**JIRA Gadgets Plugin**'. It is disabled by default because it is a potential security risk, as it can contain arbitrary HTML which could potentially make your JIRA system vulnerable to XSS attacks.

Adding the 'Text' Gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Text**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Text gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Title**' — type a heading for this gadget.
b. '**Body**' — type the body text; this may include HTML.

4. Click the **'Save'** button.

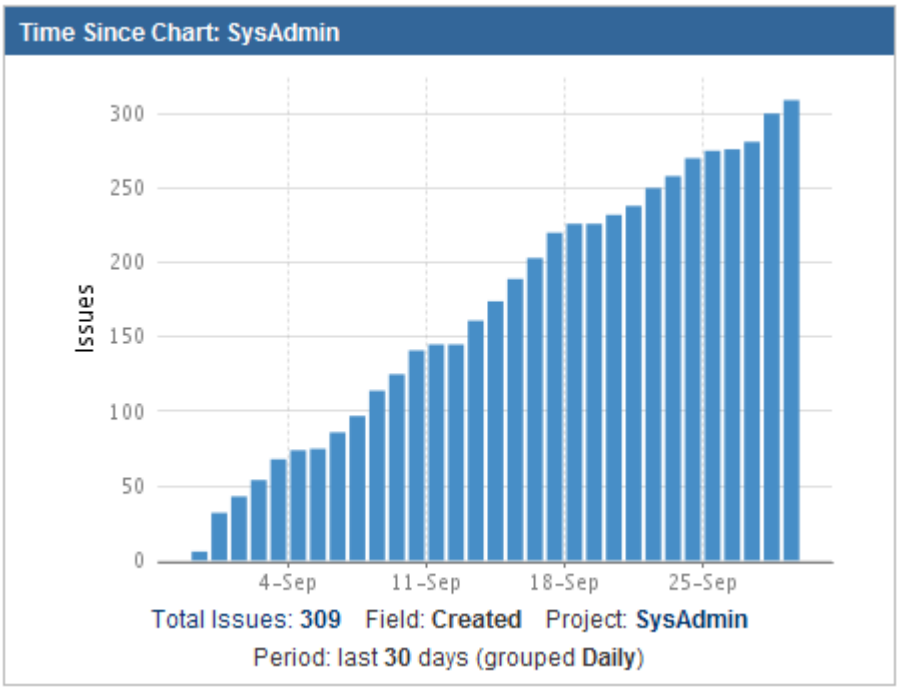
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Time Since Issues Gadget

The **'Time Since'** gadget displays a bar chart showing the number of issues for which your chosen date field (e.g. 'Created', 'Updated', 'Due', 'Resolved', or a custom field) was set on a given date. 'Resolved' here is the system Resolution Date field, which is the last date that the system Resolution field was set to any non-empty value. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).

What does it look like?

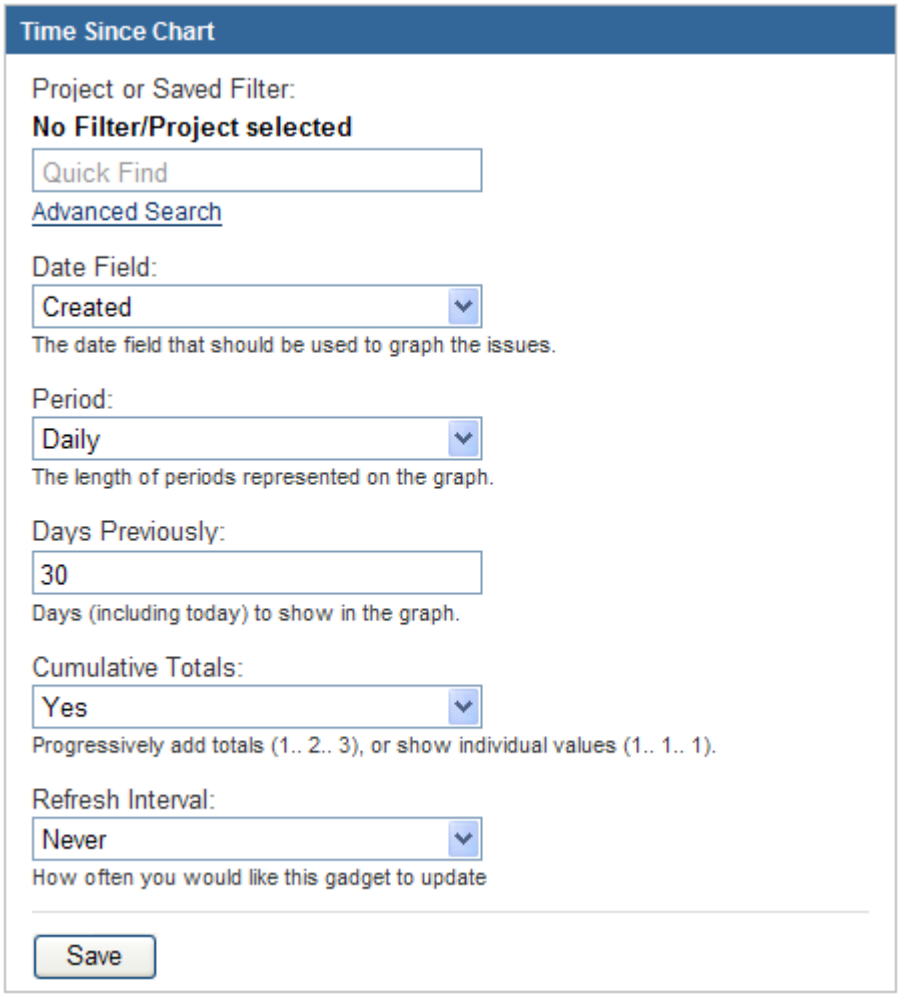
The **'Time Since'** gadget will appear as follows on the dashboard:



Click the **'more detail'** link to go to the full-size report and data table.

Adding the 'Time Since' Gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Time Since**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Time Since gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Project or Saved Filter**' — start typing the name of the project or filter, or click the '**Advanced Search**' link to search for a project or filter.
b. **'Date Field'** — select the date in which you are interested (e.g. **'Created'**, **'Updated'**, **'Due'** *, **'Resolved'**, or a custom field of type 'Date').
   *Note: only available if time tracking has been enabled by your JIRA administrator.
c. **'Period'** — select the timeframe on which the report will be based: **'Hourly'** / **'Daily'** / **'Weekly'** / **'Quarterly'** / **'Yearly'**
d. **'Days Previously'** — enter the number of days' worth of data (counting backwards from today) to be included in the report.
e. **'Cumulative Totals?'** — choose either:
   - **'Yes'** to progressively add data to the preceding column; or
   - **'No'** to show just a single value in each column.
f. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

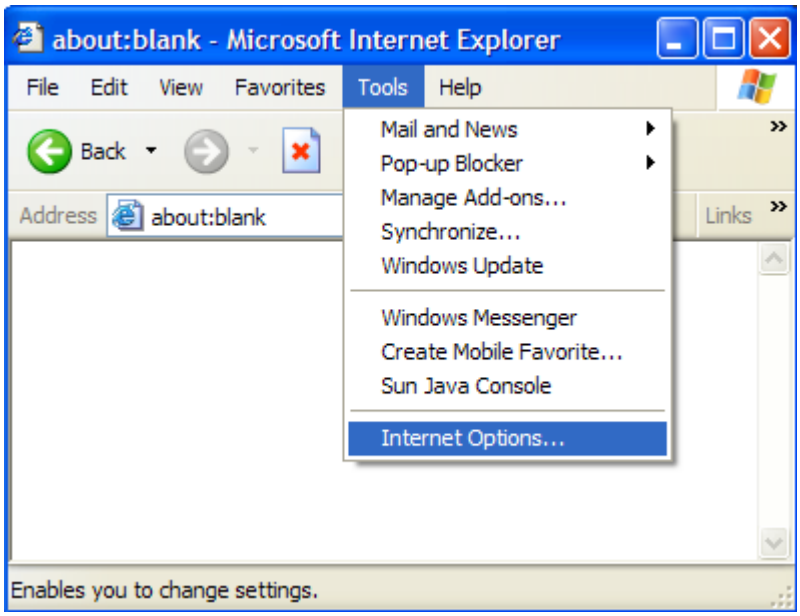
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

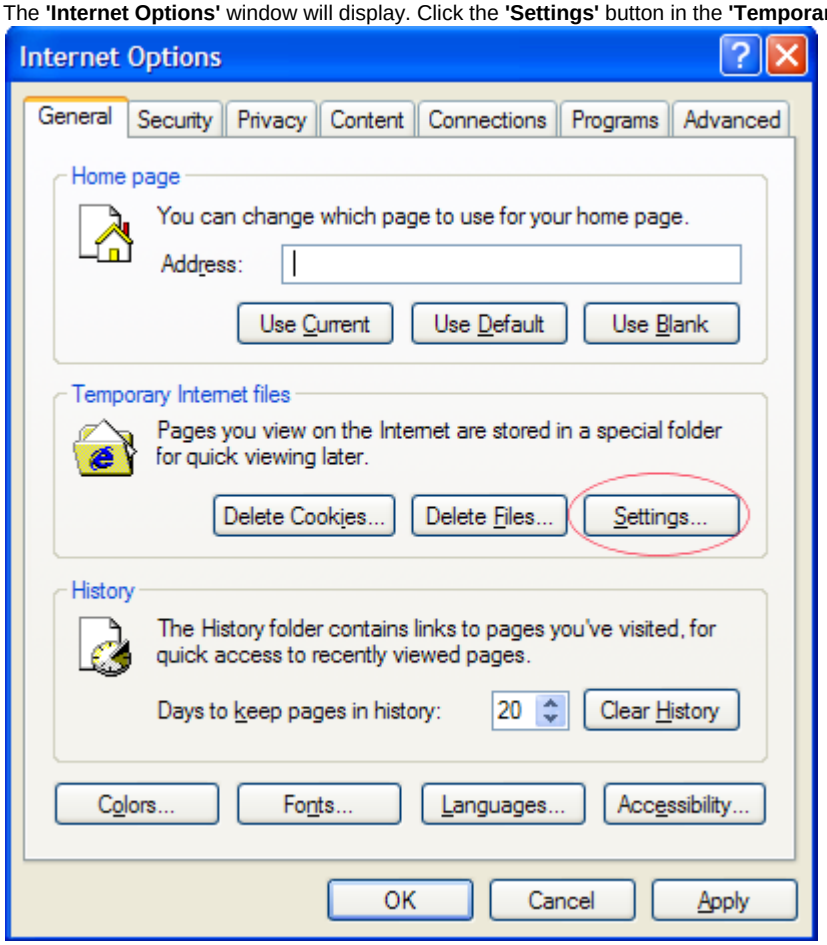
Configuring your Internet Explorer cache settings

If you use Internet Explorer, you will need to configure your browser to be able to print pages with charts correctly:

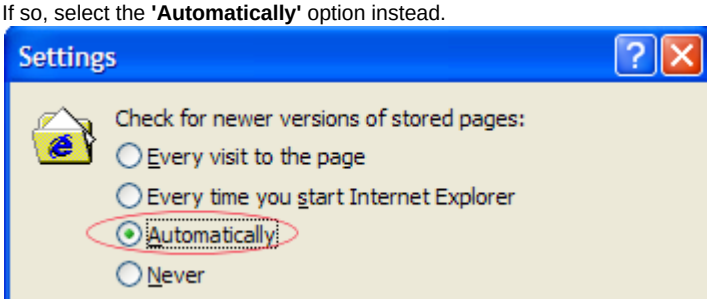
1. Select **'Internet Options'** from the **'Tools'** menu:



2. The **'Internet Options'** window will display. Click the **'Settings'** button in the **'Temporary Internet files'** (i.e. cache) section:



3. The **'Settings'** window will display. Ensure that you have do not have the **'Every visit to the page'** (i.e. no caching) option selected. If so, select the **'Automatically'** option instead.



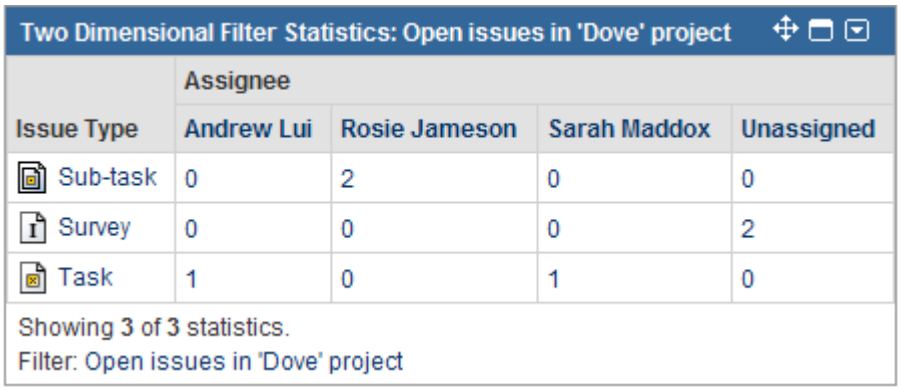
Adding the Two-Dimensional Filter Statistics Gadget

The **Two Dimensional Filter Statistics** gadget displays statistical data based on a specified issue filter, in a configurable table format.

For example, you could create a filter to retrieve all open issues in a particular project. You can then configure the gadget to display the statistical data on this collection of issues, in a table with configurable axes — e.g. Assignee versus Issue Type.

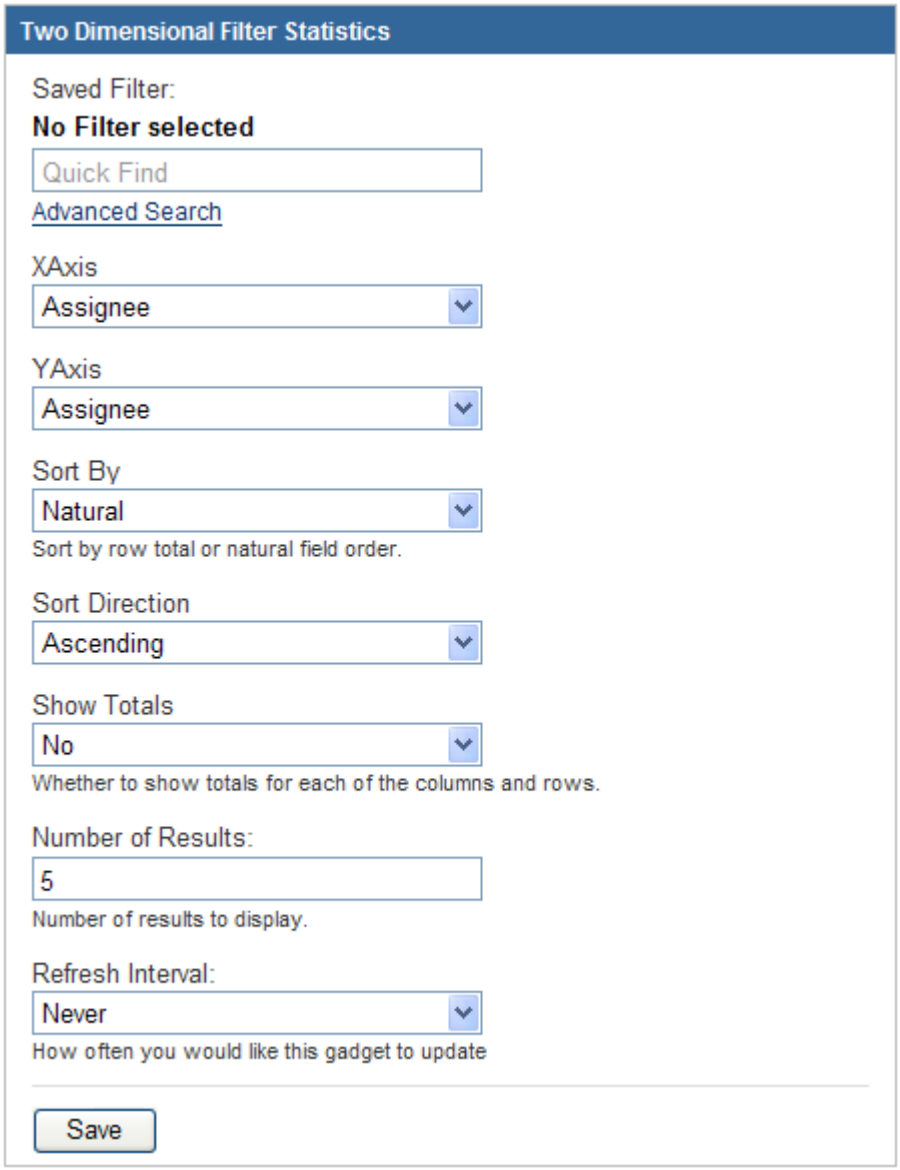
What does it look like?

The **Two Dimensional Filter Statistics** gadget should appear as follows on the dashboard:



Adding the Two Dimensional Filter Statistics gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Two Dimensional Filter Statistics**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Two Dimensional Filter Statistics gadget will appear on your dashboard as follows, ready for you to configure:



a. '**Saved Filter**' — start typing the name of the filter, or click the '**Advanced Search**' link to search for a filter/select one of your favourite filters/select a filter that you have created.
b. '**X Axis**' — select an issue field[1] on which the X-axis will be based.
c. '**Y Axis**' — select an issue field[1] on which the Y-axis will be based.
d. '**Sort By**' — select how to sort the values of your selected field:
    - '**Natural**' — this will use the field's native sorting order, e.g. for the "Assignee" field, the assignee names would be sorted alphabetically.
    - '**Total**' — this will sort by the number of issues that match each value, e.g. for the "Assignee" field, the assignee names would be sorted by the number of issues assigned to each person.
e. '**Sort Direction**' — select whether the field values should be sorted in **Ascending** or **Descending** order.
f. '**Show Totals**' — select whether to show row/column totals.
g. '**Number of Results**' — type the maximum number of rows/columns that you want the gadget to display per page.
h. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

[1] If choosing a custom field, please note that the following types of custom fields are supported by this gadget out-of-the-box: 'Group picker', 'Multi select', 'User picker', 'Select list' and 'Version picker'.

Adding the Voted Issues Gadget

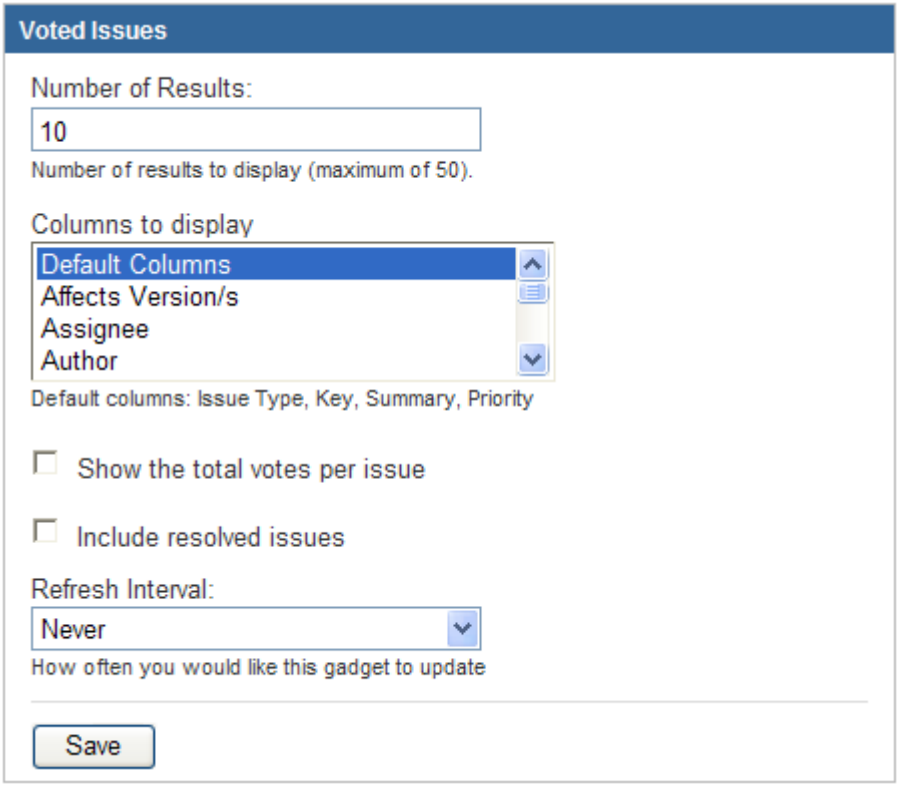
The **Voted Issues** gadget shows issues for which you have voted.

What does it look like?

The **Voted Issues** gadget should appear as follows on the dashboard:

Adding the 'Voted Issues' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Voted Issues**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Voted Issues gadget will appear on your dashboard as follows, ready for you to configure:



   a. **'Number of results'** — specify the maximum number of issues you wish the gadget to display per page.
   b. '**Columns to display**' — select the column(s) (i.e. issue fields) that you want the gadget to display, or choose '**Default Columns**' to display Issue Type, Key, Summary and Priority.
   c. **'Show total votes'** — select this if you wish the gadget to display the number of people who have voted for each issue.
   d. **'Show resolved issues'** — select this if you wish the gadget to display all issues on which you have ever voted. Leave it unselected if you wish the gadget to only display unresolved issues.
   e. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Watched Issues Gadget

The **Watched Issues** gadget shows issues which you are watching.

What does it look like?

The **Watched Issues** gadget should appear as follows on the dashboard:

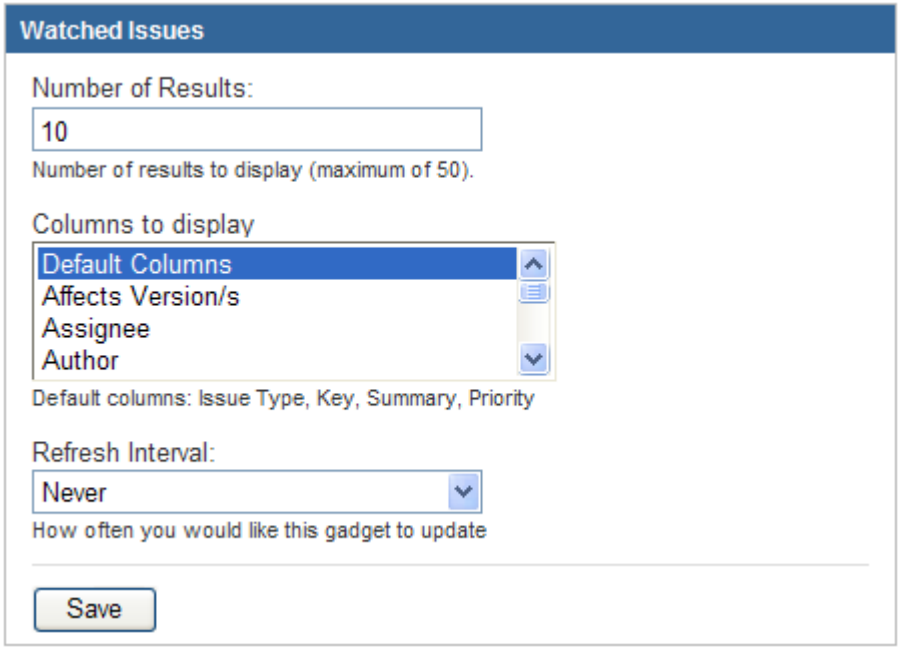
Watched Issues

T	Key	Summary	Pr	Status
	UI-1003	ALL Cover images for PDF exports of Atlassian Product Documentation		Closed
	ADM-5023	URL redirect from WAC to CAC required for "Upgrading JIRA" for JIRA 4.0		Closed
	ADM-3467	Request to import a Confluence space backup to CAC		Closed
	ADM-3420	Andrew Lui and Rosie Jameson are unable to access Orion via Putty		Closed
	ADM-3177	Admin access to Extranet JIRA		Closed
	ADM-2971	Please create email box, eFax number and EACJ project for Documentation ACLA		Closed
	ADM-876	please update the Page Layout on the new 'FISHEYE013' space		Closed
	ADM-823	please give Andrew Lui rights to SVN; add him to SVN docs watchers; and add him to the orion sudo file		Closed
	ADM-699	User on CAC for documentation export		Closed
	ADM-568	Move wireless network hub from top of bookshelf on CornX level 1		Closed

Displaying issues **1** to **10** of **11** matching issues. **1** 2 Next >>

Adding the 'Watched Issues' Gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Watched Issues**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Watched Issues gadget will appear on your dashboard as follows, ready for you to configure:



   a. '**Number of results**' — type the maximum number of issues that you want the gadget to display per page.
   b. '**Columns to display**' — select the column(s) (i.e. issue fields) that you want the gadget to display, or choose '**Default Columns**' to display Issue Type, Key, Summary and Priority.
   c. '**Refresh Interval**' — select how often you want the gadget to update the displayed activity (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the

gadget.

Adding the Heat Map Gadget

The **Heat Map** gadget displays the relative weighting of values of a specified field in issues returned from a specified project or saved filter.

For instance, the gadget can be configured to display a heat map of the popularity of the different priorities of issues in a particular project.

What does it look like?

The **Heat Map** gadget should appear as follows on the dashboard:

Adding the 'Heat Map' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Heat Map**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Heat Map gadget will appear on your dashboard.
    a. '**Project or Saved Filter**' — start typing the name of the project or filter, or click the '**Advanced Search**' link to search for a project or filter.
    b. '**Statistic Type**' — select the field (e.g. Assignee; Priority; etc) on which the issues will be grouped.
    c. '**Refresh Interval**' — select how often you want the gadget to update (never / every 15 minutes / every 30 minutes / every hour / every two hours).
4. Click the **'Save'** button.

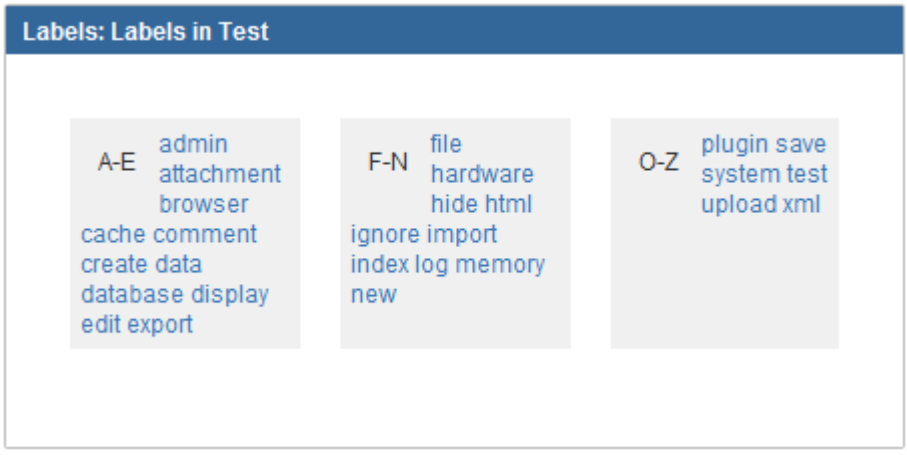
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Adding the Labels Gadget

The **Labels** gadget displays a list of all the labels in a specified project.

What does it look like?

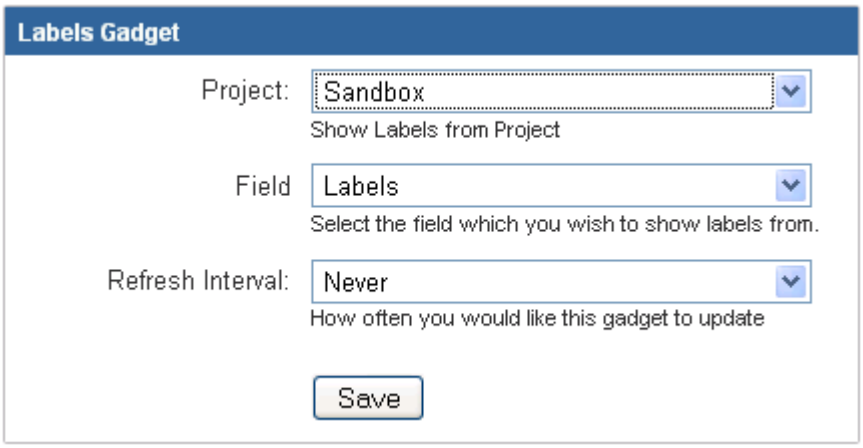
The **Labels** gadget should appear as follows on the dashboard:



You can click any label to go to the issue Navigator and view the issues which have that label.

Adding the 'Labels' gadget to your Dashboard

1. Go to your JIRA dashboard and click '**Add Gadget**'.
2. The '**Gadget Directory**' will appear. Locate the '**Labels**' gadget and click the '**Add it Now**' button. Then click the '**Finished**' button at the bottom of the Gadget Directory.
3. The Labels gadget will appear on your dashboard, ready for you to configure:



a. '**Project**' — select the name of the project in which you are interested.
b. '**Labels**' — select the field in which you are interested. The list will include the field 'Labels', plus any other custom fields of type 'Label' that have been defined by your JIRA administrator.
   If you are using GreenHopper, you may want to choose the Labels field called 'Epics'.
c. '**Refresh Interval**' — select how often you want the gadget to update (never / every 15 minutes / every 30 minutes / every hour / every two hours).

4. Click the **'Save'** button.

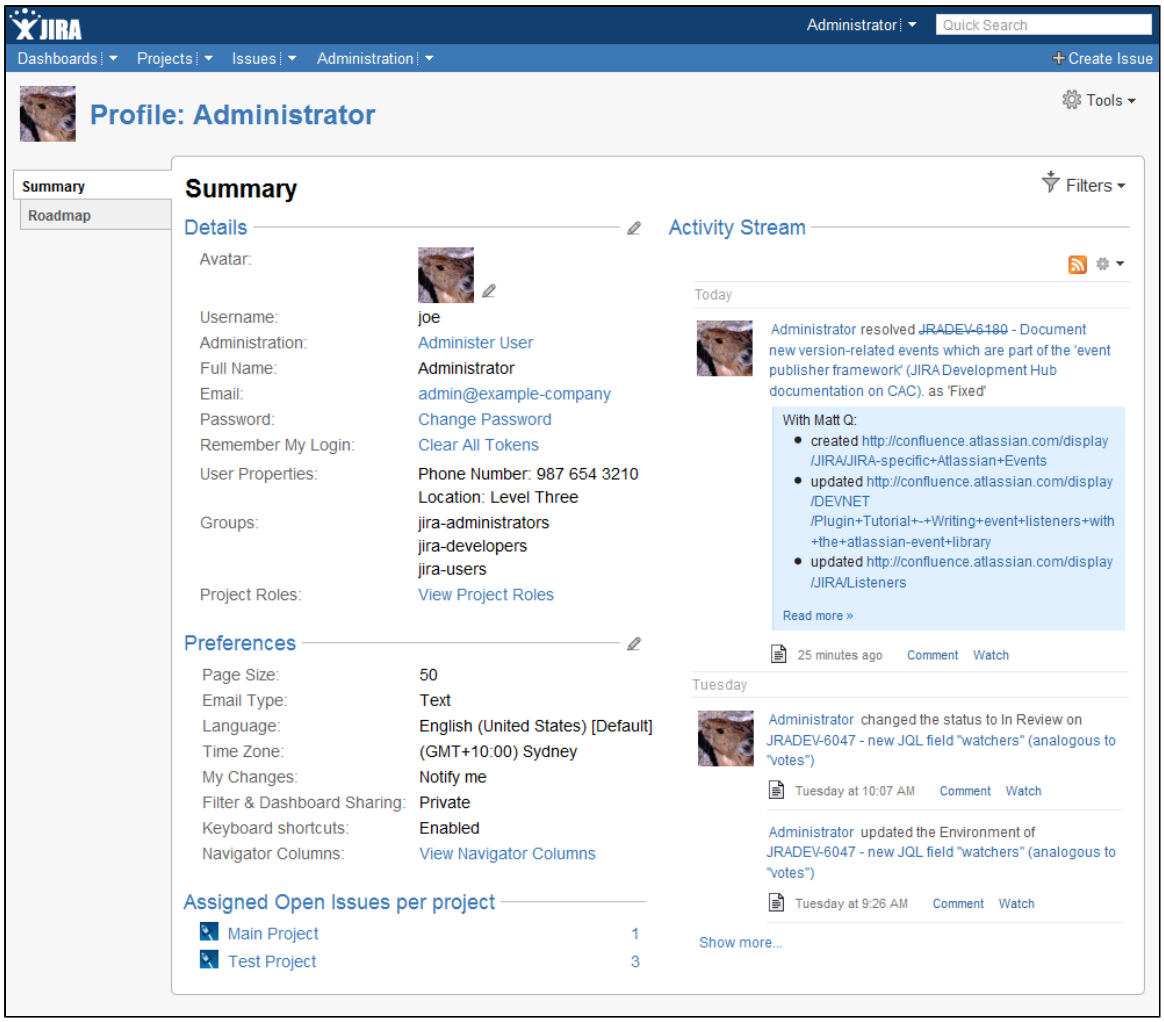
To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behaviour of the gadget.

Managing your User Profile

Your JIRA user profile is where you specify your JIRA settings (e.g. your email address, and the format in which you would like to receive email notifications). It also contains useful links to a number of personalised reports.

To view your JIRA user profile:

- Click your user name at the top-right of the browser window. The **Summary** tab page of your **User Profile** will be displayed:



On this page:

- Using the Summary tab
  - Details
  - Preferences
  - Assigned Open Issues per Project
  - Activity Stream
  - Filters
- Using the Roadmap tab
- Using the Tools menu

Using the Summary tab

The **Summary** tab page shows your personal details registered in JIRA, your personal JIRA preferences, the number of open issues assigned to you by project and a list of your recent activity.

Details

In the **Details** section on the **Summary** tab page, you can do the following:

- Click the **edit** icon at the top-right of the section to open the **Edit Profile**dialog box. From here, you can edit the following details:
  - **Full Name** – your display-name – that is, the name by which you are known in JIRA.
  - **Email** – the email address to which your JIRA notifications will be sent.
  - Click the '**Update**' button to save your changes.
- Click the **add user avatar** icon (or its adjacent **edit** icon) to add an avatar to your user profile.
  If you have already added a user avatar to your JIRA profile, that avatar will appear in place of the **add user avatar** icon above.
- Click the '**Administer User**' link to view or edit your user details in JIRA's administration area.

This option is only available to if you are a **JIRA Administrator**.

- Click the **email address** to send an email to that address via your registered email client application.
- Click the '**Change Password**' link to change your JIRA password.
- Click the '**Clear All Tokens**' link to clear your 'Remember my login' tokens. This feature is useful if you have accessed JIRA in a public environment, selected the '**Remember by login...**' check box before logging in, but you may have forgotten to log out and do not wish others to access JIRA through your account. See Clearing 'Remember my login' Tokens for more information.
- Click the '**View Project Roles**' link to view or edit that user's project roles in JIRA's administration area.

  This option is only available to users who are **JIRA Administrators**.

> **Note**
>
> If your JIRA administrator has configured the user directory containing your account with external password management, the **Edit Profile** and **Change Password** links may not be available.

Preferences

In the **Preferences** section on the **Summary** tab page, you can do the following:

- Click the **edit** icon at the top-right of the section to open the **Updated User Preferences**dialog box. From here, you can edit the following details:
  - **Page Size** – The number of issues displayed on each Issue Navigator page. This field is mandatory and the default value is 50.
  - **Email Type** – The format (text or HTML) in which JIRA sends its outgoing email notifications.
  - **Language** – Your preferred language.
  - **Time Zone** – Your preferred time zone.
  - **My Changes** – Choose between making JIRA send you email notifications about issue updates made by either both you and other people (**Notify me**) or other people only (i.e. **Do not notify me**).
  - **Filter & Dashboard Sharing** – Choose between making JIRA share dashboards and filters with all other users by default upon creation (**Public**) or restrict
    dashboards and filters to your viewing only by default upon creation (**Private**).
  - **Keyboard Shortcuts** – Choose between making JIRA's Keyboard Shortcuts feature either **Enabled** or **Disabled**.
- Click '**View Navigator Columns**' to choose which fields appear in your Issue Navigator.

Assigned Open Issues per Project

In the **Assigned Open Issues per project** section on the **Summary** tab page, you can do the following:

- Click the name of the project (on the left) to browse that project's roadmap.
- Click the number of open issues (on the right) to display the Issue Navigator, which shows your list of open issues associated with the project on the left.

Activity Stream

In the **Activity Stream** on the right of the **Summary** tab page, you can:

- Click any item to jump to an issue or other activity in which you have recently participated. Your Activity Stream can include:
  - Issues in your local JIRA system.
  - Issues in another JIRA system (provided your administrator has configured a two-way Application Link).
  - Activity from another Atlassian application, such as:
    - document updates (from Confluence)
    - code commits (from FishEye)
    - code reviews (from Crucible)
    - builds (from Bamboo)

      Note that this requires your administrator to configure a two-way Application Link.
  - Activity from remote applications (note that your administrator will need to set this up via the REST API or the provider plugin API, or locally via Java).
- Click the RSS icon to generate an RSS feed of information that is relevant to you.
- Click the cog drop-down to refresh the displayed Activity Stream.

The Activity Stream is also available as a gadget.

Filters

Click the '**Filters**' menu at the top of the **Summary** tab page. From this menu you can:

- Click '**Assigned**' to list all issues that are assigned to you, irrespective of their current status.
- Click '**Assigned & Open**' to list the issues that are assigned to you and are unresolved.

- Click '**Assigned & In Progress**' to list the issues that are assigned to you and whose current status is **In Progress**.
- Click '**Reported**' to list the issues that were created by you, irrespective of their current status.
- Click '**Reported & Open**' to list the issues that were created by you and are unresolved.
- Click '**Voted**' to view the list of issues for which you have voted, irrespective of their current status.
- Click '**Voted & Open**' to view the list of issues for which you have voted and are unresolved.
- Click '**Watched**' to view the list of issues that you are watching, irrespective of their current status.
- Click '**Watched & Open**' to view the list of issues that you are watching and are unresolved.

Using the Roadmap tab

The **Roadmap** tab page shows your 'Personal Road Map' report, which provides quick access to work assigned to you across all projects.

On the **Roadmap** tab page, you can do the following:

- In the **Change Project** field, select a project to show a personal road map report for work assigned to you for that project. This is similar in functionality to browsing a project's roadmap, although the personal road map shows only issues assigned to you.
- Click the '**View global road map**' link to show all work required for that project.

Using the Tools menu

Click the '**Tools**' menu in the top right to open it. From this menu you can click '**View OAuth Access Tokens**' to view and edit your OAuth Tokens.

Related topics:

- To change your Dashboard layout, and add gadgets, see Customising the Dashboard
- To view and edit your issue filters, see Saving Searches ('Issue Filters')

Adding a User Avatar

A user avatar is used as the icon for your profile to illustrate your comments on an issue and your Hover Profile.

On this page:

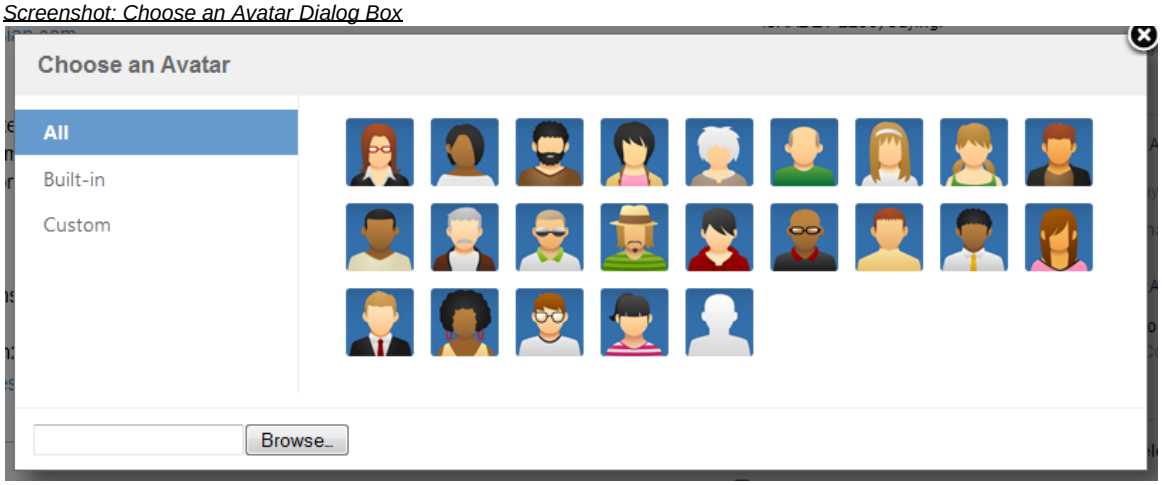
- Choosing a User Avatar
- Uploading a Custom Avatar

Choosing a User Avatar

To choose your user avatar:

1. Click your user name at the top-right of the page to open your User Profile at the **Summary** tab page.
2. In the **Details** section, click the **add user avatar** icon (or its adjacent **edit** icon). The **Choose an Avatar** dialog box opens.
   If you have already added a user avatar to your JIRA profile, that avatar will appear in place of the **add user avatar** icon above.

   Screenshot: Choose an Avatar Dialog Box



3. On the left of the dialog box, choose one of the following categories from which to select your user avatar:
   - **All** — Shows all built-in and custom user avatars.
   - **Built-in** — Shows only the user avatars which are pre-packaged with JIRA.

- **Custom** — Shows only the custom user avatars which you have uploaded (see below).

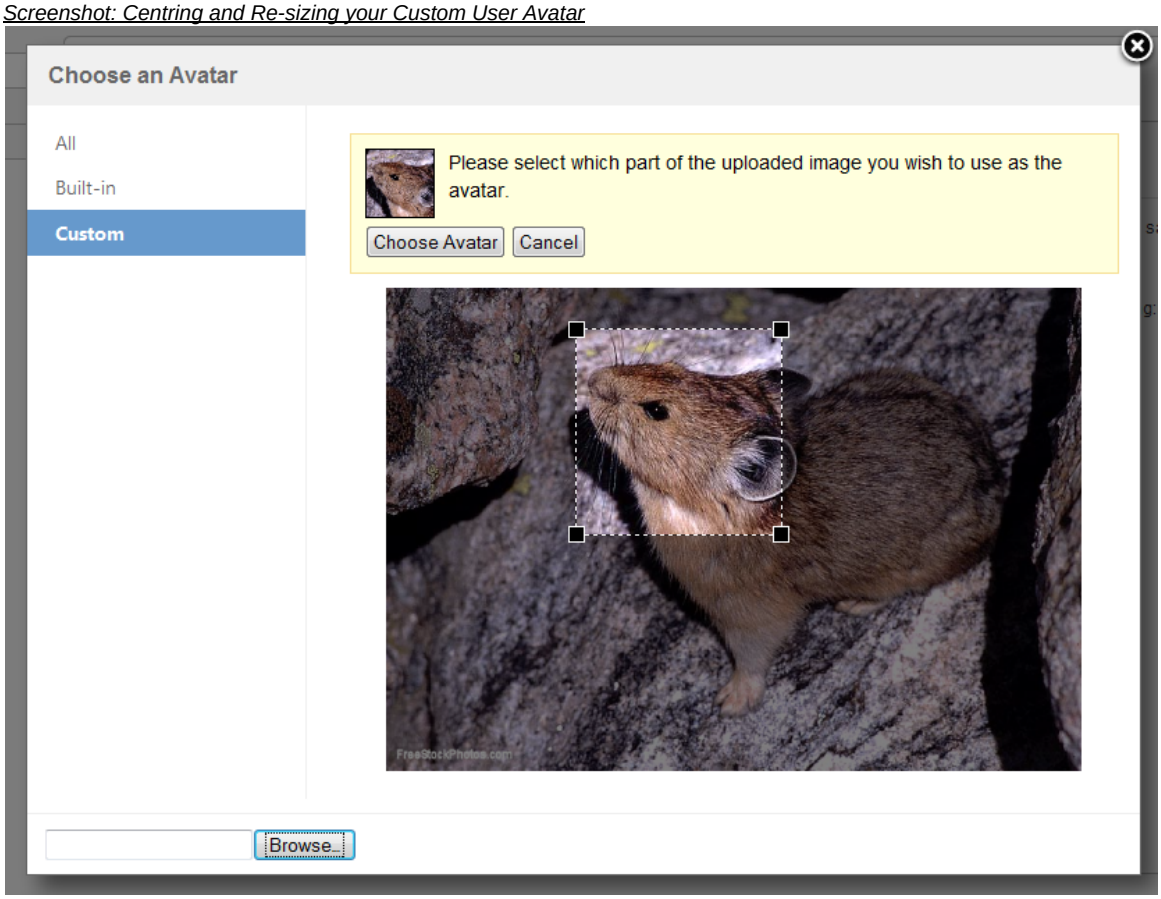
4. Click a user avatar on the right of the dialog box to select it.

Uploading a Custom Avatar

To create a 'custom' user avatar from an image file:

1. Click your user name at the top-right of the page to open your User Profile at the **Summary** tab page.
2. In the **Details** section, click the **add user avatar** icon (or its adjacent **edit** icon). The **Choose an Avatar** dialog box opens (see screenshot above).
   If you have already added a user avatar to your JIRA profile, that avatar will appear in place of the **add user avatar** icon above.
3. Click the '**Browse**' button and in the resulting dialog box, browse for and choose an image file.
4. Click and drag the centre of the superimposed square to select the centre of the new image.
   If desired, drag the corners of the square to re-size the area for your new image.

   Screenshot: Centring and Re-sizing your Custom User Avatar



5. Click the '**Choose Avatar**' button to create the custom user avatar.

- The image from your selected area will be cropped, re-sized to 48x48 pixels and saved.
- A separate 16x16 pixel version of your custom user avatar will be generated for use in comments.
- Custom user avatars can only be selected by the user who uploaded them.

Allowing OAuth Access

On this page:

- About OAuth Access Tokens
- Issuing OAuth Access Tokens
- Revoking OAuth Access Tokens
  - OAuth Access Token Table Details

About OAuth Access Tokens

OAuth access tokens allow you to:

- Use a JIRA gadget on an external, OAuth-compliant web application or website (also known as a 'consumer')

AND

- Grant this gadget access to JIRA data which is restricted or privy to your JIRA user account.

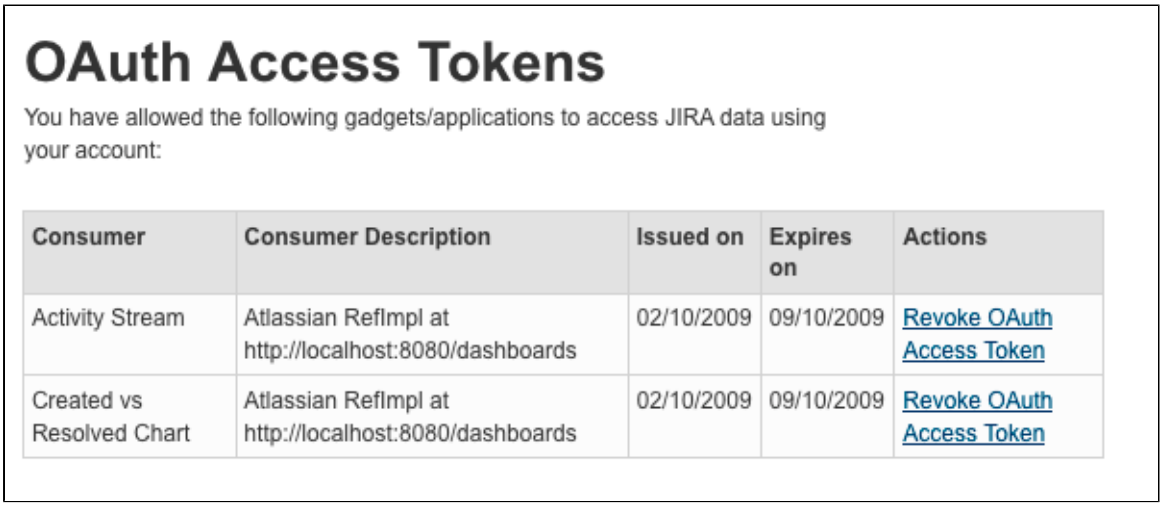
Before this can happen, your JIRA administrator must establish an OAuth relationship with this external web application or site by approving it as an OAuth consumer. For example, if you want to add a JIRA gadget to your Bamboo homepage and allow this gadget to access your restricted JIRA data, then your JIRA administrator must first approve Bamboo as an OAuth consumer.

Next, the JIRA gadget on the 'consumer' is granted access to your JIRA data via an 'OAuth access token', which acts as a type of 'key'. As long as the consumer is in possession of this access token, the JIRA gadget will be able to access JIRA data that is both publicly available and privy to your JIRA user account. You can revoke this access token at any time from your JIRA user account, otherwise, all access tokens expire after seven days. Once the access token is revoked or has expired, the JIRA gadget will only have access to publicly available data on your JIRA site.

An OAuth access token will only appear in your user profile if the following conditions have been met:

1. Your JIRA Administrator has established an OAuth relationship between your JIRA site and the consumer. JIRA Administrators should refer to Configuring OAuth for more information about establishing these OAuth relationships.
2. You have accessed a JIRA gadget on a consumer and have allowed this gadget access to your JIRA data. See Issuing OAuth Access Tokens, below for details on this process.

Screenshot: Viewing your OAuth Access Tokens



OAuth Access Tokens

You have allowed the following gadgets/applications to access JIRA data using your account:

Consumer	Consumer Description	Issued on	Expires on	Actions
Activity Stream	Atlassian RefImpl at http://localhost:8080/dashboards	02/10/2009	09/10/2009	Revoke OAuth Access Token
Created vs Resolved Chart	Atlassian RefImpl at http://localhost:8080/dashboards	02/10/2009	09/10/2009	Revoke OAuth Access Token

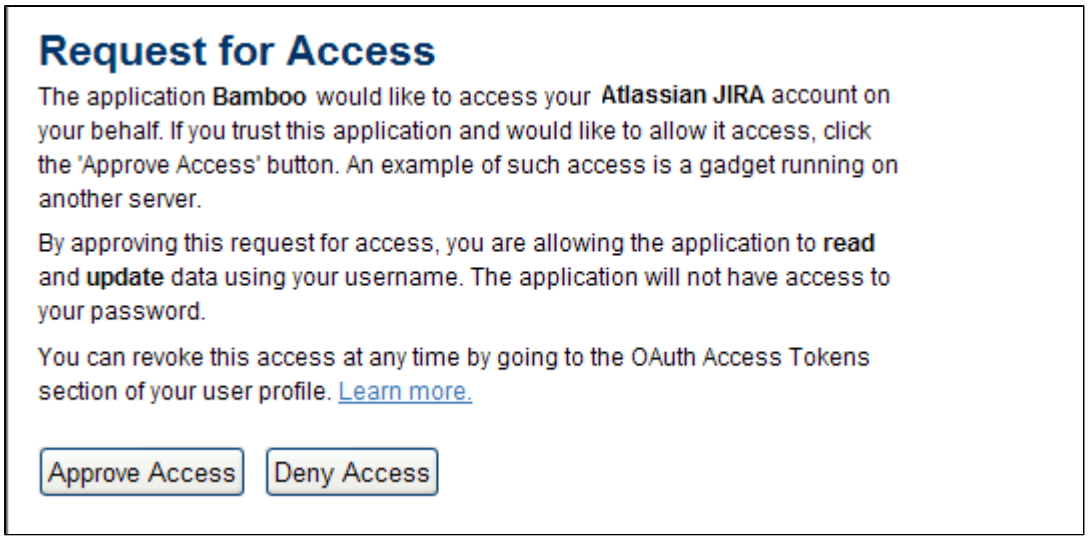
Issuing OAuth Access Tokens

An OAuth access token is issued by JIRA to provide one of its gadgets on a consumer, access to your JIRA data (that is, data which is restricted to your JIRA user account).

To allow a JIRA gadget on a consumer, access your JIRA data,

1. When you are using a JIRA gadget on a consumer (such as Bamboo) and this gadget requires access to your JIRA data, you will first be prompted to log in to JIRA (if you have not already done so).
2. Once you have logged in to JIRA, you will be prompted with a '**Request for Access**' message:

   Screenshot: Request for Access Message



At this point, JIRA is preparing to issue the JIRA gadget (on the consumer) with an OAuth access token.

3. To grant the gadget access to your JIRA data, click the '**Approve Access**' button. The consumer application will receive the OAuth access token from your JIRA site. This access token is specific to this gadget and as long as the token resides with the gadget, your gadget will have access to your JIRA data.

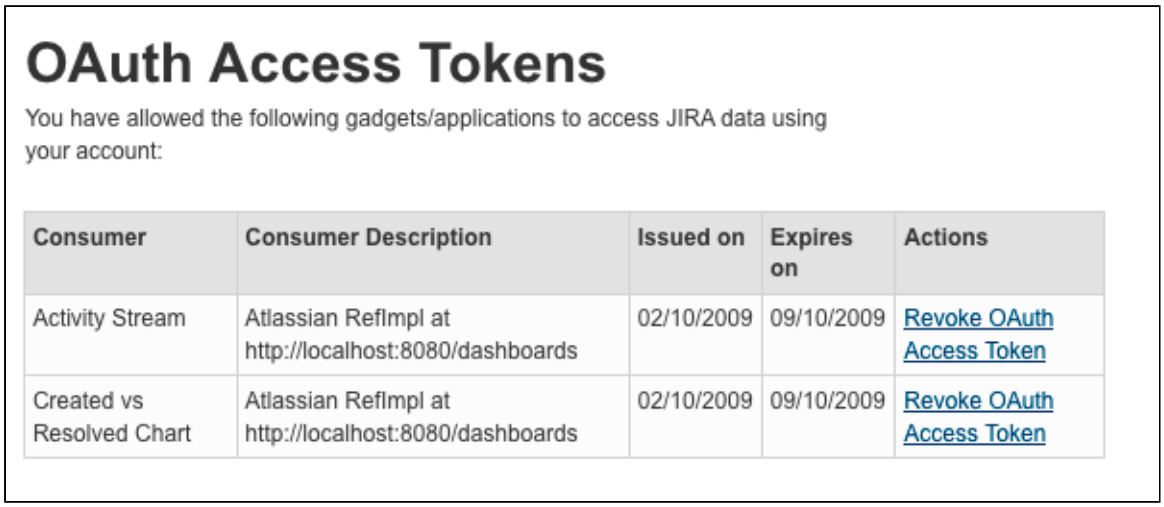
Revoking OAuth Access Tokens

You can revoke an OAuth access token to deny a JIRA gadget on a consumer access to JIRA data which is restricted to your JIRA user account. You can only revoke OAuth access tokens that you have allowed JIRA to issue previously.

To prevent a JIRA gadget on a consumer, from accessing your JIRA data,

1. Click your user name at the top-right of the page to open your User Profile at the 'Summary' tab page.
2. Click the '**Tools**' menu and select the '**View OAuth Access Tokens**' menu item.
3. The '**OAuth Access Tokens**' page will be displayed.

   Screenshot: Viewing your OAuth Access Tokens



Your list of OAuth access tokens is presented in a tabular format, with each access token presented in separate rows and each property of these tokens presented in a separate columns. Refer to the OAuth Access Token Table Details section below for more information about this table.

4. Locate the JIRA gadget and its associated consumer application whose OAuth access token you wish to revoke and click its 'Revoke OAuth Access Token' link in the 'Actions' column.
5. You may be prompted to confirm this action. If so, click the '**OK**' button.

The gadget's access token is revoked and the JIRA gadget on the consumer will only have access to publicly available JIRA data.

OAuth Access Token Table Details

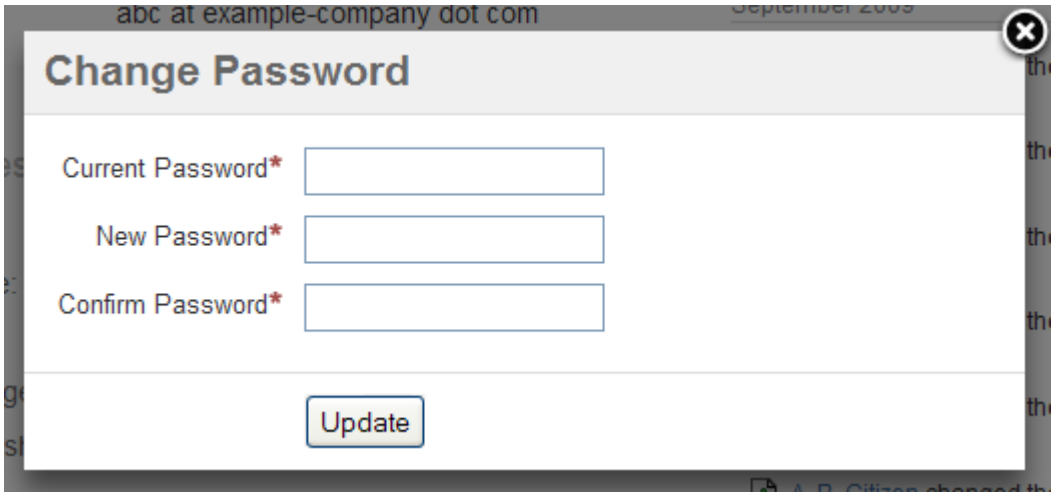
Column Name	Description
Consumer	The name of the JIRA gadget that was added on the consumer.
Consumer Description	A description of this consumer application. This information would have been obtained from the consumer's own OAuth settings when an OAuth relationship was established between JIRA and that consumer. If the consumer is another Atlassian application, this information is obtained from the Consumer Info tab's 'Description' field of the OAuth Administration settings. The application's administrator can customise this Consumer Info detail.
Issued On	The date on which the OAuth access token was issued to the consumer by JIRA. This would have occurred immediately after you approved this gadget access to your JIRA data (privy to your JIRA user account).
Expires On	The date when the OAuth access token expires. This is seven days after the 'Issued On' date. When this date is reached, the access token will be automatically removed from this list.
Actions	The functionality for revoking the access token.

Changing your Password

To change your JIRA password:

1. Click your user name at the top-right of the page to open your User Profile at the **Summary** tab page.
2. In the **Details** section, click the '**Change Password**' link. The **Change Password** dialog box opens.
3. Type your old password into the **Current Password** field, and type your new password into the **New Password** and **Confirm Password** fields:

   Screenshot: Change Password Dialog Box



4. Click the '**Update**' button.

> If your JIRA administrator has configured the user directory containing your account with external password management, the '**Change Password**' link will not be available.

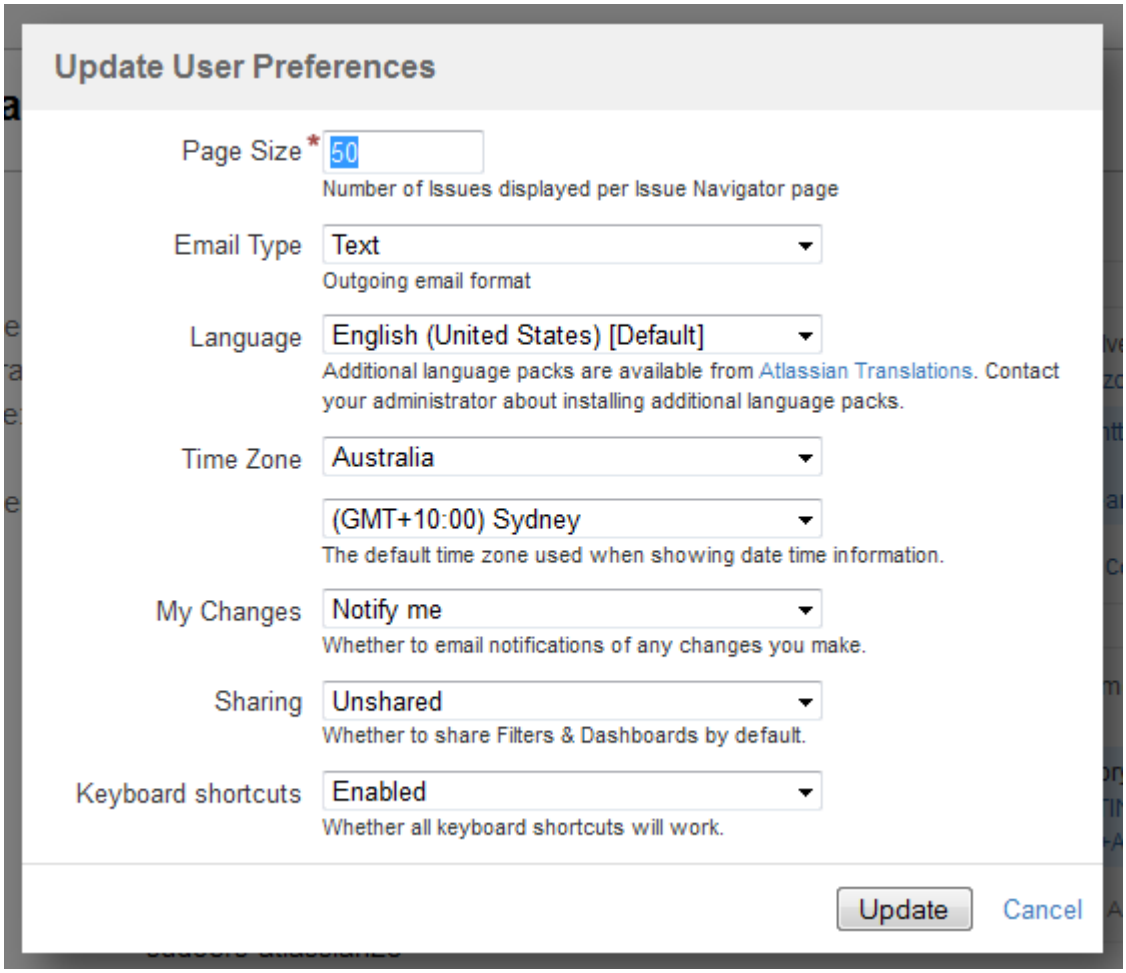
Choosing a Language

The default language is set by your JIRA administrator (see Configuring JIRA Options), but you can personalise your JIRA account to use a language of your choice.

To choose a language:

1. Click your user name at the top-right of the page to open your User Profile at the 'Summary' tab page.
2. In the 'Preferences' section, click the **edit** icon at the top-right. The 'Updated User Preferences' dialog box opens.
3. Select your language from the '**Language**' drop-down list:

Screenshot: Update User Preferences Dialog Box



4. Click the '**Update**' button.

Obtaining Additional Languages

If your particular language is not available from the 'Language' dropdown menu, contact your JIRA System Administrator to request them to install your particular language pack for JIRA.

For more information, see Translating JIRA and if necessary, ask your JIRA System Administrator to refer to the Managing JIRA's Plugins page for instructions on how to install JIRA plugins (including JIRA language packs).

Using Hover Profile

Hover Profile is a convenient popup balloon that provides quick access to key information about other JIRA users throughout the JIRA interface and issues they have been working on.

On this page:

- Accessing Hover Profile
- Using the Hover Profile Popup Balloon

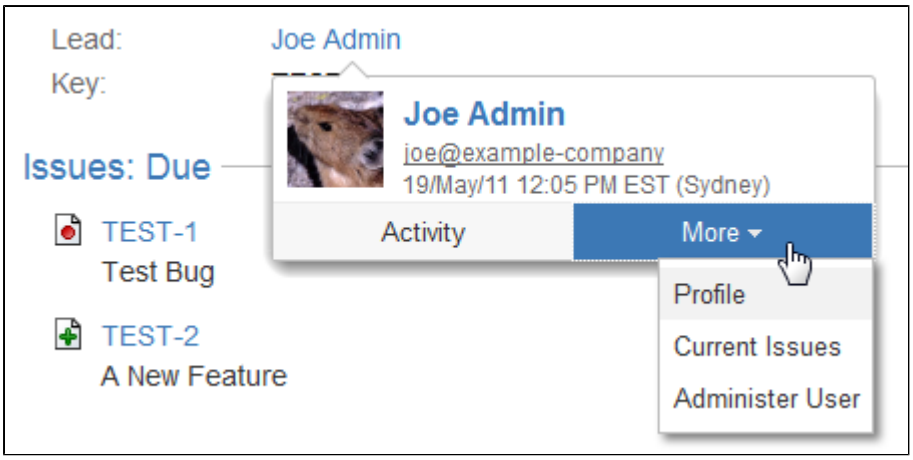
Accessing Hover Profile

When you move or hover your mouse over a user's **username** or **full name** on:

- an issue view,
- any issue listed in the Issue Navigator, or
- any of the project browser screens,

an interactive popup balloon appears.

Screenshot: The Hover Profile popup balloon



Please Note: The Hover Profile popup balloon is not available on user names which appear on activity streams and dashboard gadgets throughout the JIRA interface.

Using the Hover Profile Popup Balloon

The top part of the Hover Profile popup balloon shows the user's full name, avatar, email address and time zone, as defined in their user profile. You can email a user from their Hover Profile by clicking their email address link, which opens up a new email message in your email client with that email address in the **To:** field.

The lower part of the Hover Profile popup balloon also provides easy access to the following information about a user, via the following links:

- **Activity** — the user's recent activity on the JIRA site.
- Click **More**, then:
  - **Profile** — the user's user profile page.
  - **Current Issues** — the user's list of unresolved issues (via the Issue Navigator).
  - **Administer User** *(only visible to **JIRA Administrators** who have the **JIRA Users** permission)* — the user's details in JIRA's user management area of the administration console.

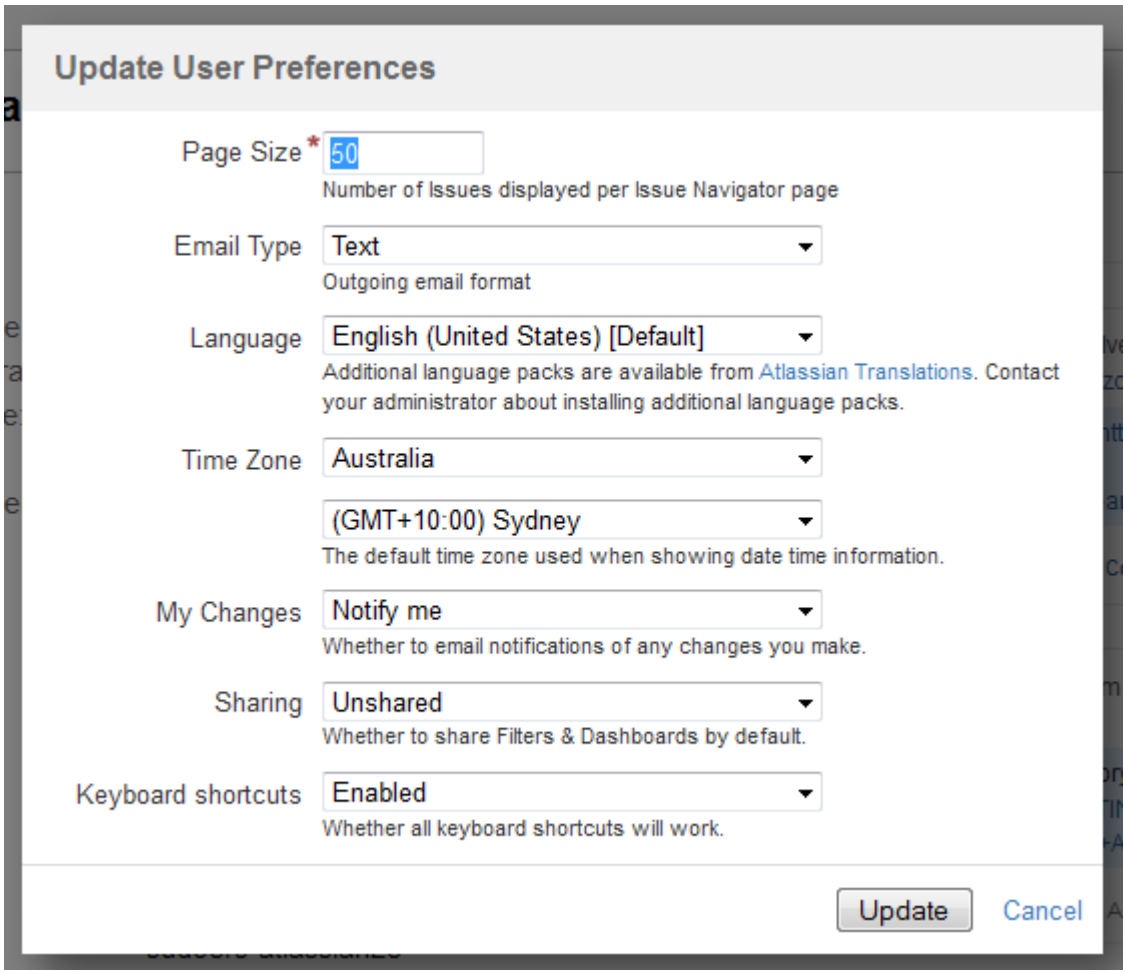
Choosing a Time Zone

The default time zone is set by your JIRA administrator (see Configuring JIRA Options), but you can personalise your JIRA account to use a time zone of your choice. This will affect all time-date fields throughout JIRA.

To choose a time zone:

1. Click your user name at the top-right of the page to open your User Profile at the 'Summary' tab page.
2. In the 'Preferences' section, click the **edit** icon at the top-right. The 'Updated User Preferences' dialog box opens.
3. Select your region (or country) and time zone from the '**Time Zone**' drop-down list:

   Screenshot: Update User Preferences Dialog Box



4. Click the '**Update**' button.
5. All time fields in JIRA will now be displayed in your local time zone. Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information) so are not affected by time zone settings.

JIRA Administrator's Guide

This manual contains information on administering your JIRA system:

Getting Help

Configuring the Layout and Design

- Customising the Look and Feel
- Choosing a Default Language
  - Translating JIRA
- Configuring the Default Issue Navigator
- Configuring the Default Dashboard
  - Using Dashboard Gadgets
  - Adding a Gadget to the Directory
  - Subscribing to Another Application's Gadgets
- Configuring an Announcement Banner
- Enabling Logout Confirmation

User and Group Management

- Managing Users
- Managing Groups
- Managing Project Roles
- Migrating User Groups to Project Roles
- Configuring User Directories
  - Configuring the Internal Directory
  - Connecting to an LDAP Directory
    - Configuring an SSL Connection to Active Directory
  - Connecting to an Internal Directory with LDAP Authentication

        - Connecting to Crowd or Another JIRA Server for User Management
        - Managing Multiple Directories
        - Synchronising Data from External Directories
        - Managing Nested Groups
        - Diagrams of Possible Configurations for User Management
        - User Management Limitations and Recommendations
        - Allowing Other Applications to Connect to JIRA for User Management
    - Viewing User Sessions
        - User access logging
    - Clearing 'Remember my login' Tokens
        - Disabling Remember My Login on this Computer
    - Enabling Public Signup and CAPTCHA

Project Management

- Defining a Project
- Managing Project Role Membership
- Defining a Component
- Managing Versions
    - Running a Bamboo Build when Releasing a Version
    - Creating Release Notes
- Configuring Project Keys

Configuring Security

- Configuring Issue Level Security
- Managing Project Permissions
- Managing Global Permissions
- Configuring Secure Administrator Sessions
- Preventing Security Attacks
- JIRA Cookies

Configuring Fields and Screens

- Configuring Built-in Fields
    - Defining 'Issue Type' Field Values
        - Associating Issue Types with Projects
    - Defining 'Priority' Field Values
    - Defining 'Resolution' Field Values
    - Defining 'Status' Field Values
    - Translating Resolutions, Priorities, Statuses and Issue Types
- Adding a Custom Field
    - Configuring a Custom Field
    - Creating Help for a Custom Field
- Specifying Field Behaviour
    - Associating Field Behaviour with Issue Types
    - Configuring Renderers
- Defining a Screen
    - Associating a Screen with an Issue Operation
    - Associating a Screen with an Issue Type

Configuring Workflow

- Activating Workflow
- Adding a Custom Event
- Configuring the Initial Status
- Workflow Properties

Configuring Email

- Configuring Email Notifications
    - Configuring JIRA's SMTP Mail Server to Send Notifications
    - Creating a Notification Scheme
    - Customising Email Content
- Creating Issues and Comments from Email
    - Configuring JIRA to Receive Email from a POP or IMAP Mail Server
- Using Gmail as a JIRA Mail Server

Migrating from Other Issue Trackers

- Importing Data from Bugzilla
- Importing Data From A FogBugz Server
- Importing Data From FogBugz OnDemand
- Importing Data From Mantis
- Importing Data From Pivotal Tracker
- Importing Data From Trac
- Importing Data From CSV

- Commonly Asked CSV Questions and Known Issues
- How to Import CSV Data with PVCS Command

Moving or Archiving Individual Projects

- Archiving a Project
- Splitting a JIRA instance

Integrating with a Source Control System

- Integrating JIRA with FishEye
- Integrating JIRA with CVS and ViewCVS
- Integrating JIRA with Subversion
- Integrating JIRA with Perforce
- Integrating JIRA with ClearCase

Integrating with a Build Management System

- Integrating JIRA with Bamboo

Configuring Global Settings

- Configuring Time Tracking
- Configuring JIRA Options
- Setting Properties and Options on Startup
  - Recognized System Properties for JIRA
- Advanced JIRA Configuration
  - Changing the constraints on historical time parameters in gadgets
  - Changing the Default Order for Comments from Ascending to Descending
  - Limiting the number of issues returned from a search view such as an RSS feed
- Configuring File Attachments
- Configuring Application Links
  - Adding an Application Link
  - Configuring Authentication for an Application Link
    - Configuring Basic HTTP Authentication for an Application Link
    - Configuring OAuth Authentication for an Application Link
    - Configuring Trusted Applications Authentication for an Application Link
    - Incoming and Outgoing Authentication
  - Editing an Application Link
  - Making an Application Link the Primary Link
  - Relocating an Application Link
  - Upgrading an Application Link
  - Deleting an Application Link
  - Configuring Project Links across Applications
    - Adding Project Links between Applications
    - Making a Project Link the Primary Link
    - Deleting a Project Link
- Configuring Issue Cloning
- Configuring Issue Linking
- Configuring the Whitelist
- Configuring Sub-tasks
- Managing Shared Filters
- Managing Shared Dashboards

Server Administration

- Increasing JIRA Memory
- Using the Database Integrity Checker
- Precompiling JSP pages
- Database Indexing
- Logging and Profiling
  - Logging email protocol details
- Restoring Data
  - Restoring a Project from Backup
- Optimising Performance
- Backing Up Data
  - Automating JIRA Backups
  - Preventing users from accessing JIRA during backups
- Search Indexing
  - Re-Indexing after Major Configuration Changes
- Using robots.txt to hide from Search Engines
- Updating your JIRA License Details
- Viewing your System Information
- Generating a Thread Dump
- Performance Testing Scripts
- Finding the JIRA Support Entitlement Number (SEN)

Appendix A - Extending JIRA

- Managing JIRA's Plugins
- Listeners
- Services
- Jelly Tags
- JIRA Toolkit (Customer Support Extensions)
- Developer Guides
- Building JIRA from Source
    - How to Make a JIRA Patch
- API Documentation

Please see the *JIRA User's Guide* for an introduction to the concepts of issues and projects.

Getting Help

On this page:

- Where to Start
- Raising a Support Request
    - To raise a support request via your JIRA system
    - To raise a support request via the internet
- Creating a Support Zip

Where to Start

If you encounter any problems using or setting up JIRA, please let us know — we're here to help!

You may want to first search the following:

- the Atlassian Answers site (JIRA Forum), where Atlassian staff and JIRA users can answer your questions.
- the JIRA Knowledge Base.

If you need further assistance, please raise a support request (see below).

Alternatively, if you feel you have encountered a bug in JIRA, or wish to request a feature, please file an issue. It is a good idea to first scan JIRA's Popular Issues — this helps to prevent duplicates.

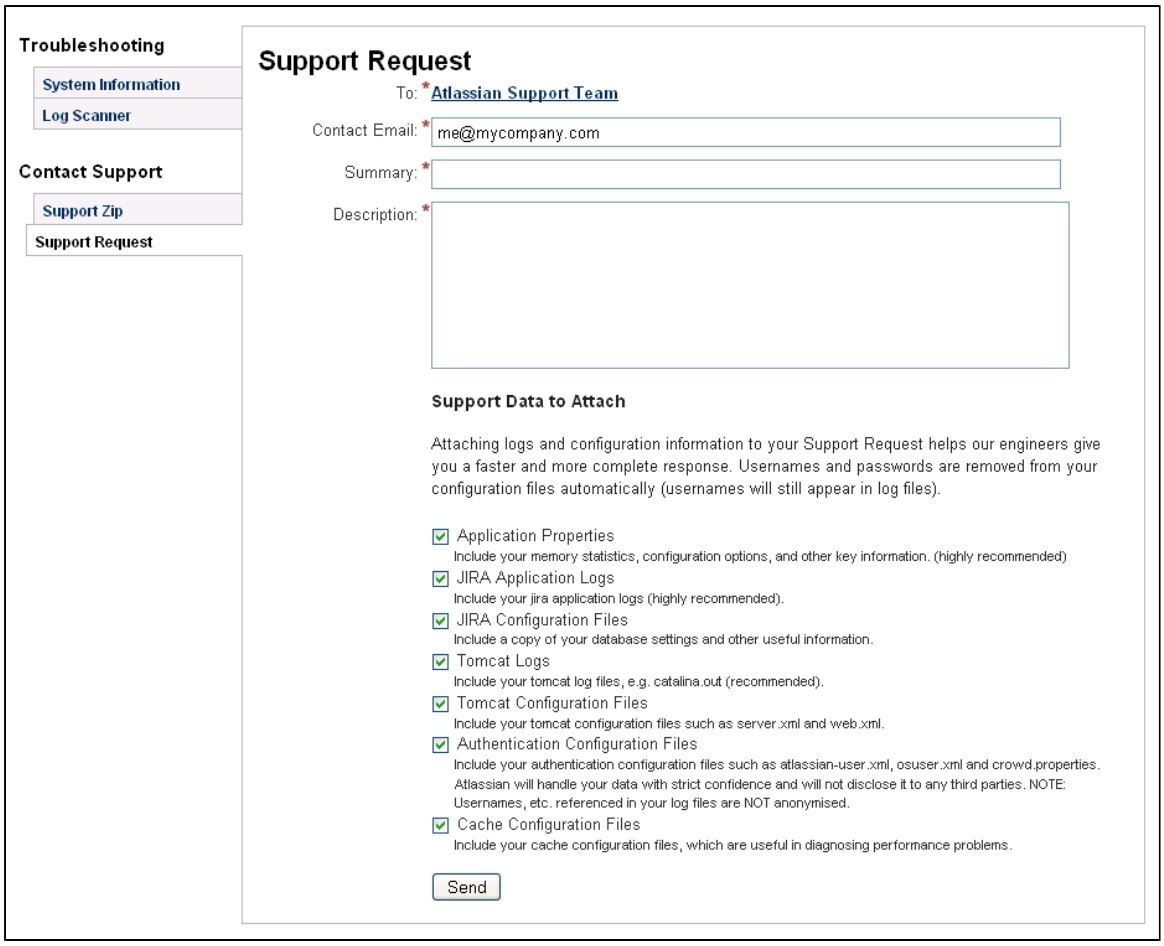
Looking for other helpful information? You can receive news, product information and code tips via our newsletter, blogs and forums. Stay in touch with us here.

Raising a Support Request

You can raise a support request either in JIRA or via the internet, as described below:

To raise a support request via your JIRA system

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Bring up the administration page by clicking either the **Administration** link on the top bar or the title of the Administration box on the dashboard.
3. On the panel on the left, under the title **System**, click the **Atlassian Support Tools** link.
4. Click the **Support Request** tab. The **Support Request**form will be displayed:
    - Please provide as much information as possible, including any error messages that are appearing on the console or in the logs.



5. Once you have submitted your support request, you will receive email updates about its progress. You can also view the status of your support request by visiting the Atlassian Support System

OR:

To raise a support request via the internet

1. Please visit the Atlassian Support System and create a support request.
2. Please provide as much information as possible, including any error messages that are appearing on the console or in the logs. Please also mention the operating system, database and version of JIRA you are using.

> Sometimes it is necessary to adjust JIRA's logging levels to get a more detailed error message or a stack trace. Please see the logging section of the documentation for information on how to do this.

Creating a Support Zip

If you have created a support request via the internet, you may want to create a 'Support Zip' (which contains information about your JIRA system) and attach it to the support request. This will assist our support engineers in troubleshooting the issue.

To create a Support Zip:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **Administration > System > Atlassian Support Tools** to open the **Atlassian Support Tools** page.
   Keyboard shortcut: **g + g +** start typing **support**
3. Click the **Support Zip** tab. The **Support Zip** page will be displayed.
4. Leaving all the boxes ticked, click the **Create** button at the bottom of the screen.

You can now go to your support request and attach the Support Zip.

Configuring the Layout and Design

The following pages contain information on configuring the layout and design of JIRA:

- Customising the Look and Feel
- Choosing a Default Language
- Configuring the Default Issue Navigator

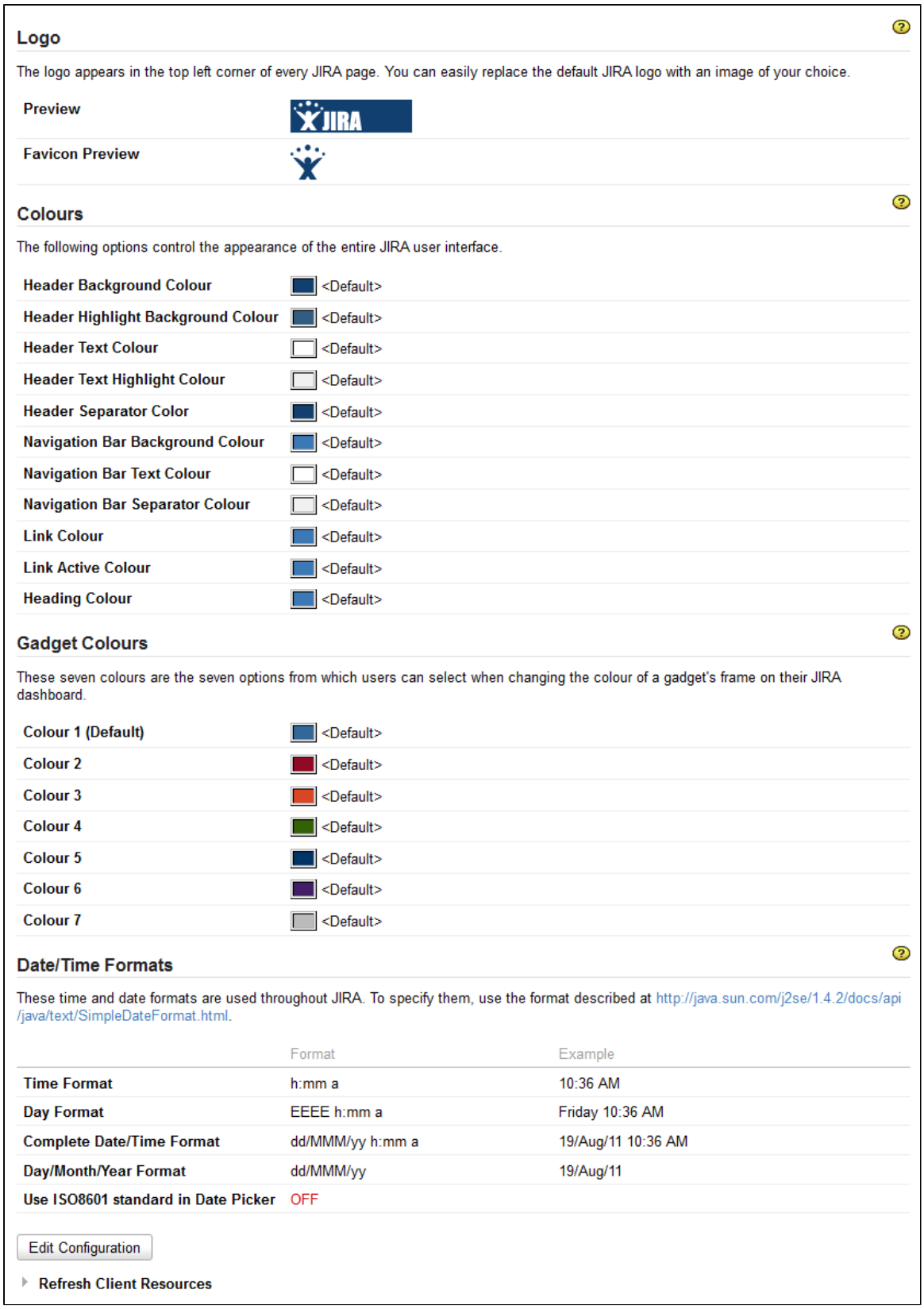
- Configuring the Default Dashboard
- Configuring an Announcement Banner
- Enabling Logout Confirmation

You may also wish to extend JIRA's functionality by installing and/or enabling new plugins. Read the Managing JIRA's Plugins documentation for further information.

Customising the Look and Feel

You can easily customise JIRA's look and feel to suit your needs:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'User Interface' > 'Look and Feel'**.
   Keyboard shortcut: **'g' + 'g' +** start typing **'look and feel'**
3. The **'Look and Feel Configuration'** page will be displayed as follows:
   Screenshot: Look and Feel Configuration



4. To edit the logo, colours or time format, click the '**Edit Configuration**' button at the bottom of the page, which opens the page in edit mode. In edit mode:
    - To access the remaining options on the page, click '**Customize Colours And Dates**' to expand this section of options.
    - To save your configuration changes, click the '**Update**' button.
    - To leave edit mode and return to the 'Look and Feel Configuration' page, click the '**Cancel**' button.
    - To reset JIRA back to its default look and feel, click the '**Reset Defaults**' button.
      Be aware that this is an immediate action and does not require the 'Update' button to be clicked.

Here is a list of the different configuration options available, and what they do.

- Logo
- Colours
- Gadget Colours

- Date/Time Formats
  - Configuring Date Picker Formats

On this page:

- Logo
- Colours
- Gadget Colours
- Date/Time Formats
  - Configuring Date Picker Formats

Logo

The logo appears in the top left corner of every JIRA page while the favicon appears typically to the left of your your browser's URL field and on browser tabs displaying a page on your JIRA site. You can easily replace the default JIRA logo and/or favicon with an image of your choice.

Option	Explanation
Preview (Logo)	A preview of your JIRA site's current logo is shown here.
Favicon Preview (Favicon)	A preview of your JIRA site's current favicon is shown here.

In edit mode:

- Click '**Upload**' to upload a new image for the logo/favicon.
- Click '**URL**' to point to the location of an existing image.
  - A URL beginning with '`http://`' or '`https://`' is treated by JIRA as an absolute URL/path.
  - A URL beginning with a forward slash '`/`' is treated as a path relative to the `<jira-application-dir>` subdirectory of your JIRA Installation Directory.
- Click '**Default**' to revert to using JIRA's default logo/favicon.

✅ **Tip:** If you use a JIRA WAR distribution, it is recommended that you add your logo images to the `edit-webapp` subdirectory of your JIRA Installation Directory prior to building your WAR distribution file. For details on building JIRA WAR distributions, refer to the application server-specific documentation in the Installing JIRA WAR section.

⚠️ **Please Note:** If the JIRA logo/favicon does not appear after changing it to a custom one, ensure that the URL specified uses the correct case as this may be case-sensitive. Try copying and pasting the URL into a browser and check that the image appears.

Colours

The following options control the appearance of the entire JIRA user interface.

Option	Explanation
Header Background Colour	The background colour of the top bar (the one that includes the image).
Header Highlight Background Colour	The background colour of the text that sits inside the top bar, when selected or when the mouse hovers over it.
Header Text Colour	The colour of the text that sits inside the top bar (such as your user name when you are logged in).
Header Text Highlight Colour	The colour of the text that sits inside the top bar, when selected or when the mouse hovers over it.
Header Separator Colour	The colour of the horizontal line between the top bar and the navigation bar.
Navigation Bar Background Colour	The background colour of the bar that contains the links to 'Dashboards', 'Projects', etc.
Navigation Bar Text Colour	The text color of the links in the menu bar (e.g. 'Dashboards').

Navigation Bar Separator Colour	The colour of the vertical dotted line between each menu item and its drop-down symbol (triangle).
Link Colour	The colour of the text links on any JIRA page.
Link Active Colour	The colour of the text links on any JIRA page, when selected.
Heading Colour	The colour of the text headings on any JIRA page.

Please note:

- The colours you specify for each of the following options can be anything that is valid for both a font tag, and a stylesheet's 'color:' attribute.
- When specifying a colour, you can use the pop-up colour chooser, or specify your own (eg. *'#FFFFFF', 'red'*).
- To return to the original colour scheme, just clear any values that you have set.

Gadget Colours

These seven colours are the seven options from which users can select when changing the colour of a gadget's frame on their JIRA dashboard. Colour 1 is the default frame colour for newly-added gadgets.

Please note:

- The colours you specify for each of the eight options can be anything that is valid for both a font tag, and a stylesheet's 'color:' attribute.
- When specifying a colour, you can use the pop-up colour chooser, or specify your own (eg. '#FFFFFF', 'red').
- To return to the original colour scheme, just clear any values that you have set.

Date/Time Formats

The 'Look and Feel' page allows you to customise the time and date formats used throughout JIRA.

When specifying dates and times, they should be based on the Java SimpleDateFormat.

When you are not in edit mode on the 'Look and Feel' page, the examples in the rightmost column of the 'Date/Time Formats' section show you how the various formats will appear in JIRA.

Configuring Date Picker Formats

To set the format of date pickers in JIRA, see Configuring Advanced Settings.

The date or date/time formats are defined by two (Java and JavaScript) properties. These two properties must match for the date (or date/time) picker to work correctly.

For Java formats, specify date/time formats based on the Java SimpleDateFormat.

For JavaScript formats, specify date/time formats based on the Unix date format.

Here are some example US-based date configurations:

Preferred Date	**Value of the `jira.date.picker.java.format` property**	**Value of the `jira.date.picker.javascript.format` property**
Oct/1/10	MMM/d/yy	%b/%e/%y
10/01/10	MM/dd/yy	%m/%d/%y
Oct 1, 2010	MMM d, yyyy	%b %d, %Y
10/01/2010	MM/dd/yyyy	%m/%d/%Y

Here are some examples of date/time configurations:

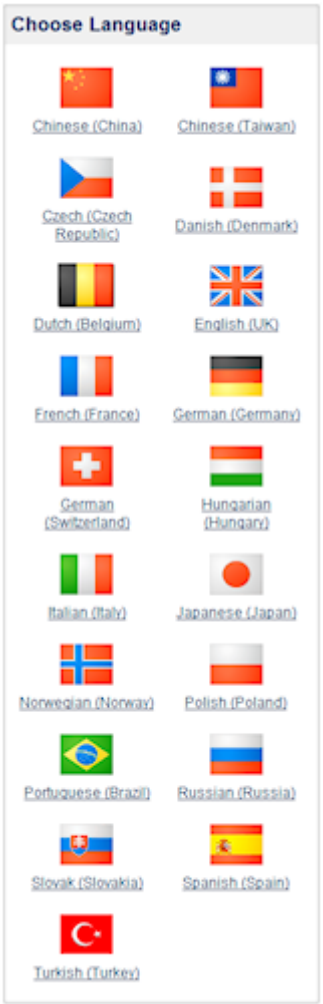
Preferred Date/Time	Value of the `jira.date.time.picker.java.format` property	Value of the `jira.date.time.picker.javascript.format` property
15/Oct/10 8:50 AM	dd/MMM/yy h:mm a	%e/%b/%y %I:%M %p
10/15/10 08:50 AM	MM/dd/yy hh:mm a	%b/%e/%y %I:%M %p

Choosing a Default Language

Overview

Most user-visible pages in JIRA are now internationalised. Chinese, Czech, Danish, English, French, German, Italian, Norweigian, Polish, Portuguese (Brazilian), Russian, Japanese, Slovak and Spanish translations are available (at time of writing), with more in development.

When JIRA is first installed, the default language may be chosen by clicking on a flag:

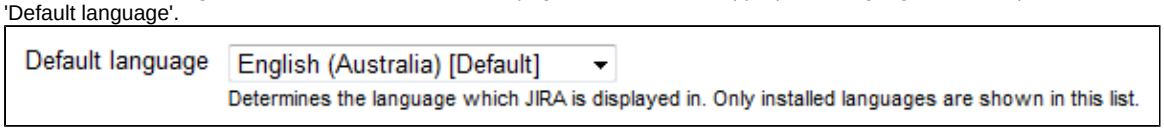


On this page:

- Overview
- Changing the default language
- Per-user language selection
- Overriding the default translations of Issue Types, Resolutions, Statuses and Priorities
- Related Topics

Changing the default language

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'General Configuration'** to open the 'General Configuration' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'general configuration'**
3. Click the '**Edit Configuration**' button at the end of the page, then select the appropriate language in the dropdown box next to 'Default language'.



   Any additional languages you have installed will appear in the list. See Translating JIRA.

Per-user language selection

Individual users can choose their own language, which will override the default language (see above).

Overriding the default translations of Issue Types, Resolutions, Statuses and Priorities

Should you wish, you can easily specify your own translations for the values of the following JIRA issue fields:

- Issue Type
- Priority
- Status
- Resolution

Your specified translations will override the values specified in the JIRA translation.

Related Topics

- Translating JIRA

Translating JIRA

This page contains information about translating JIRA into languages other than English.

On this page:

- Atlassian Translations – a collaborative environment for creating translations of JIRA
- What translations of JIRA are currently available?
- What about translations of the documentation?

Atlassian Translations – a collaborative environment for creating translations of JIRA

The Atlassian Translations site provides a collaborative environment for customers to translate JIRA. (Refer to the instructions for more information). At present there are thousands of accepted translations across a number of languages. We need your help to make this even better! If you are looking at updating or creating a language pack please use Atlassian Translations and tell us about your experience. You can log in with your My Atlassian account. To provide feedback or submit an existing language pack for import please contact The Internationalisation Team.

What translations of JIRA are currently available?

Currently, JIRA ships with a number of translations in the most commonly-requested languages. You can easily update these via the Universal Plugin Manager — please see Managing JIRA's Plugins.

As a JIRA administrator, you can choose the default language from the list of installed languages: see Choosing a Default Language for the latest list.

Individual users can also choose their preferred language from the same list: see Choosing a Language.

What about translations of the documentation?

We do not currently offer translations of the JIRA documentation into other languages. However, we do offer a page where people can contribute the guides they have written in languages other than English: JIRA Documentation in Other Languages.

Configuring the Default Issue Navigator

The Issue Navigator is used within JIRA to find and filter issues, and to display the search results in various formats ('views'). It is possible to select which issue fields will be displayed as columns in the Issue Navigator.

JIRA **administrators** can configure which columns appear in the Issue Navigator by default, for all users that do not have their personal navigator columns configured. Each authenticated JIRA **user** can override these defaults by configuring their own Issue Navigator columns to fit their needs. Note that only users who can see at least one issue in the JIRA system are able to configure Issue Navigator columns.

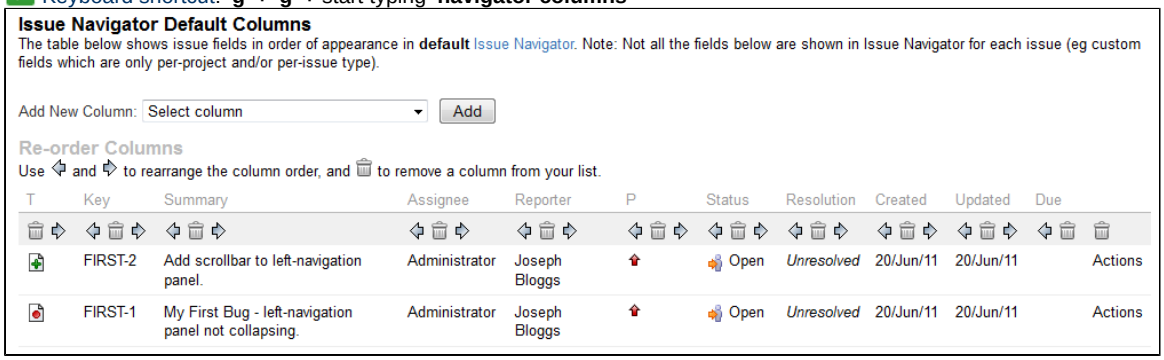
JIRA administrators can also select which views are available in the JIRA system, as views are configurable via plugins.

On this page:

- Configuring the Default Issue Navigator Columns
- Related Topics

Configuring the Default Issue Navigator Columns

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'User Interface' > 'Navigator Columns'** to display the 'Issue Navigator Default Columns' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'navigator columns'**



3. On the 'Issue Navigator Default Columns' page, you can do the following:
   - **To move a column left or right,** click on the left-arrow or right-arrow icon that appears under the column's heading.
   - **To remove a column from the list,** click the bin icon which appears under the column's heading.
   - **To add a column to the list,** select the issue field name from the drop-down box titled **'Add New Column'** and click the **'Add'** button. The column will appear as the right-most column in the list. You can then position the column where desired by using the arrow icons.
   - **To hide the 'Actions' column,** click the **'Hide Column'** link.
   - If the column order has been modified from the defaults, users can restore the global defaults by clicking the **'Restore Defaults'** link (which will appear only if they have modified their Issue Navigator from the global defaults). When configuring the global defaults (only available to administrators), the link is called **'Restore System Defaults',** and when clicked restores the configuration that JIRA ships with by default.

Note:

- When configuring their personal Issue Navigator columns, a user can only see columns for issue fields that have not been hidden.
- It is possible to add any of the existing custom fields to the Issue Navigator column list. When configuring the columns a user can choose any custom field that they have permissions to see. That is, any custom field except those that are project-specific and apply only to a project that the user does not have permissions to browse. Some custom fields, even if selected as Issue Navigator columns, will not appear in the Issue Navigator for all issues. For example, project-specific custom fields will be shown only if the filter has been restricted to that project only. Issue type custom fields will only appear if the filter has been restricted to that issue type.
- When administrators are configuring default Issue Navigator columns, their permissions are ignored, so that they can add a project-specific custom field from a project that they do not have permissions to browse. The field would never be actually shown to users that do not have permissions to see it.

Related Topics

- Customising your Issue Navigator

Configuring the Default Dashboard

The default dashboard is the screen that all JIRA users see the first time they login. Any users who have not added any dashboard pages as favourites also see the default dashboard.

JIRA allows Administrators to configure the default dashboard. The gadgets on the default dashboard can be re-ordered, switched between the left and right columns, additional gadgets can be added, and some gadgets can be configured. The layout of the dashboard (e.g. number of columns) can also be configured.

All changes made to the default dashboard will also change the dashboards of all users currently using the default. However, gadgets that users do not have permissions to see will not be displayed to them. For example, the 'Administration' gadget, although it may exist in the default dashboard configuration, will not be visible to non-admin users.

Gadgets are the information boxes on the Dashboard. JIRA comes pre-configured with a set of standard dashboard gadgets. It is also possible to develop custom gadgets and plug them into JIRA using its flexible plugin system.

On this page:

- Adding and Configuring Gadgets on the Default Dashboard
- See Also

Adding and Configuring Gadgets on the Default Dashboard

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'User Interface' > 'System Dashboard'** to display the 'Configure System Dashboard' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'system dashboard'**
3. On the 'Configure System Dashboard' page, you can do the following:
   - To move the current gadgets to a different position on the default dashboard, simply drag-and-drop them.
   - To re-configure the existing gadgets, please see Changing the Look and Behaviour of a Gadget.
   - To choose a different layout for the default dashboard, please see Customising the Dashboard.

Please note:

- JIRA's default dashboard is limited to only one dashboard page. However, users can add multiple pages to their own dashboards if they wish.
- By default, there is a limit of 20 gadgets per dashboard page. If you wish to raise this limit, edit the jira-config.properties file, set `jira.dashboard.max.gadgets` to your preferred value and then restart JIRA.

See Also

- Using Dashboard Gadgets
- Adding a Gadget to the Directory
- Subscribing to Another Application's Gadgets

- Customising the Dashboard

Using Dashboard Gadgets

On this page:

- About gadgets
- Preinstalled gadgets
- Extension gadgets
- Creating new gadgets

About gadgets

JIRA provides the ability to display summary information about project/issue data on the dashboard, through the use of 'gadgets'. Each gadget can be configured to display project and issue details relevant to particular users. Gadgets can be added to the dashboard — providing a central location for quick access to this information.

Adding Atlassian gadgets to external websites

You can also add Atlassian gadgets to compatible external websites, like iGoogle. For instructions on how to do this, please refer to Adding an Atlassian Gadget to iGoogle and Other Web Sites.

Preinstalled gadgets

JIRA provides a set of standard gadgets out-of-the-box:

Gadget	Description
Activity Stream Gadget	The **Activity Stream** gadget displays a summary of your recent activity.
Administration Gadget	The **Administration** gadget displays quick links to administrative functions.

Assigned To Me Gadget	The **Assigned To Me** gadget displays all open issues in all projects assigned to the current user viewing the dashboard.
Average Age Gadget	The **Average Age** gadget displays a bar chart showing the average number of days that issues have been unresolved.
Bamboo Charts Gadget *	The **Bamboo Charts** gadget displays various charts and plan statistics from a particular Bamboo server.
Bamboo Plan Summary Chart Gadget *	The **Bamboo Plan Summary** gadget displays a graphical summary of a build plan.
Bamboo Plans Gadget *	The **Bamboo Plans** gadget displays a list of all plans on a Bamboo server, and each plan's current status.
Bugzilla ID Search Gadget	The **Bugzilla ID Search** gadget allows the user to search all JIRA issues for references to Bugzilla IDs.
Calendar Gadget *	The **Issue Calendar** gadget shows issues and versions in a calendar format based on their due date. Calendars can be based on an issue filter or on a project.
Clover Coverage Gadget *	The **Clover Coverage** gadget displays the Clover coverage of plans from a particular Bamboo server.
Created vs Resolved Gadget	The **Created vs Resolved** gadget displays a difference chart showing the issues created vs resolved over a given period.
Crucible Charts Gadget *	The **Crucible Charts** gadget displays various charts showing statistical summaries of code reviews.
Favourite Filters Gadget	The **Favourite Filters** gadget displays a list of all the issue filters that have currently been added by you as a favourite filter.
Filter Results Gadget	The **Filter Results** gadget displays the results of a specified issue filter.
FishEye Charts Gadget *	The **FishEye Charts** gadget displays two charts showing showing statistics about a given sourcecode repository.
FishEye Recent Changesets Gadget *	The **FishEye Recent Changesets** gadget displays a number of recent changesets from a FishEye repository.
In Progress Gadget	The **In Progress** gadget displays all issues that are currently in progress and assigned to the current user viewing the dashboard.
Introduction Gadget	The **Introduction** gadget displays a configurable introduction message on the dashboard.
Issue Statistics Gadget	The **Issue Statistics** gadget displays the collection of issues returned from a specified filter, broken down by a specified field.
JIRA: News Gadget	The **JIRA:News** gadget displays recent Atlassian news about JIRA.
Pie Chart Gadget	The **Pie Chart** gadget displays issues from a project or issue filter, grouped by a statistic type, in pie-chart format. The issues can be grouped by any statistic type (e.g. Status, Priority, Assignee, etc).

Projects Gadget	The **Projects** gadget provides information and various filters related to a specified project(s).
Quick Links Gadget	The **Quick Links** gadget displays a number of useful links to issues associated with the current user.
Recently Created Issues Gadget	The **Recently Created Issues** gadget displays a bar chart showing the rate at which issues are being created, as well as how many of those created issues are resolved.
Resolution Time Gadget	The **Resolution Time** gadget displays a bar chart showing the average resolution time (in days) of resolved issues.
Road Map Gadget	The **Road Map** gadget shows versions which are due for release within a specified period of time, and a summary of progress made towards completing the issues in those versions.
Text Gadget *	The **Text** gadget displays a configurable HTML text on the dashboard.
Time Since Issues Gadget	The **Time Since Issues** gadget displays a bar chart showing the number of issues that something has happened to within a given time period. The 'something has happened' is based on a date field that you choose, such as 'Created', 'Updated', 'Due', 'Resolved' or a custom field.
Two Dimensional Filter Statistics Gadget	The **Two Dimensional Filter Statistics** gadget displays statistical data based on a specified filter in a configurable table format.
Voted Gadget	The **Voted Issues** gadget shows issues for which you have voted.
Watched Gadget	The **Watched Issues** gadget shows issues which you are watching.

See the big list of all Atlassian gadgets for more ideas.

*This gadget will only be available if you have installed/configured the relevant plugin.

Extension gadgets

Other gadgets are available as plugins on the JIRA Extensions site. These plugins include:

- Calendar plugin
- Timesheet plugin

Should you wish to you use these plugins, you need to first install them (using the instructions provided with each plugin) then enable them.

Creating new gadgets

New gadgets can be created by writing an XML descriptor file, packaged as an Atlassian plugin. See Writing an Atlassian Gadget for more information.

RELATED TOPICS

The big list of Atlassian gadgets

Adding a Gadget to the Directory

The JIRA gadget directory displays all the gadgets that are available for JIRA users to add to their dashboard.

You need to have administrator privileges to add a gadget to the directory. If you have permission to add gadgets to and remove gadgets from the directory itself, you will see the '**Add Gadget to Directory**' and '**Remove**' buttons on the 'Add Gadget' screen, as shown below.

On this page:

- Adding a Gadget that is Not a Plugin
- Adding a Gadget that must be Installed as a Plugin

*The information on this page does not apply to **JIRA OnDemand**.*

Security implications
Add only gadgets from sources that you trust. Gadgets can allow unwanted or malicious code onto your web page and into your application. A gadget specification is just a URL. The functionality it provides can change at any time.

There are two types of gadgets: those that must be installed as plugins, and those that can be added as simple gadget URLs.

Adding a Gadget that is Not a Plugin

If the gadget is hosted on another server and can be added to the directory as a simple URL, then you can simply add it via your dashboard's 'Add Gadget' option.

To add a gadget to your directory,

1. First you need to find the URL for the gadget's XML specification file. Gadget authors and publishers make their gadget URLs available in different ways. Below are the instructions for an Atlassian gadget and a Google gadget.
    - Follow the steps below if you need to find the URL for a gadget that is published by an Atlassian application, such as JIRA or Confluence: A gadget's URL points to the gadget's XML specification file. Gadget URLs are shown on the '**Gadget Directory**' screen that is displayed when you click '**Add Gadget**'. In general, a gadget's URL looks something like this:

      ```
      http://example.com/my-gadget-location/my-gadget.xml
      ```

      If the gadget is supplied by a plugin, the URL will have this format:
      `http://my-app.my-server.com:port/rest/gadgets/1.0/g/my-plugin.key:my-gadget/my-path.`

      For example:
      `http://mycompany.com/jira/rest/gadgets/1.0/g/com.atlassian.streams.streams-jira-plu`

      To find a gadget's URL in JIRA:

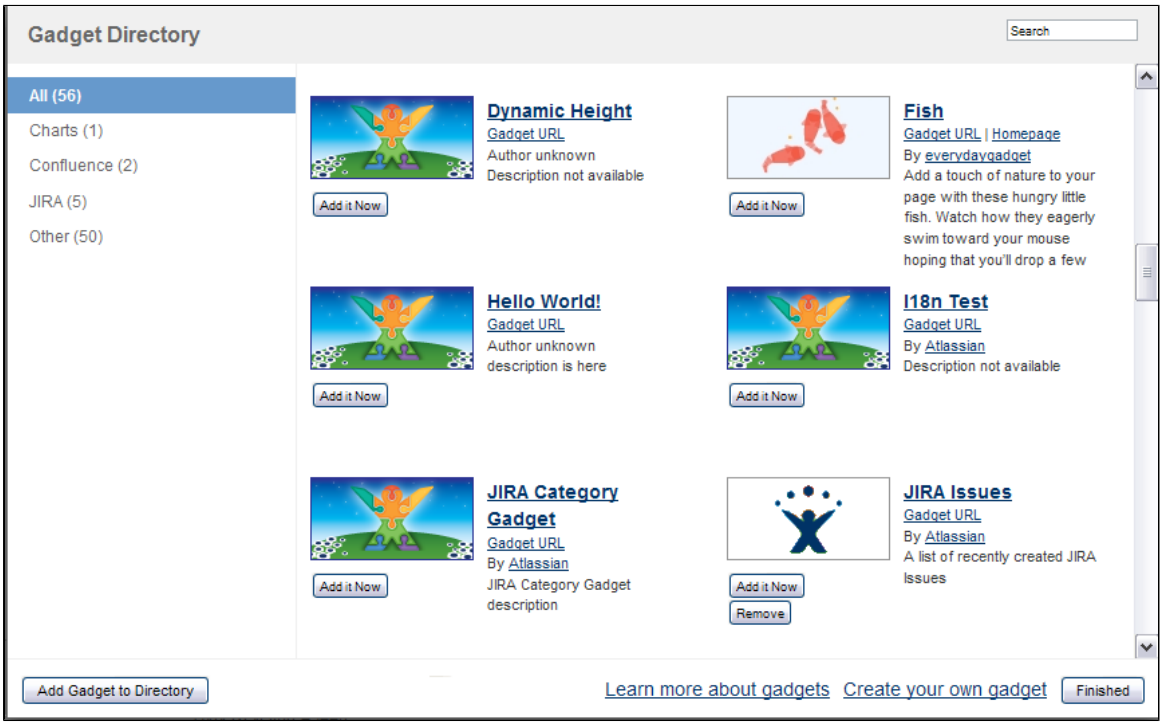
        - Go to your dashboard by clicking the '**Dashboards**' link at the top left of the screen.
        - Click '**Add Gadget**' to see the list of gadgets in the directory.
        - Find the gadget you want, using one or more of the following tools:
            - Use the scroll bar on the right to move up and down the list of gadgets.
            - Select a category in the left-hand panel to display only gadgets in that category.
            - Start typing a key word for your gadget in the '**Search**' textbox. The list of gadgets will change as you type, showing only gadgets that match your search term.
        - Right-click the '**Gadget URL**' link for that gadget and copy the gadget's URL into your clipboard.

      To find a gadget's URL in Confluence:

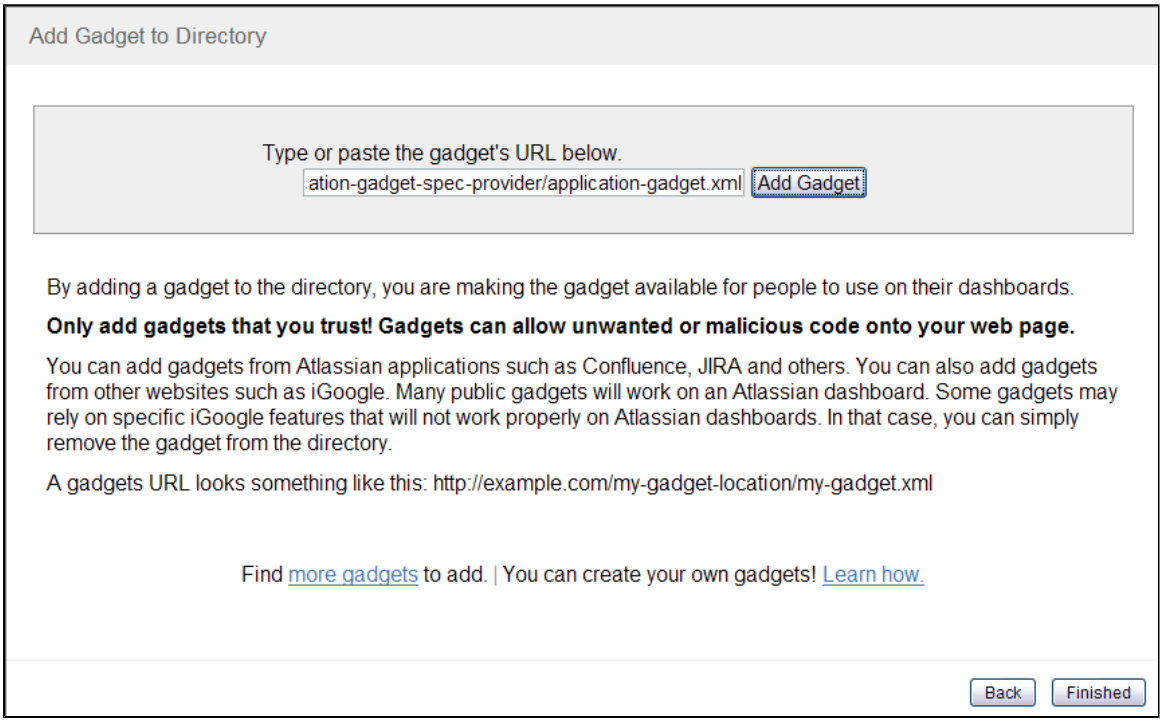
        - Open the '**Browse**' menu and click '**Confluence Gadgets**' to see the list of available Confluence gadgets.
        - Find the gadget you want.
        - Right-click the '**Gadget URL**' link for that gadget and copy the gadget's URL into your clipboard.

    - Follow the steps below if you need to find the URL for a Google gadget:
        a. Go to the Google gadget directory. (You can also get there by clicking 'Add Stuff' from your iGoogle home page.)
        b. Search for the gadget you want.
        c. Click the link on the gadget to open its home page.
        d. Find the '**View source**' link near the bottom right of the page. Right-click the link and copy its location to your clipboard. This is the gadget's URL.
2. Now you can add the gadget to your directory. Go to the dashboard by clicking the '**Dashboard**' link or the '**Home**' link at the top left of the screen.
3. The dashboard will appear. Click '**Add Gadget**'.
4. The '**Add Gadget**' screen appears, showing the list of gadgets in your directory. See screenshot 1 below. Click '**Add Gadget to Directory**'.
   You will only see this button if you have administrator permissions for your dashboard.
5. The '**Add Gadget to Directory**' screen appears. See screenshot 2 below. Type or paste the gadget URL into the text box.
6. Click '**Add Gadget**'.
7. The gadget appears in your gadget directory. (It will be highlighted for a short time, so that you can see it easily.)

Screenshot 1: Gadget directory with 'Add Gadget to Directory' button



Screenshot 2: Adding a gadget to the directory



Adding a Gadget that must be Installed as a Plugin

If the gadget must be installed as a plugin, you cannot add it via the gadget directory user interface.

Instead, you will need to follow the instructions for adding a plugin, as described in Managing JIRA's Plugins.

Once you have installed your plugin, the gadget will automatically appear in the directory.

RELATED TOPICS

The big list of Atlassian gadgets

Subscribing to Another Application's Gadgets

Security Implications

Add only gadgets from sources that you trust. Gadgets can allow unwanted or malicious code onto your web page and into your application. A gadget specification is just a URL. The functionality it provides can change at any time.

*The information on this page does not apply to **JIRA OnDemand**.*

If you have administrator privileges, you can configure your application to subscribe to gadgets from other Atlassian applications. This feature allows administrators to make all the gadgets from one application available in another application, without having to enable each gadget individually via the gadget URL.

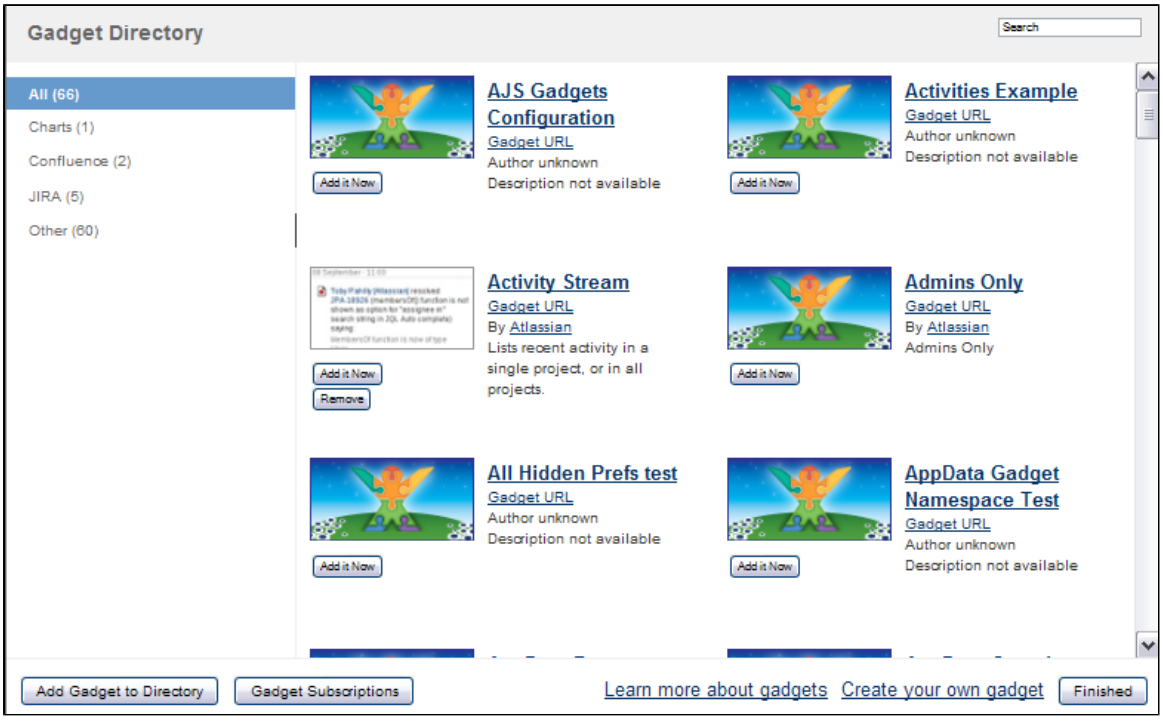
To make use of this feature, you will need two or more applications that support the feature.

The gadgets included are those provided by the other application or via plugins installed into that application. They do *not* include external gadgets that the other application has added to its directory.

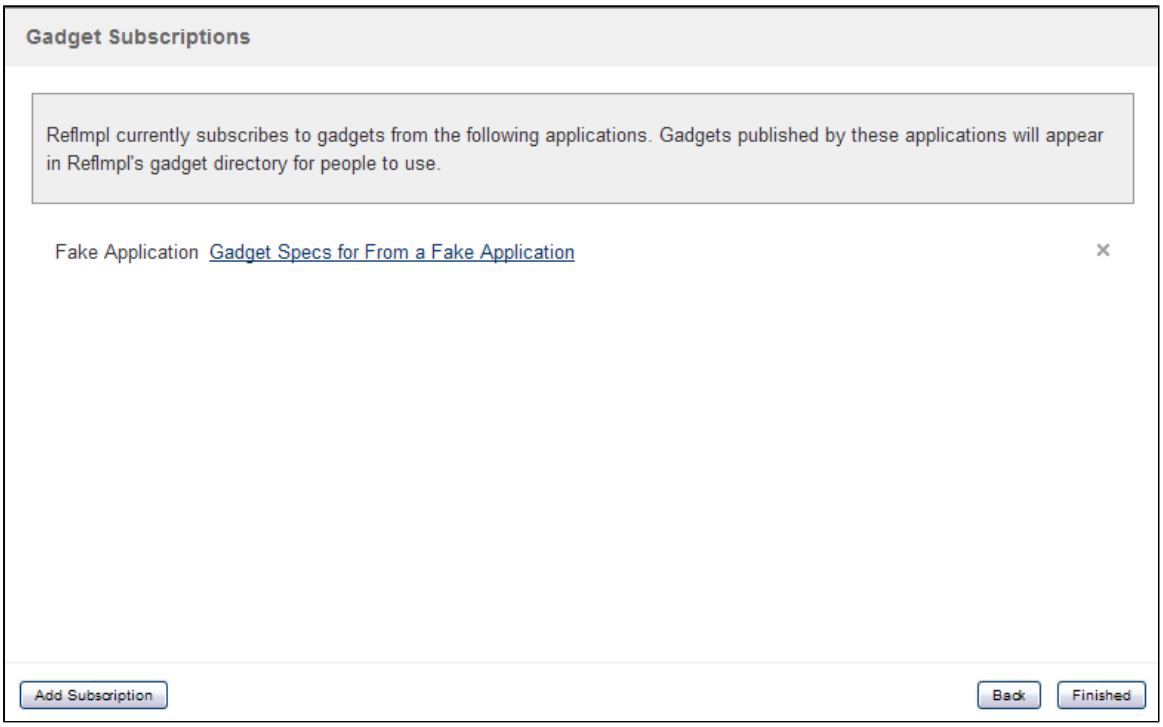
To subscribe to gadgets from another application,

1. Go to the dashboard by clicking the '**Dashboard**' link or the '**Home**' link at the top left of the screen.
2. The dashboard appears. Click '**Add Gadget**'.
3. The '**Add Gadget**' screen appears, showing the list of gadgets in your directory. See the gadget directory screenshot below. Click '**Gadget Subscriptions**'.
   You will only see this button if you have administrator permissions for your dashboard, and if your application supports gadget subscriptions.
4. The '**Gadget Subscriptions**' screen appears, showing the applications to which your application already subscribes. See the subscriptions screenshot below. Click '**Add Subscription**'.
5. The '**Add Subscription**' screen appears. See the screenshot below. Enter the base URL of the application you want to subscribe to. For example, `http://example.com/jira` or `http://example.com/confluence`.
6. Click '**Finished**' to add the subscription.

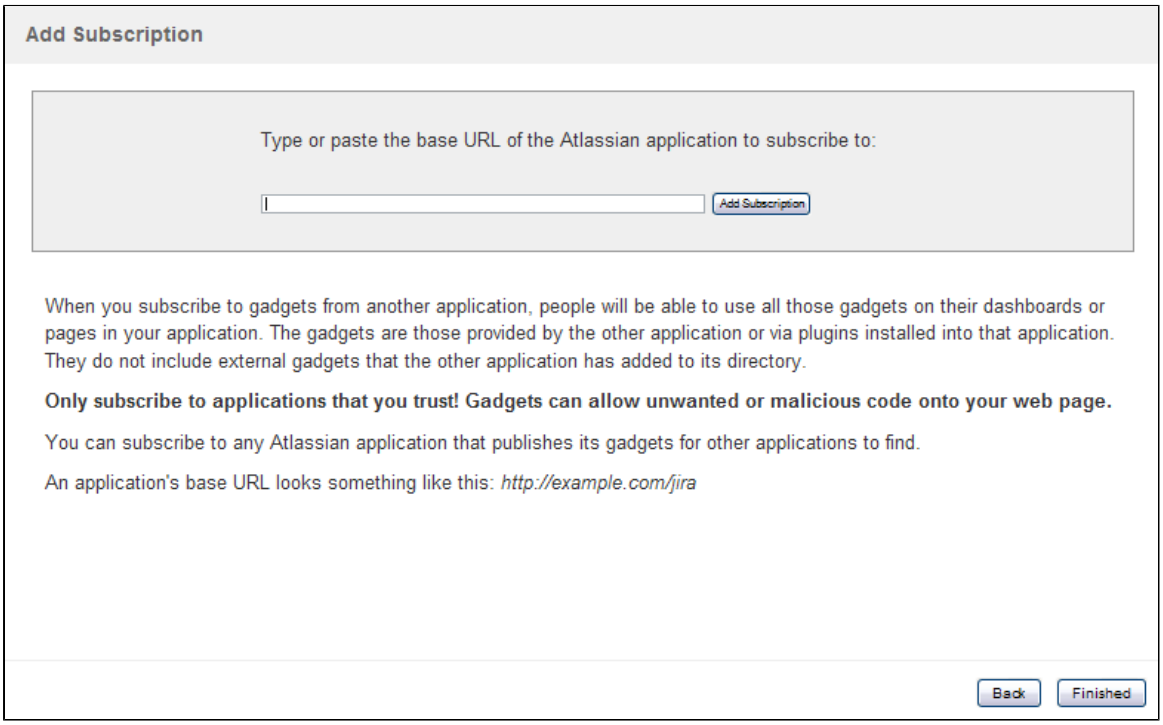
Screenshot 1: Gadget directory with 'Gadget Subscriptions' button



Screenshot 2: List of existing gadget subscriptions



Screenshot 3: Adding a gadget subscription



RELATED TOPICS

The big list of Atlassian gadgets

Configuring an Announcement Banner

Administrators can configure an **announcement banner** to display pertinent information on all JIRA pages. The banner can be used to relate important information (e.g. scheduled server maintenance, approaching project deadlines, etc.) to all users. Further, the banner visibility leve can be configured to display to all users or just logged-in users.

The banner can be configured to contain HTML text.

On this page:

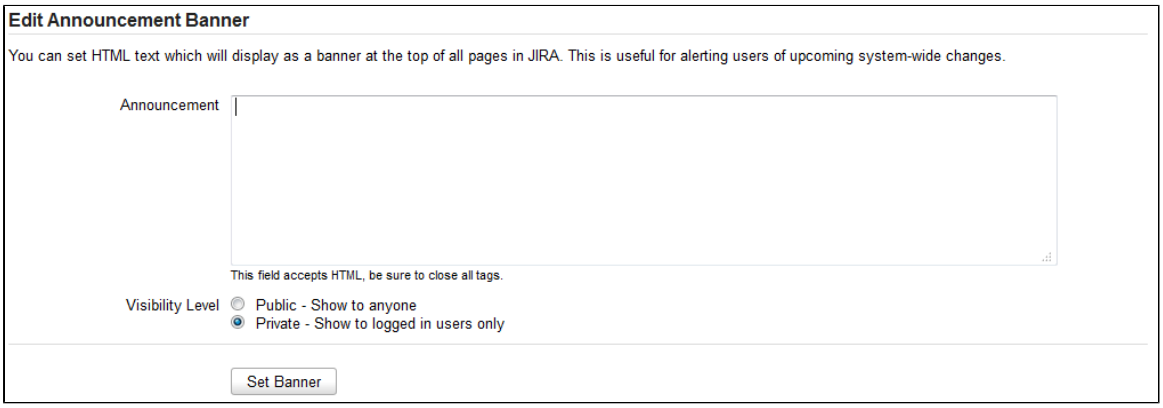
- Configuring an Announcement Banner
- Banner Visibility Mode

Configuring an Announcement Banner

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'User Interface' > 'Announcement Banner'** (tab) to open the 'Edit Announcement Banner' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'announcement banner'**
3. Enter the required text in the 'Announcement' field.
4. Select the required 'Visibility Level' for the banner.
5. Click the **Set Banner** button.

Depending on the visibility level selected, the banner will become visible throughout JIRA.

Screenshot: Configuring the Announcement Banner



Banner Visibility Mode

The announcement banner visibility level can be configured to specify to whom the banner will be displayed. There are two modes:

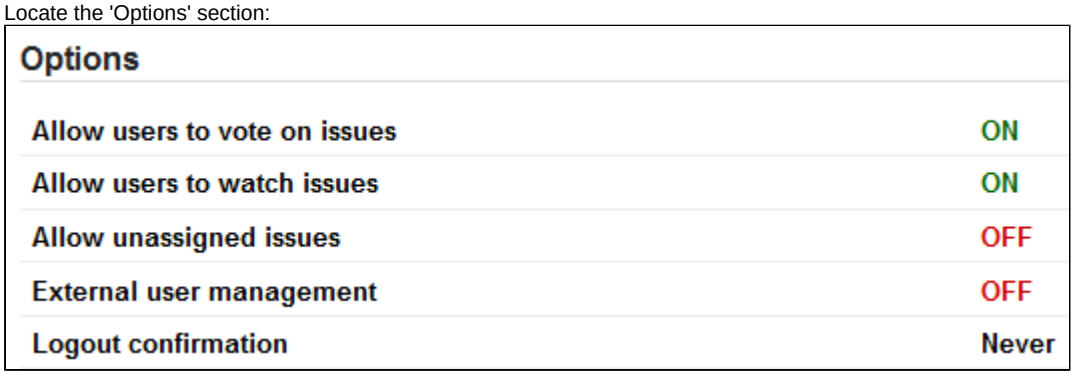
- **Public** — the banner is visible to everyone
- **Private** — the banner is visible to logged-in users **only**

Enabling Logout Confirmation

Administrators can configure JIRA to prompt users with a confirmation before logging them out.

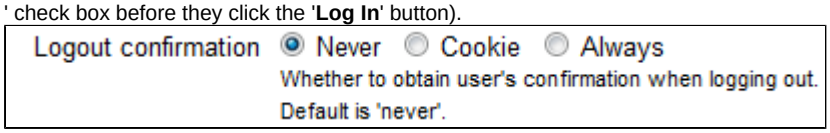
By default, JIRA will not prompt users to confirm logging out. To change this:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'General Configuration'** to open the 'General Configuration' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'general configuration'**
3. Locate the 'Options' section:



4. By default, JIRA will not prompt users to confirm logging out by default. To change this, click the '**Edit Configuration**' button at the end of the page.
   The 'Logout Confirmation' field will now be editable. The **Never** and **Always** settings are self-explanatory. When set to **Cookie**, your

JIRA users will only be prompted if they have logged in using a cookie (i.e. by selecting the '**Remember my login on this computer** ' check box before they click the '**Log In**' button).



User and Group Management

The following pages contain information about user and group management in JIRA:

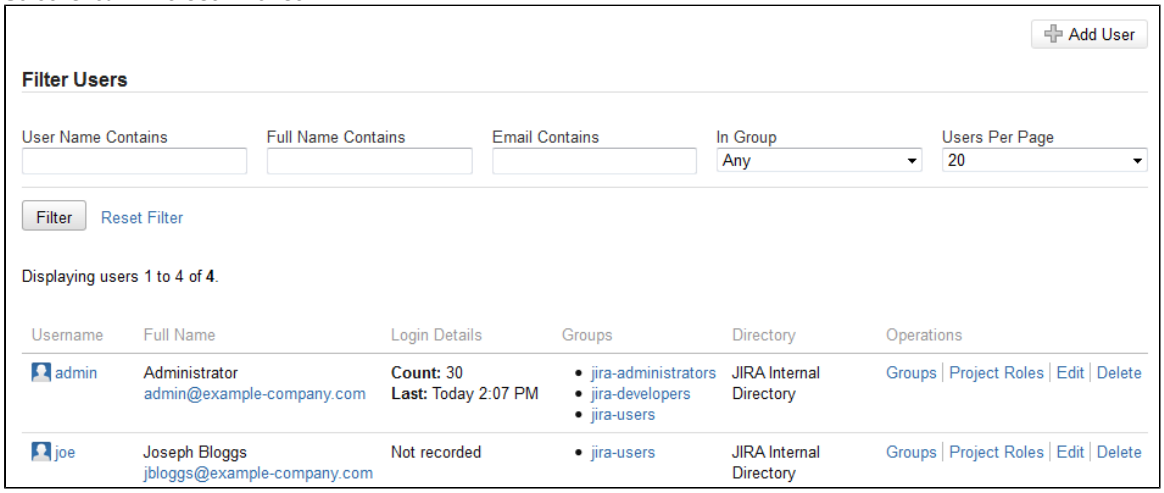
- Managing Users
- Managing Groups
- Managing Project Roles
- Migrating User Groups to Project Roles
- Configuring User Directories
- Viewing User Sessions
- Clearing 'Remember my login' Tokens
- Enabling Public Signup and CAPTCHA

Managing Users

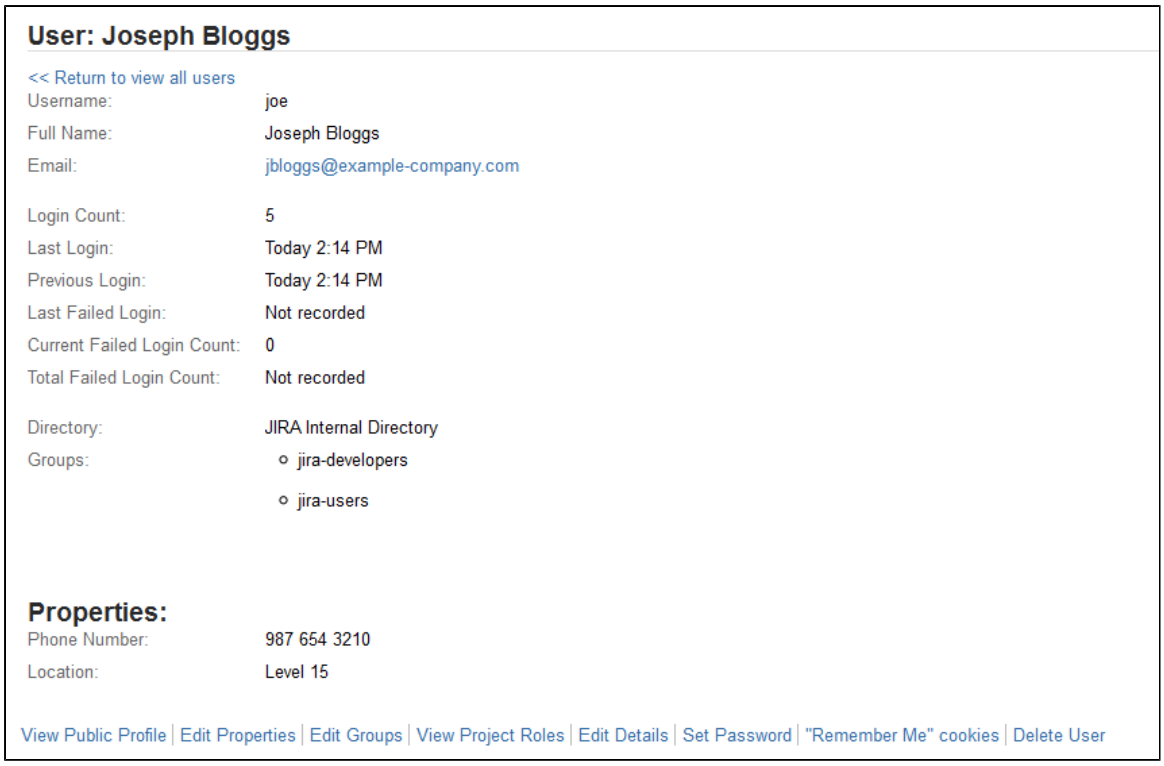
Viewing Users

To view a list of JIRA users:

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Select **Administration > Users > Users** to open the 'User Browser' page.
   Keyboard shortcut: **g + g +** start typing **users**
   Screenshot 1: The User Browser



3. To restrict the list of users shown in the **User Browser**, use the Filter form at the top of the User Browser.
   Specifying (part of) the user's username, full name, email address and/or group membership, then clicking the **Filter** button, will reduce the list to only those users who match those criteria.
4. To view details and login information about a user in the list, click their **Username** or **Email** Address.
   Screenshot 2: User Details

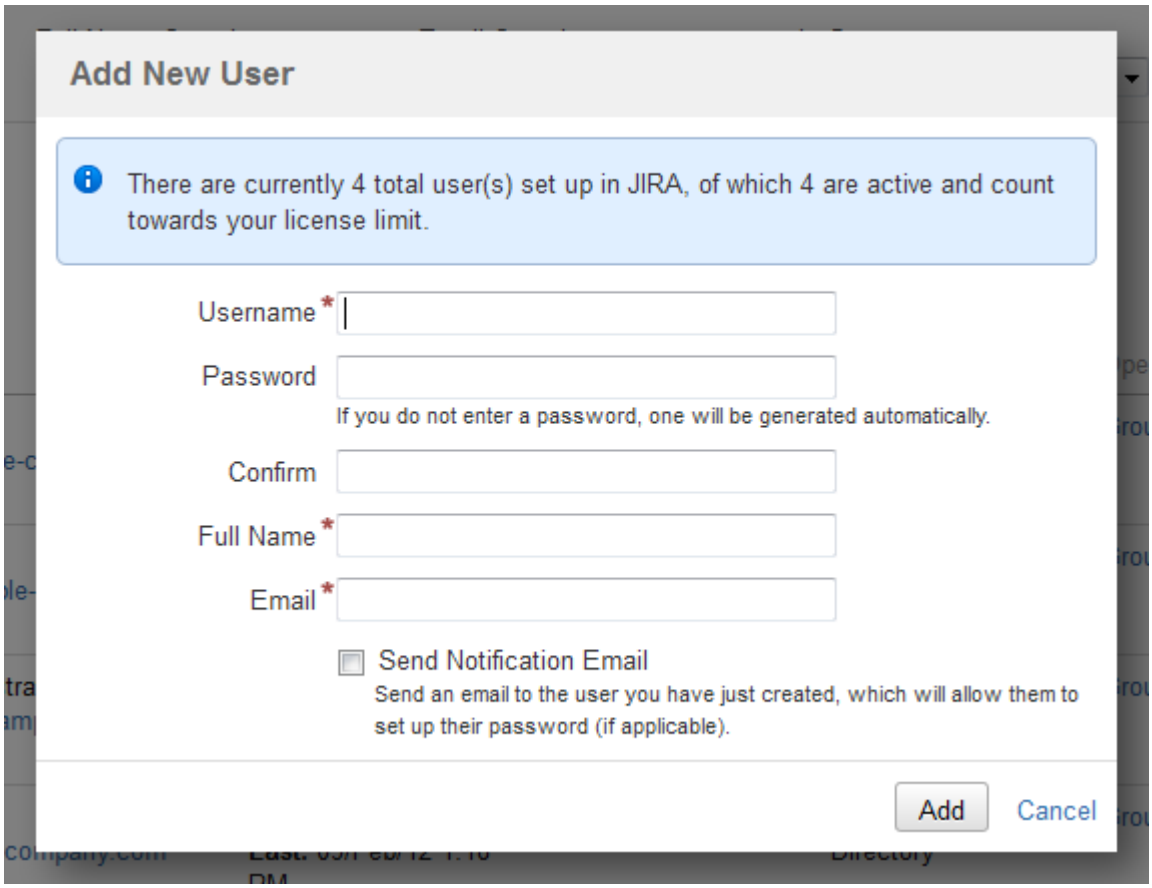


On this page:

- Viewing Users
- Adding a User
- Assigning a User to a Group
- Assigning a User to a Project Role
- Changing a User's Name or Email Address
- Changing a User's Password
- Adding a Property to a User
- Deactivating a User
- Deleting a User
- Notes

Adding a User

1. Open the **User Browser** (see Viewing Users above) and click the **Add User** button to open the 'Add New User' dialog box.
2. Enter the **Username** (note that this value cannot be changed once the user is created), **Password**, **Full Name** and **Email** address.



3. Optionally, select the **Send Notification Email** check box to send the user an email containing:
    - their login name; and
    - a link from which to set their password (this link is valid for 24 hours).
4. Click the **Create** button.

Users can also be created via:

- **Signup** — see Enabling Public Signup.
- **Email** — e.g. you can use a mail handler to allow JIRA to create new users based on the sender's email address (of email messages processed by JIRA). See Creating Issues and Comments from Email for more information about how to configure mail handlers for this purpose.
- **Connecting to an Internal Directory with LDAP Authentication** — see Copying Users on First Login.

Please Note: If you have a user limited license (e.g. starter license) and have reached your user limit, any further users added will not have permission to log in to JIRA.

Assigning a User to a Group

When a user is created, they will be added to any groups that are set up to have new users automatically added to them.

To change a user's group membership:

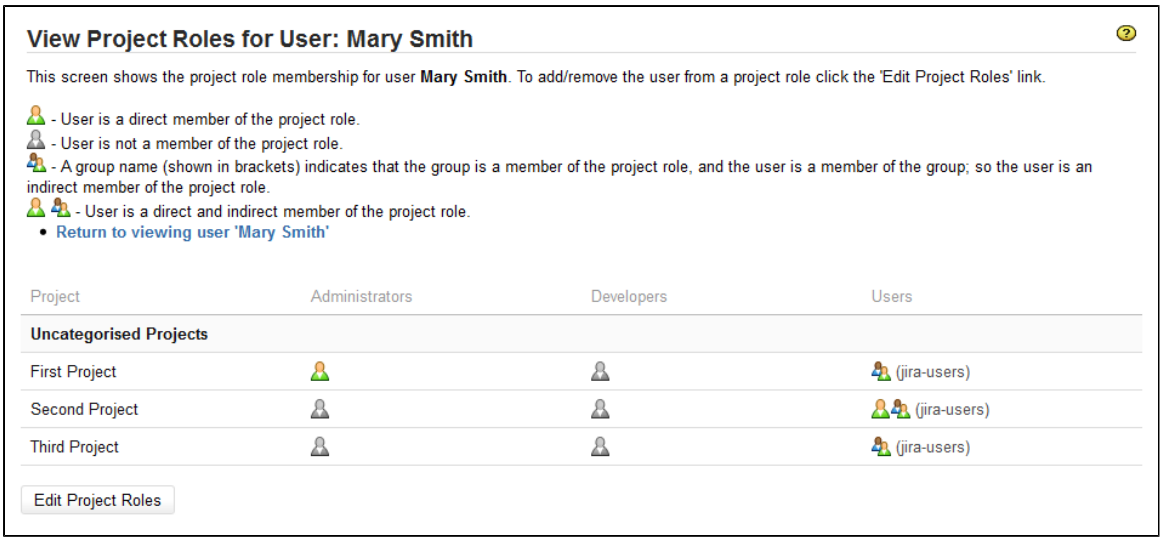
1. Locate the user in the **User Browser** (see Viewing Users above) and click the **Groups** link in the **Operations** column.
2. This will display two lists; the one on the left shows all available groups, and the one on the right shows all groups to which the user currently belongs. Use the **Join** and **Leave** buttons to add the user to or remove them from your selected group.

Please Note: If you have a user limited license (e.g. starter license) and have reached your user limit, you will not be able to assign any further users to groups with login permissions (i.e. jira-users permission) without first reducing the number of users with login permissions.

Assigning a User to a Project Role

Assigning a user to a project role enables them to fulfil a particular function in a particular project.

To view a user's project role membership, locate the user in the **User Browser** (see Viewing Users above) and click the **Project Roles** link in the **Operations** column. This will display a table showing all the projects and project roles that exist in JIRA, and the user's current project role membership for each project:



e.g. this screenshot shows that, for the 'First Project' project:

- Mary is a member of the 'Administrators' project role.
- Mary is not a member of the 'Developers' project role.
- Mary is indirectly a member of the 'Users' project role, through being a member of the 'jira-users' group.
  (Also note that, for the 'Third Project' project, Mary is both a direct *and* an indirect member of the 'Users' project role.)

Click the **Edit Project Roles** button. The check boxes will then be available for you to select (to add the user to a project role) or clear (to remove the user from a project role).

Changing a User's Name or Email Address

1. Locate the user in the **User Browser** (see Viewing Users above) and click their **Edit** link in the **Operations** column. This displays a form where you can change the user's Full Name or Email Address.
2. Click **Update** to confirm the change.

Changing a User's Password

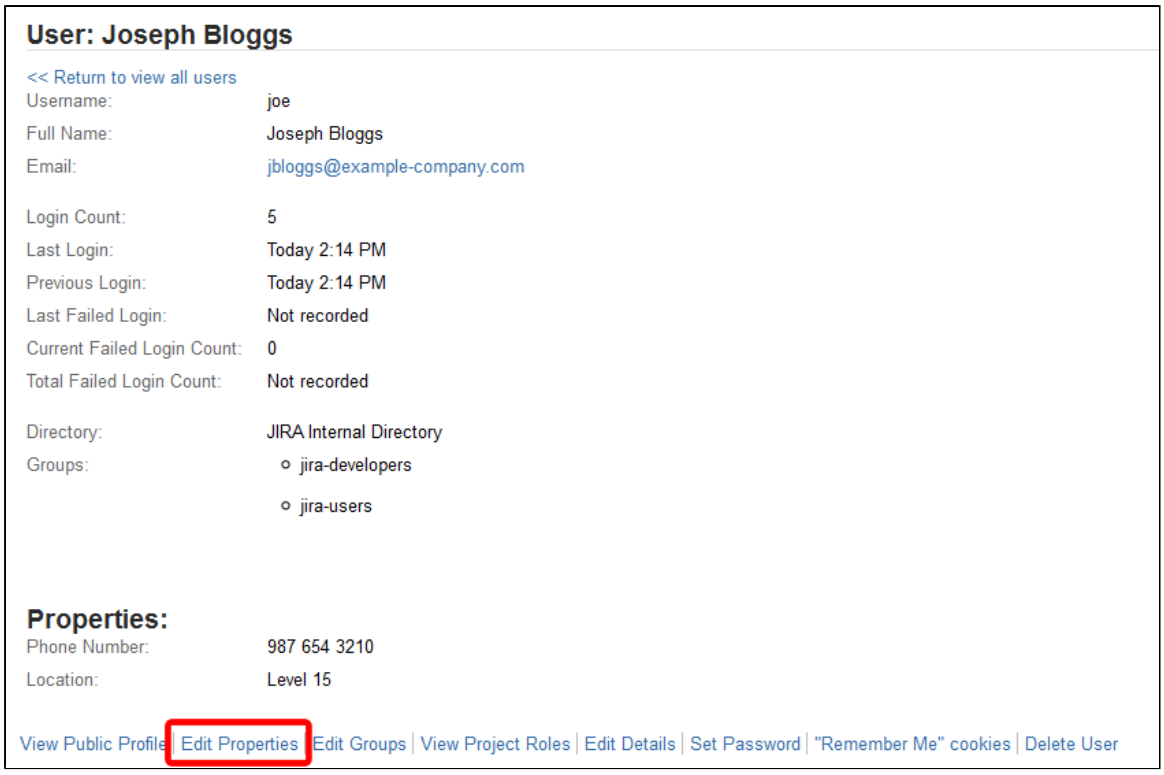
1. Locate the user in the **User Browser** (see Viewing Users above) and click their **Username**. This displays the user's details, below which are several links.
2. Click the **Set Password** link. This displays the **Set Password** screen.
3. Enter and confirm the new password.
4. Click the **Update** button.

Adding a Property to a User

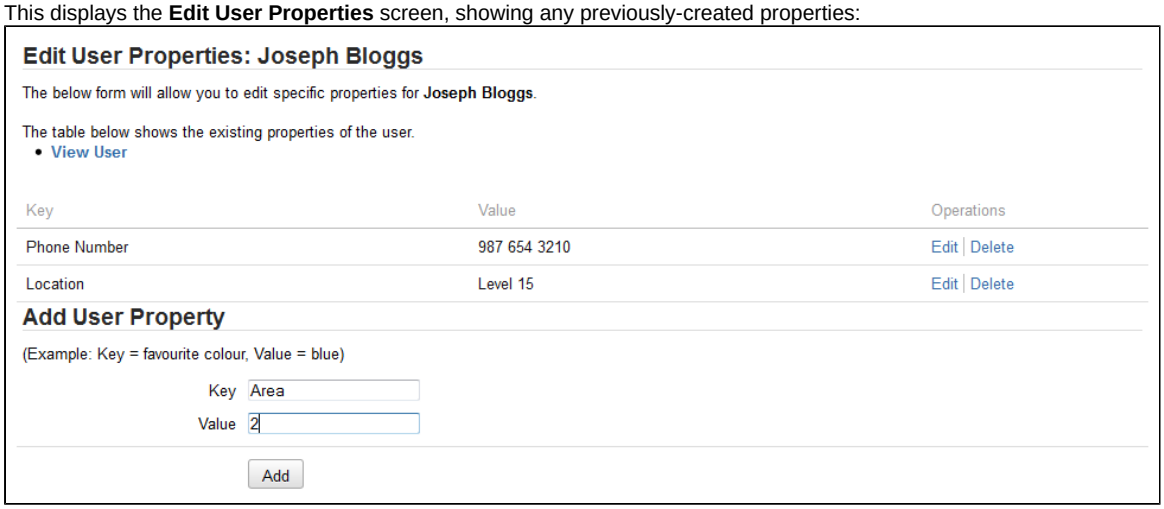
A **'Property'** is an extra piece of information that you can store regarding a user. A Property consists of a **Key** of your choice (eg. 'Phone number', 'Location') plus a corresponding **Value** (eg. '987 654 3210', 'Level Three').

To create a new Property for a user:

1. Locate the user in the **User Browser** (see Viewing Users above) and click their **Username**. This displays the user's details in a box.
2. Click the **Edit Properties** link towards the end of the page.



3. This displays the **Edit User Properties** screen, showing any previously-created properties:



4. Enter the new **Key** and its **Value**, then click the **Add** button.

Deactivating a User

Before you deactivate a user, you should:

- **Reassign any open issues assigned to that user**. You will need the '**Assign Issue**' permission to change the assignee for the issues.
- **Make sure the user is not the 'Default Assignee' for any project(s)**. You will need the '**Administer Project**' permission to change the Default Assignee for the project(s).

To deactivate a user account:

1. Remove the user from all groups. Read Managing Groups for more information.
2. Remove the user from all project roles. Read Managing Project Role Membership for more information.

Deactivating the user account will result in the following:

- the user will not count towards your **license limit**.
- **work log entries** associated with the user will remain.
- **filter subscriptions** will continue to be sent to the user — If this is a problem, you can change the user's email address in JIRA to an imaginary address, e.g. `user@example.com`.

There is currently no *"disabled user"* user type in JIRA.

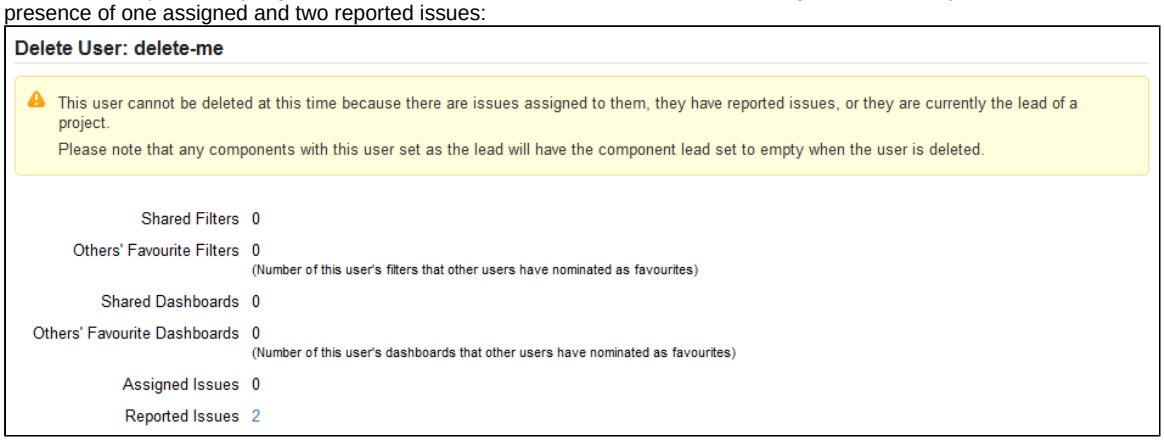
Deleting a User

Rather than deleting a user, we recommend that you **deactivate their account** instead (as described above). Deactivating a user's account will prevent that account from being used and prevent anyone from being able to log in to JIRA using that account. However, it will preserve that user's issues history.

Please Note: Before you delete a user, you should bulk-edit the issues involved and **change the reporter** to someone else. You will need the **'Modify Reporter'** permission to change the reporter for the issues. You will also need to allow editing of closed issues if some of the issues the user created are closed and you do not wish to reopen them.

To delete a user:

1. Locate the user in the **User Browser** (see Viewing Users above) and click the **Delete** link in the **Operations** column.
2. The confirmation screen that follows will summarise any involvement of that user in the system by showing current issues assigned to and reported by that user, etc. These connections between the user and other parts of the system may prevent the deletion of that user. For example, attempting to delete a user called *test-user* results in the following screen, which prevents deletion due to the presence of one assigned and two reported issues:



   As well as reassigning any issues, you may need to bulk-edit the issues created by the user and change the 'Reporter' to someone else. You'll need the 'Modify Reporter' permission to do this.
3. If there are no issues assigned to, or reported by the user, the confirmation screen will display a **Delete** button; click this to proceed with the deletion.

Please Note:

- Please note that the filters and dashboards of a user will be deleted when the user is deleted, regardless of whether the filters or dashboards are shared with other users.
- Any numbers of issues which have been reported by or assigned to the user you are attempting to delete, are respectively hyperlinked to a list of the individual issues (in the Issue Navigator).

Notes

- If you are using External User Management, you will not be able to create, edit or delete users from within JIRA; but you can still assign users to project roles, and create/edit/delete user properties.
- If you have JIRA connected to either a delegated LDAP directory or an LDAP directory set to 'Read Only' (see Connecting to an LDAP Directory for details), you will not be able to change a user password from within JIRA.
- **Multiple user directories:** You may define multiple user directories in JIRA, so that JIRA looks in more than one place for its users and groups. For example, you may use the default JIRA **internal directory** and also connect to an **LDAP** directory server. In such cases, you can define the **directory order** to determine where JIRA looks first when processing users and groups. Here is a summary of how the directory order affects the processing:
  - The order of the directories is the order in which they will be searched for users and groups.
  - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

  See Managing Multiple Directories.

Managing Groups

A JIRA group is a convenient way to manage a collection of users. Users can belong to many groups. Groups are used throughout JIRA; for example, they can:

- be granted global permissions.
- be used in project permission schemes.
- be used in email notification schemes.
- be used in issue security levels.
- be given access to issue filters.
- be given access to dashboards.
- be used in workflow conditions.
- belong to project roles *.

** Project roles are somewhat similar to groups, the main difference being that group membership is global whereas project role membership is project-specific.*

On this page:

- JIRA's default groups
- Viewing groups
- Adding a group
- Deleting a group
- Editing group membership
- Automatic group membership
- Notes

JIRA's default groups

When you install JIRA, three groups are automatically created:

Group	Description
jira-administrators	Typically contains people who are JIRA system administrators. By default, this group: • is a member of the **'Administrators'** project role. • has the **'JIRA Administrators'** and the **'JIRA System Administrators'** global permissions. if you need to give these permissions to separate people, you will need to create an additional group and grant the permissions separately, as described in *'About 'JIRA System Administrators' and 'JIRA Administrators' '*.)
jira-developers	Typically contains people who perform work on issues. By default, this group: • is a member of the **'Developers'** project role. • has the **'Browse Users', 'Create Shared Filter'** and **'Manage Group Filter Subscriptions'** global permissions.
jira-users	Typically contains every JIRA user in your system. By default, this group: • is a member of the **'Users'** project role. • has the **'JIRA Users'** and **'Bulk Change'** global permissions.

You can create and delete groups according to your organisation's requirements.

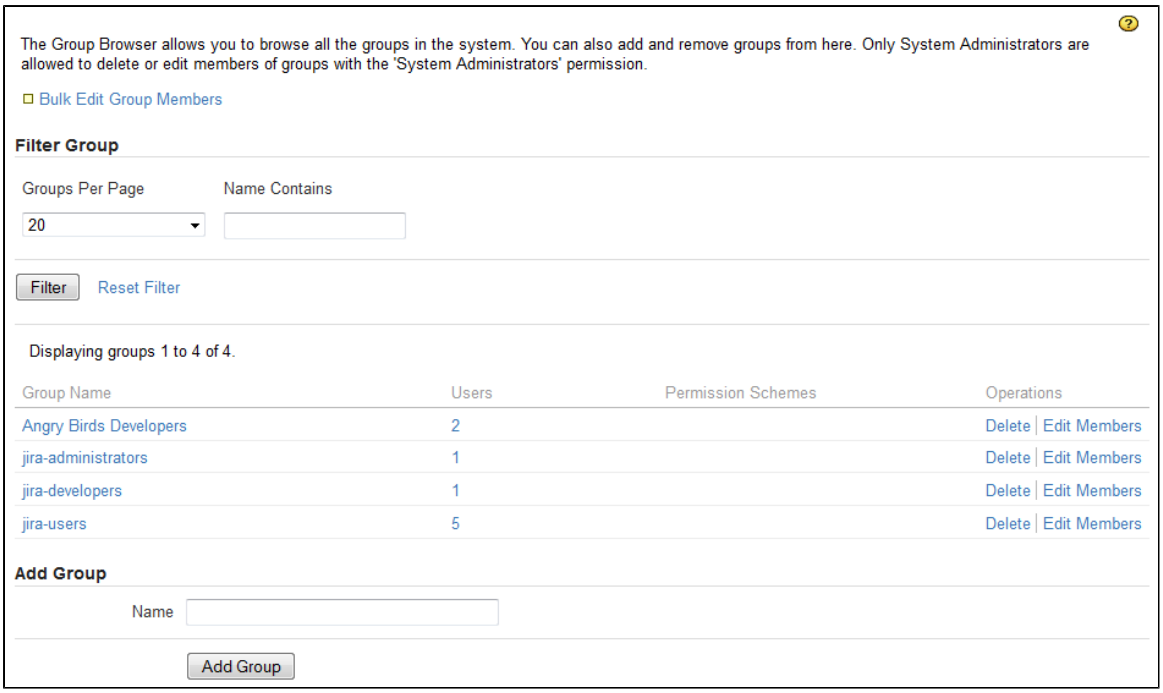
Please Note: If you are using External User Management, you will not be able to create, delete or edit groups or group membership from within JIRA; and 'Automatic Group Membership' (see below) will not apply. However, you can still assign groups to project roles.

Viewing groups

To see what groups exist, and where they are used:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Users' > 'Groups'** to open the 'Group Browser' page as shown in the screenshot below.
   Keyboard shortcut: **'g' + 'g' +** start typing **'groups'**
3. To see which permission schemes, email notification schemes, issue security levels and saved filters are using this group, click the group name.

Screenshot 1: the Group Browser



Please Note: The **'Filter Group'** form restricts the list of groups shown to those that match the **'Name Contains',** with a specified maximum per page. Click the **'Filter'** button to refresh the list with the restricting filter.

Adding a group

To create a group, enter the new group **'Name'** in the **'Add Group'** form in the **'Group Browser'** (see *'Viewing groups'* above) and click the **'Add Group'** button.

Deleting a group

To delete a group, click the **'Delete'** link for that group in the **'Group Browser'** (see *'Viewing groups'* above). The confirmation screen that follows explains that users will be removed from the group through its deletion. Be aware of the impact this may have on users in that group. For example, if that group membership was the sole conveyor of a permission for a user, then the user will no longer have that permission.

Please Note: Before deleting a group it is recommended that you check whether the groups is being used by any permission schemes, email notification schemes, issue security levels or saved filters. See *'Viewing groups'* (above).

Editing group membership

To edit a group's membership, click the **'Edit Members'** link in the row for that group in the **'Group Browser'** (see *'Viewing groups'* above). This takes you to a form allowing you to add users to or remove them from the group.

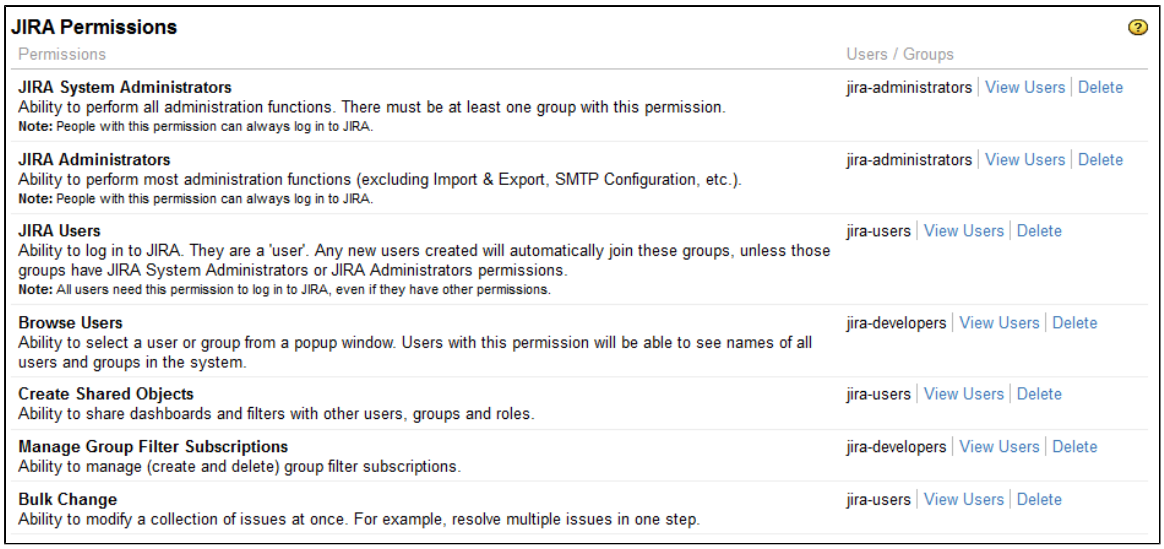
Please Note:

- If the group has the **'JIRA System Administrators'** global permission, you cannot edit its membership unless you have the **'JIRA System Administrators'** global permission.
- If you have a user limited license (e.g. personal license) and have reached your user limit, you will not be able to assign any further users to groups with login permissions (i.e. jira-users permission) without first reducing the number of users with login permissions.

Automatic group membership

To automatically add newly-created users to a particular group, you can either:

- **Grant the group the 'JIRA Users' global permission.**
  Add the **'JIRA Users'** permission to the relevant group, as described in *'Granting global permissions'*.



- **Specify the group name in the 'Default Group Memberships' option when Connecting to an LDAP Directory.**
  See Adding Users to Groups Automatically.

Notes

- **Multiple user directories:** You may define multiple user directories in JIRA, so that JIRA looks in more than one place for its users and groups. For example, you may use the default JIRA **internal directory** and also connect to an **LDAP** directory server. In such cases, you can define the **directory order** to determine where JIRA looks first when processing users and groups. Here is a summary of how the directory order affects the processing:
  - The order of the directories is the order in which they will be searched for users and groups.
  - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

  See Managing Multiple Directories.

- **Nested groups:** Some directory servers allow you to define a group as a member of another group. Groups in such a structure are called 'nested groups'. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups. See Managing Nested Groups.

Managing Project Roles

Project roles are a flexible way to associate users and/or groups with particular projects. Project roles also allow for delegated administration

- Global administrators define JIRA's project roles — that is, all projects have the same project roles available to them.
- Project administrators assign members to project roles specifically for their project(s).

Project roles can be used in:

- permission schemes
- email notification schemes
- issue security levels
- comment visibility
- workflow conditions

Project roles can also be given access to:

- issue filters
- dashboards

Project roles are somewhat similar to groups, the main difference being that group membership is global whereas project role membership is project-specific. Additionally, group membership can only be altered by JIRA administrators, whereas project role membership can be alterec by project administrators.

A **project administrator** is someone who has the project-specific 'Administer Project' permission, but not necessarily the global 'JIRA Administrator' permission. A project administrator can also manage project role membership.

On this page:

- Using project roles
- JIRA's default project roles
- Viewing project roles
- Adding a project role
- Deleting a project role
- Editing a project role
- Assigning members to a project role
- Specifying 'default members' for a project role

Using project roles

Project roles enable you to associate users with particular functions. For example, if your organisation requires all software development issues to be tested by a Quality Assurance person before being closed, you could do the following:

1. Create a project role called **Quality Assurance**.
2. Create a permission scheme called **Software Development**, in which you assign the 'Close Issue' permission to the **Quality Assurance** project role.
3. Associate the **Software Development** permission scheme with all software development projects.
4. For each software development project, add the appropriate Quality Assurance people to the **Quality Assurance** project role.

JIRA's default project roles

When you install JIRA, three project roles are automatically created:

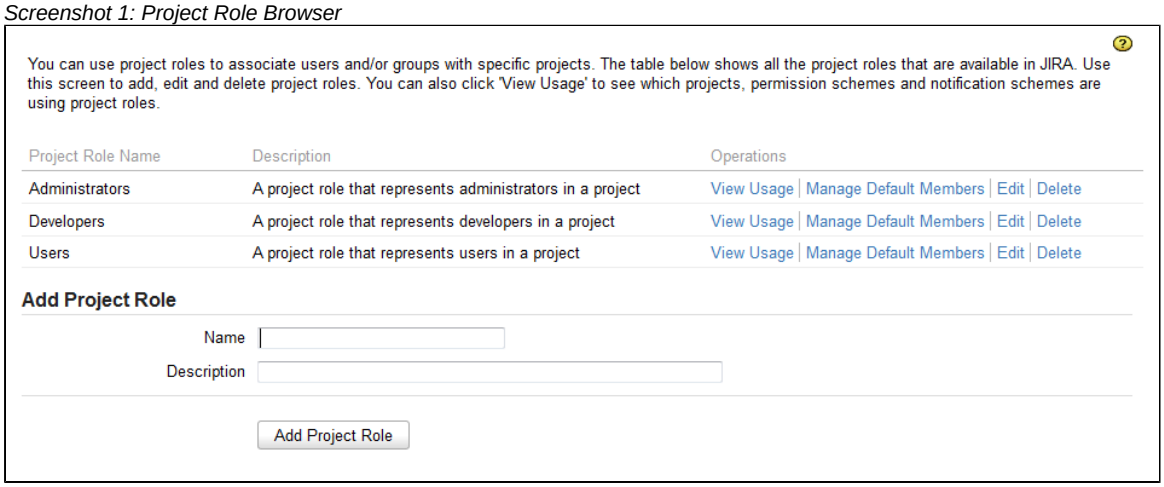
Project Role	Description
Administrators	Typically contains people who administer a given project.
Developers	Typically contains people who work on issues in a given project.
Users	Typically contains people who log issues in a given project.

You can create, edit and delete project roles according to your organisation's requirements.

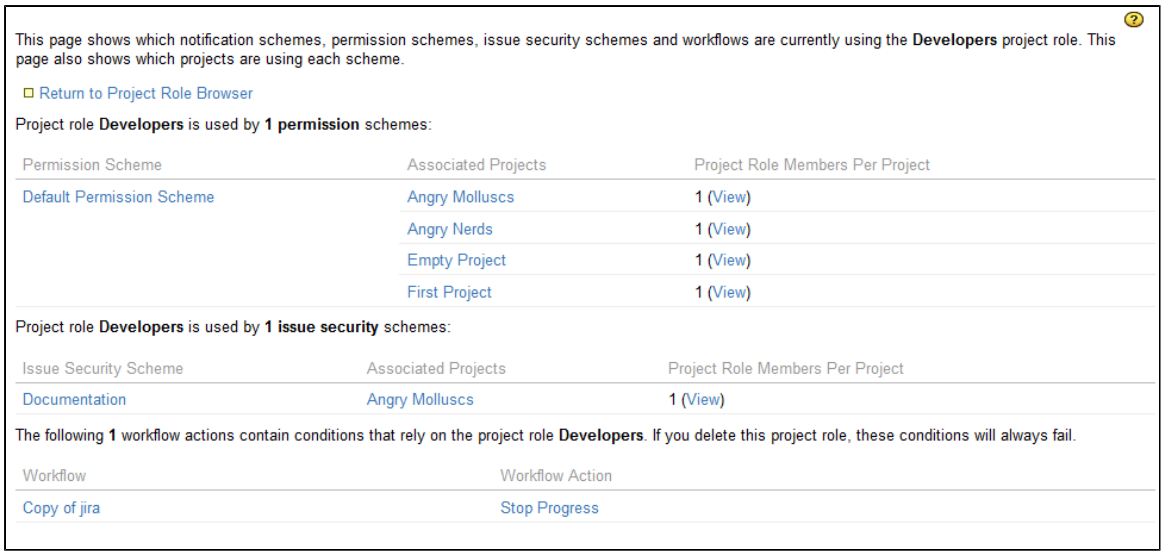
Viewing project roles

To see what project roles exist, and where they are used:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Users' > 'Roles'**.
   Keyboard shortcut: **'g' + 'g' +** start typing **'roles'**
3. You will then see the Project Role Browser, which contains a list of all the project roles in your JIRA system:
   Screenshot 1: Project Role Browser



4. To see where a project role is used, click the **'View Usage'** link. This will display a list of the project role's associated permission schemes, email notification schemes, issue security levels and workflow conditions:
   Screenshot 2: Project Role Usage


This page shows which notification schemes, permission schemes, issue security schemes and workflows are currently using the **Developers** project role. This page also shows which projects are using each scheme.

Return to Project Role Browser

Project role **Developers** is used by **1 permission** schemes:

Permission Scheme	Associated Projects	Project Role Members Per Project
Default Permission Scheme	Angry Molluscs	1 (View)
	Angry Nerds	1 (View)
	Empty Project	1 (View)
	First Project	1 (View)

Project role **Developers** is used by **1 issue security** schemes:

Issue Security Scheme	Associated Projects	Project Role Members Per Project
Documentation	Angry Molluscs	1 (View)

The following **1** workflow actions contain conditions that rely on the project role **Developers**. If you delete this project role, these conditions will always fail.

Workflow	Workflow Action
Copy of jira	Stop Progress

5. To see which users/groups are associated with a project role for a particular project, click the '**View**' link.

Adding a project role

To define a new project role, enter its Name and a Description in the **'Add Project Role'** form in the Project Role Browser (see 'Viewing Project Roles' above), and click the **'Add Project Role'** button. Note that project role names must be unique.

Once a new project role is created, it is available to all projects. Project administrators can then assign members to the project role for their project (see Managing project role membership).

Deleting a project role

To delete a project role, locate the project role in the Project Role Browser (see 'Viewing Project Roles' above), and click the **'Delete'** link. The confirmation screen that follows lists any permission schemes, email notification schemes, issue security levels and workflow conditions that use the project role.

Note that deleting a project role will remove any assigned users and groups from that project role, for all projects. Be aware of the impact this may have; for example, if the project role membership was the sole conveyor of a permission for a user, then the user will no longer have that permission.

If a project role has been used to specify who can view a comment, deleting the project role will mean that no-one can see that comment any more.

Editing a project role

To edit the **Name** and **Description** of a project role, locate the project role in the Project Role Browser (see 'Viewing Project Roles' above), and click the **'Edit'** link. This takes you to a form where you can modify the project role's **'Name'** and **'Description'**.

Assigning members to a project role

A project role's members are assigned on a project-specific basis. To assign users/groups to a project role for a particular project, please see Managing project role membership.

To see/edit *all* the project roles to which a particular user belongs, for all projects, click the '**Project Roles**' link in the User Browser.

Specifying 'default members' for a project role

The default members for a project role are users and groups that are initially assigned to the project role for all newly created projects. The actual membership for any particular project can then be modified by the project administrator.

The default members consist of the **Default Users** plus the **Default Groups** shown in the Project Role Browser (see 'Viewing Project Roles' above).

To add to the **Default Users** or the **Default Groups** for a project role, click the corresponding **'Edit'** link.

For example, if a user called Susie needs to have administration permissions for all newly created projects, you could add her to the **Default Users** for the 'Administrator' project role as follows:

1. Open the Project Role Browser.
2. Click the **'Manage Default Members'** link.
3. Click the **'Edit'** link in the **Administrators** column (next to *'None selected').*

4. In the 'Assign Default Users to Project Role' screen, click the '**User Picker**' icon.
5. Locate Susie in the 'User Picker' popup window, then click the '**Select**' button.
6. In the 'Assign Default Users to Project Role' screen, click the '**Add**' button.

Changing a project role's default members does not affect the actual project role members for projects already created.

Migrating User Groups to Project Roles

Project roles are a flexible way of associating particular users and groups with a particular project.

Why migrate to Project Roles?

- **Ease of management** — JIRA versions prior to 3.7 did not have project roles. If you previously used JIRA 3.6.x (or earlier), your system may contain multiple, project-specific groups, permission schemes and notification schemes. By implementing project roles, you may be able to reduce the number of groups, permission schemes and notification schemes in your JIRA system. This can make your system easier to manage.
- **Delegated administration** — A project administrator (that is, someone who has the 'Administer Project' permission, but not necessarily the global 'JIRA Administrator' permission) can assign users and groups to project roles for their project. If their project's permission scheme and notification scheme are using project roles, the project administrator can control who may access their project and who receives email notifications.

The instructions on this page will help you use Scheme Tools to:

- update your permission schemes and notification schemes so that they use project roles instead of groups; then
- minimise the number of permission schemes and notification schemes in your JIRA system.

On this page:

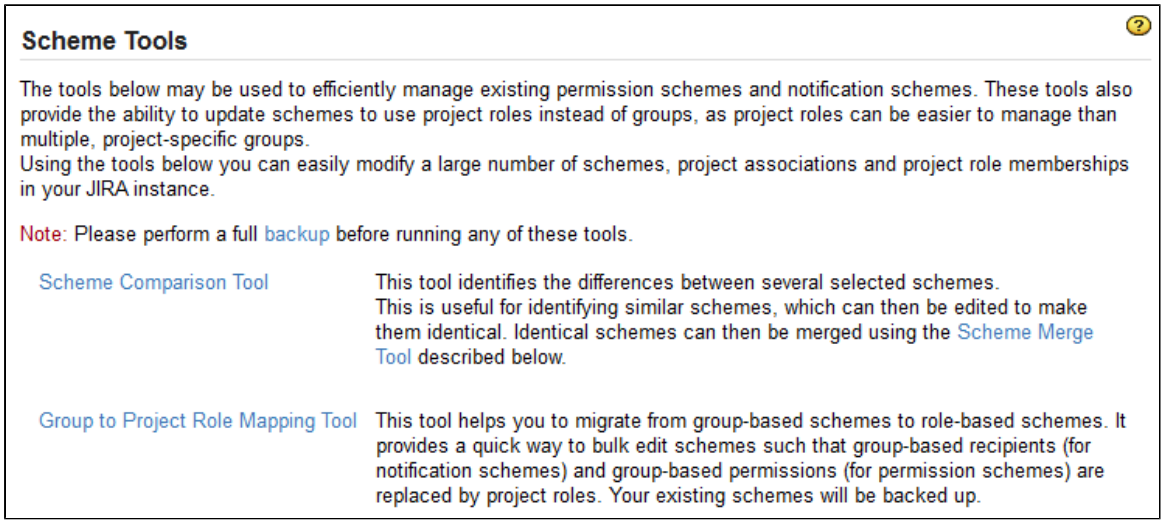
- Why migrate to Project Roles?
- Updating Permission Schemes and Notification Schemes to use Project Roles instead of Groups
- Minimising the number of Permission Schemes and Notification Schemes

Updating Permission Schemes and Notification Schemes to use Project Roles instead of Groups

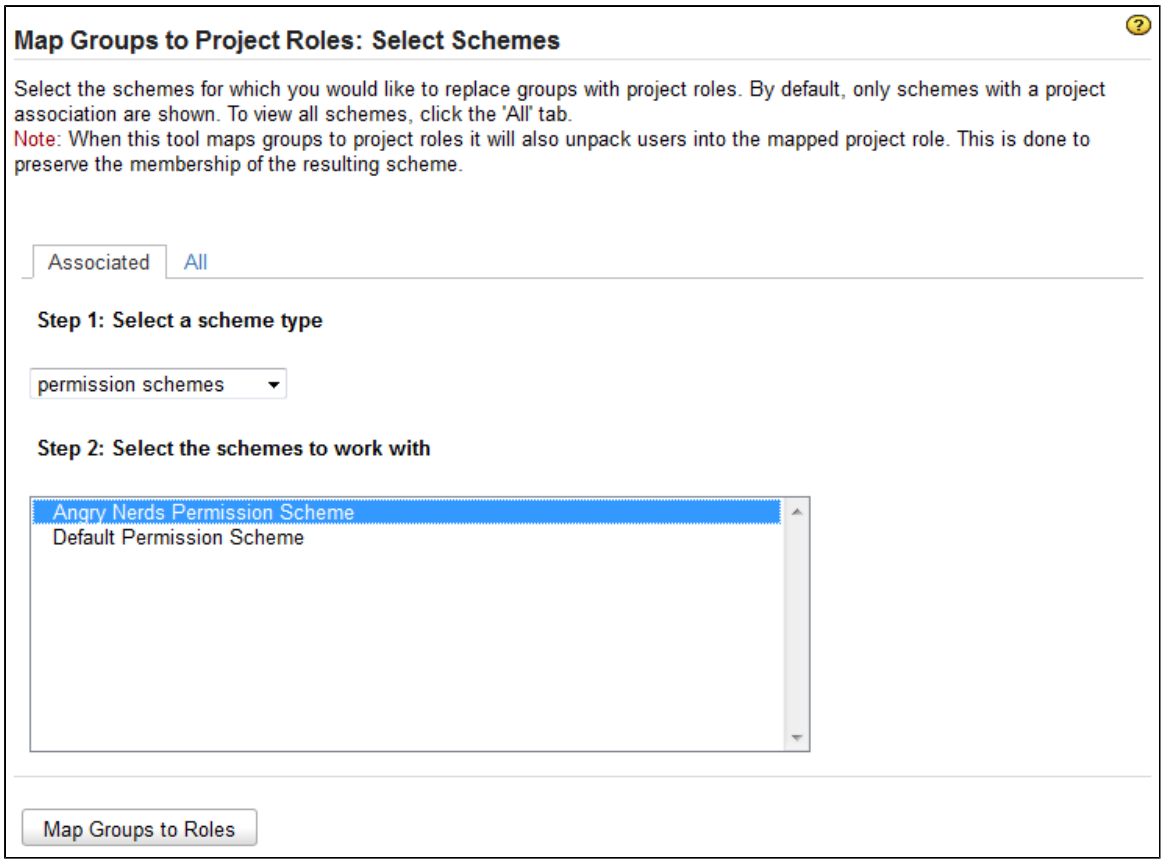
Backup your data

Before you begin, please perform a full backup.

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Advanced' > 'Scheme Tools'** to open the 'Scheme Tools' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'scheme tools'**
3. Click the **'Group to Project Role Mapping Tool'** link.
   Screenshot 1: 'Scheme Tools'

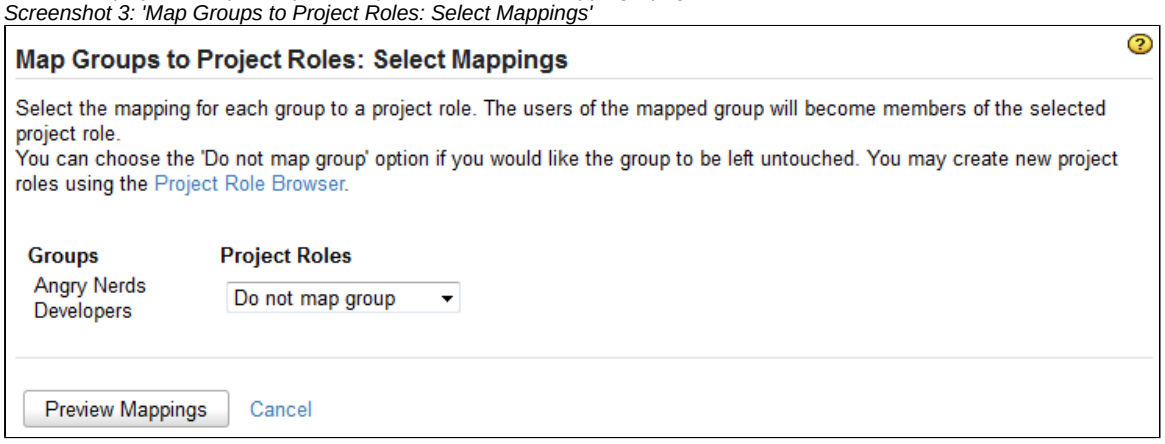


4. This will display the 'Map Groups to Project Roles: Select Schemes' page:
   Screenshot 2: 'Map Groups to Project Roles: Select Schemes'



- Note that schemes that are not associated with any projects need not usually be included in this process; but if you wish to select from all schemes in your system (including unused schemes), click '**All**'.
- Under **'Step 1: Select a scheme type'**, select whether you want to update permission schemes or notification schemes. (You can only do one type of scheme at a time, but you can easily come back and do the other type later).
- Under **'Step 2: Select the schemes to work with'**, select the schemes you want to update to use project roles instead of groups. You can use the **'Ctrl'** key to select multiple schemes.
- Click the '**Map Groups to Roles**' button.

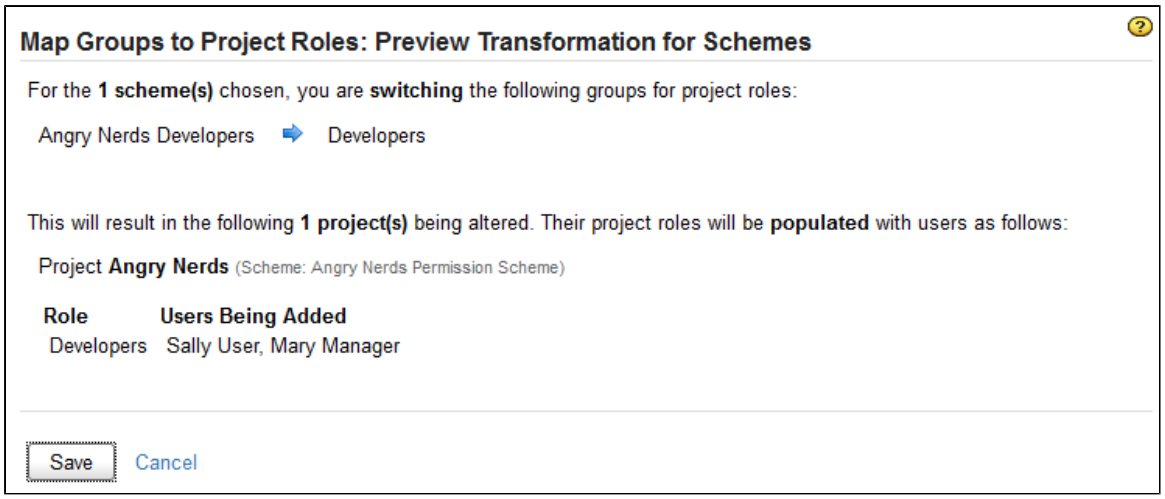
5. This will display the 'Map Groups to Project Roles: Select Mappings' page:
Screenshot 3: 'Map Groups to Project Roles: Select Mappings'



For each group, select the project role that will replace it; or, for any groups that you do not want to migrate, choose the '**Do not map group**' option. Then click the '**Preview Mappings**' button.

- For ease of maintenance, it is recommended that you do not migrate any groups to which JIRA users are automatically added (that is, groups which have the 'JIRA Users' global permission). If you migrate these groups to project roles, and you still want all new users to have access to particular projects, you will need to manually add new users to the relevant project role for each project.

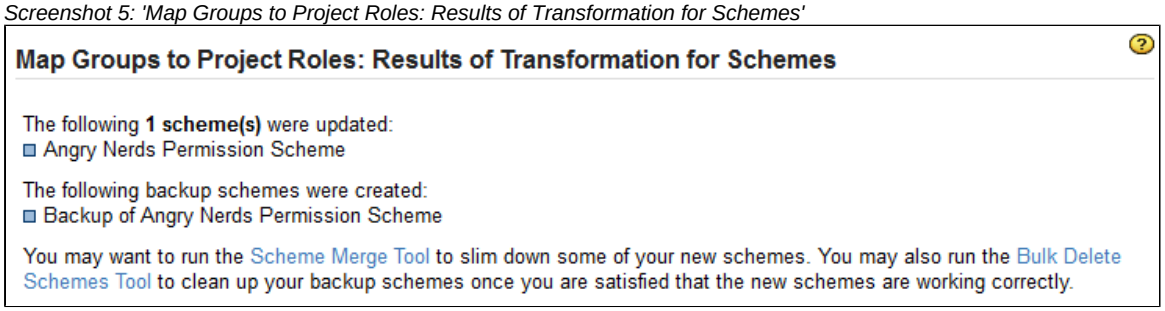
6. You will now see the 'Map Groups to Project Roles: Preview Transformation for Schemes' page:
Screenshot 4: 'Map Groups to Project Roles: Preview Transformation for Schemes'



If you are satisfied that the information shown on this page is correct, click the '**Save**' button to:

- create a backup of the scheme(s) that you selected in step 5 (you can later delete this backup scheme by using the 'Bulk Delete Schemes Tool', available from the 'Scheme Tools' page shown in step 4). This backup scheme will not be associated with any projects.
- update the scheme(s) that you selected in step 5 to use the role (left of the blue arrow) instead of the group (right of the blue arrow)
- add the users (in the right column of the table) to the project role (in the left column of the table) for each project that uses the scheme. This ensures that all users will continue to have the same permissions and notifications.

7. You will now see confirmation of the above changes on the 'Map Groups to Project Roles: Results of Transformation for Schemes' page:
Screenshot 5: 'Map Groups to Project Roles: Results of Transformation for Schemes'



After updating your permission schemes and notification schemes to use project roles instead of groups, you may find that many of your schemes are now very similar. To identify such schemes, merge them, and delete any redundant ones, please see Minimising the number of Permission Schemes and Notification Schemes (below).

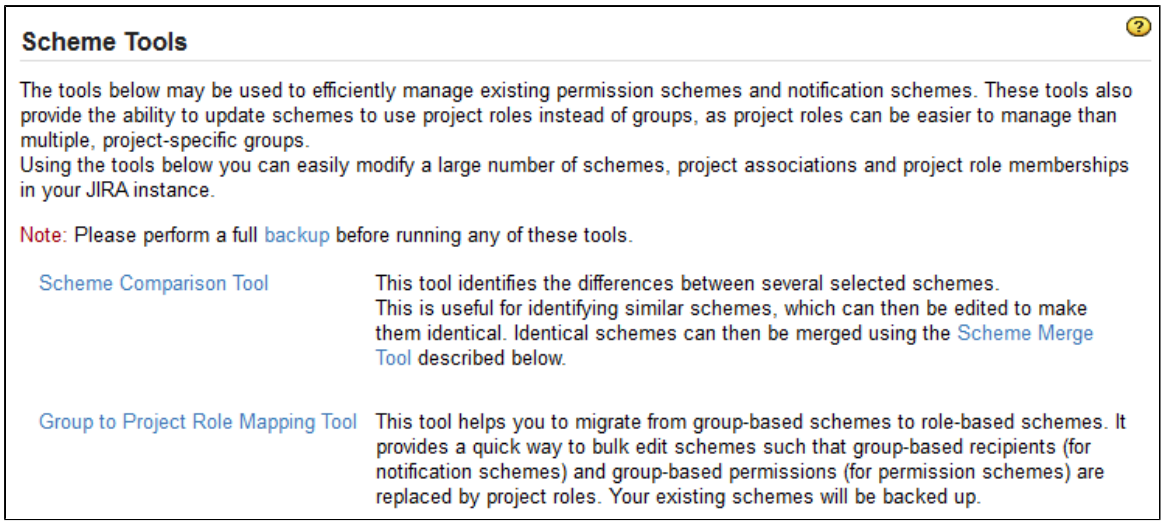
You may also find that some groups are no longer required. You can use the Group Browser to identify and delete groups that are not used by any permission schemes or notification schemes.

Minimising the number of Permission Schemes and Notification Schemes

Minimising the number of permissions schemes and notification schemes can make your JIRA system easier to manage. To identify and remove unnecessary schemes, follow the steps below:

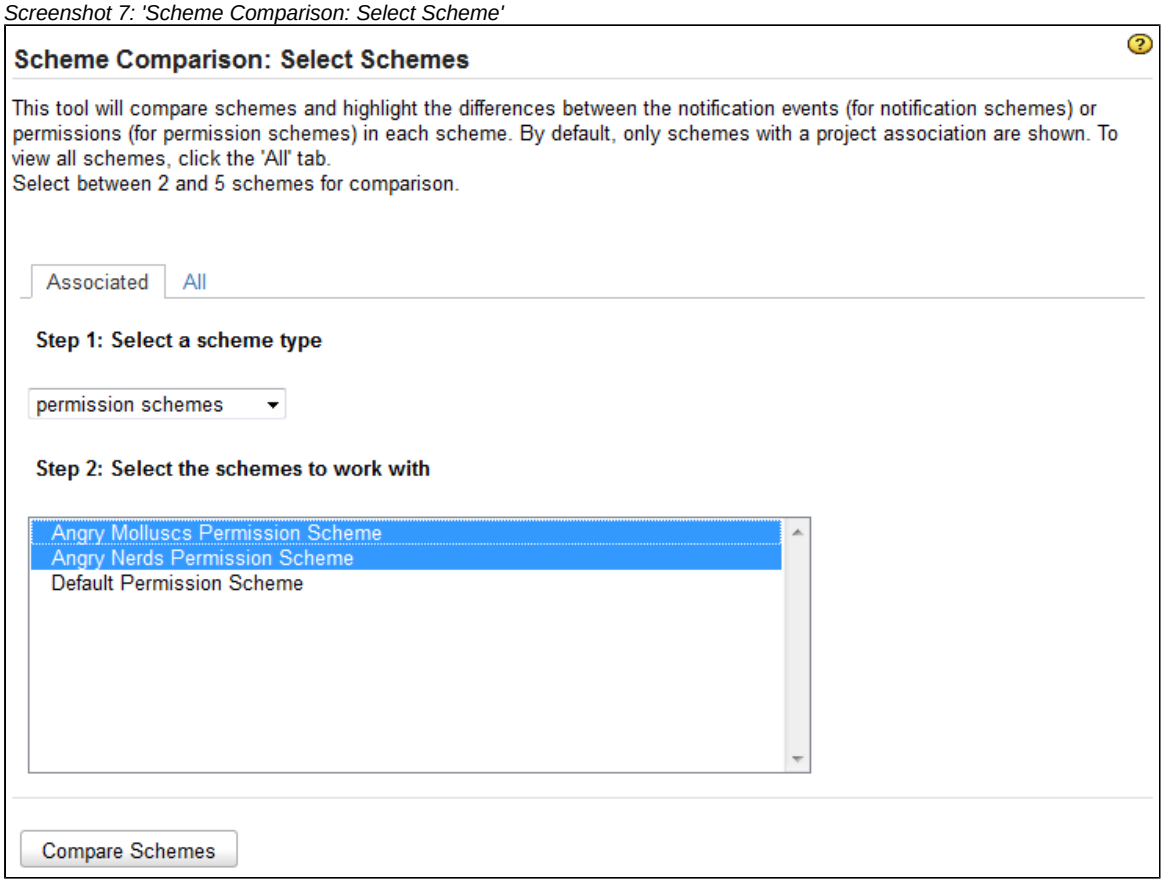
> Before you begin, please perform a full backup.

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Advanced' > 'Scheme Tools'** to open the 'Scheme Tools' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'scheme tools'**
3. Click the '**Scheme Comparison Tool**' link.
   Screenshot 6: 'Scheme Tools'



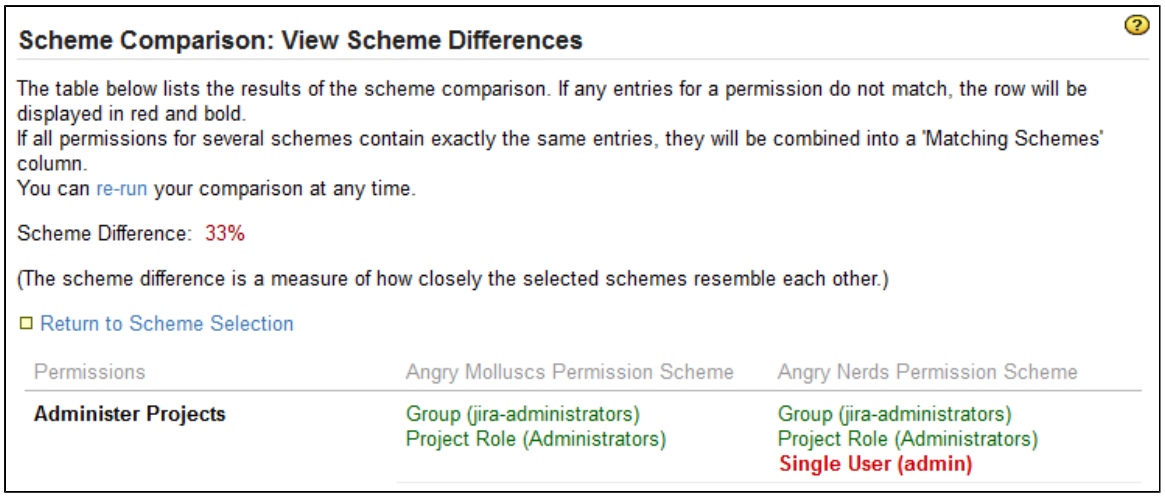
- The Scheme Comparison Tool assists you in identifying similar schemes, and if appropriate, making them identical.
- Identical schemes can later be merged using the Merge Duplicate Schemes Tool (see step 9 below).

4. This will display the 'Scheme Comparison: Select Scheme' page:
Screenshot 7: 'Scheme Comparison: Select Scheme'



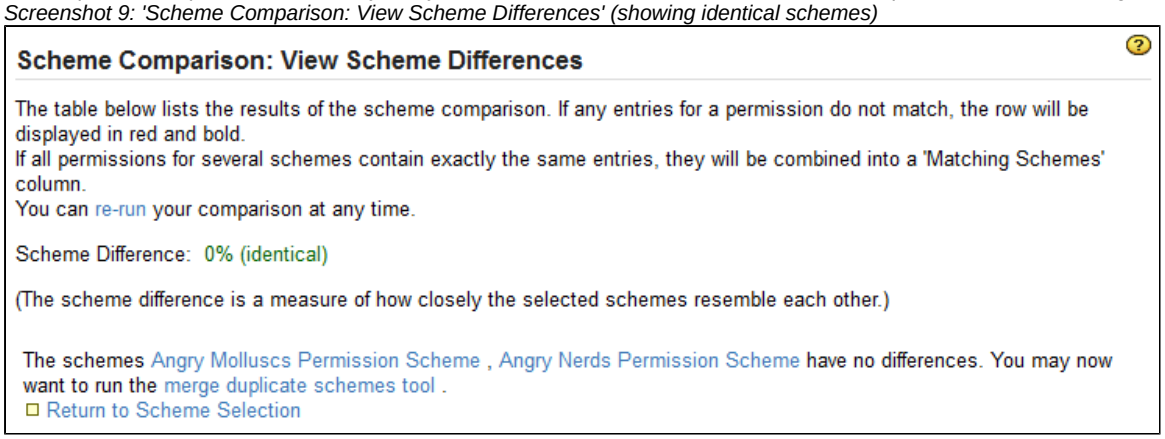
- Note that schemes which are not associated with any projects need not usually be included in this process; but if you wish to select from all schemes in your system (including unused schemes), click '**All**'.
- Under '**Step 1: Select a scheme type',** select whether you want to compare permission schemes or notification schemes. (You can only do one type of scheme at a time, but you can easily come back and do the other type later.)
- Under **Step 2: Select the schemes to work with,** select the schemes you want to compare. Select at least 2 (and no more than 5) schemes, using the **'Ctrl'** key to select multiple schemes.
- Click the '**Compare Schemes**' button.

5. This will display the 'Scheme Comparison: View Scheme Differences' page:
Screenshot 8: 'Scheme Comparison: View Scheme Differences'

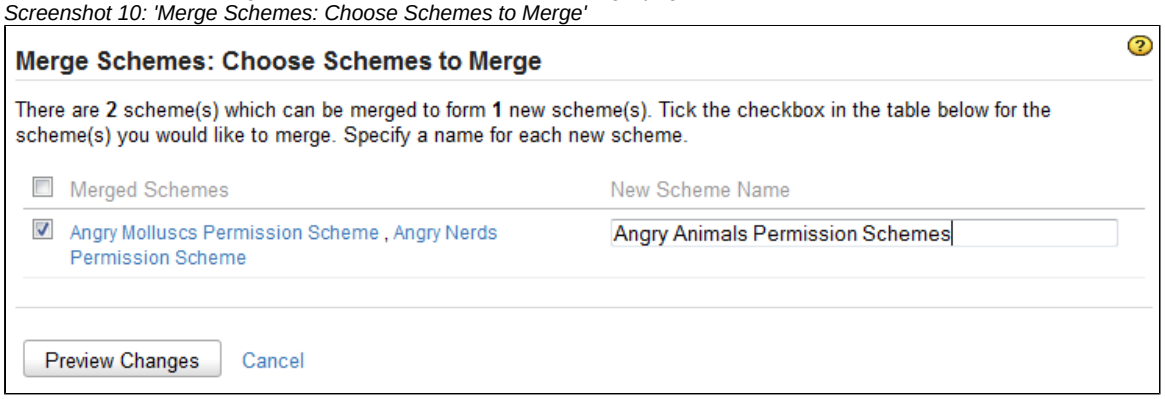


- Only the **differences** between the selected schemes are displayed. For example, in the screenshot above, only the "Administer Projects" permission is displayed; this means that all the other permissions in these two permission schemes ("Angry Nerds Permission Scheme" and "Angry Molluscs Permission Scheme") are identical.

6. If you decide it is appropriate to edit a scheme to make it the same as another one, you can edit the scheme by clicking the scheme name. For example, it may be appropriate to delete **Single User (marym)** from the "Angry Nerds Permission Scheme" if she is a member of the "Administrators" project role for the Angry Nerds project.
7. Then repeat the steps above, and verify that you have achieved a batch of 2 or more identical permission schemes, e.g.
Screenshot 9: 'Scheme Comparison: View Scheme Differences' (showing identical schemes)



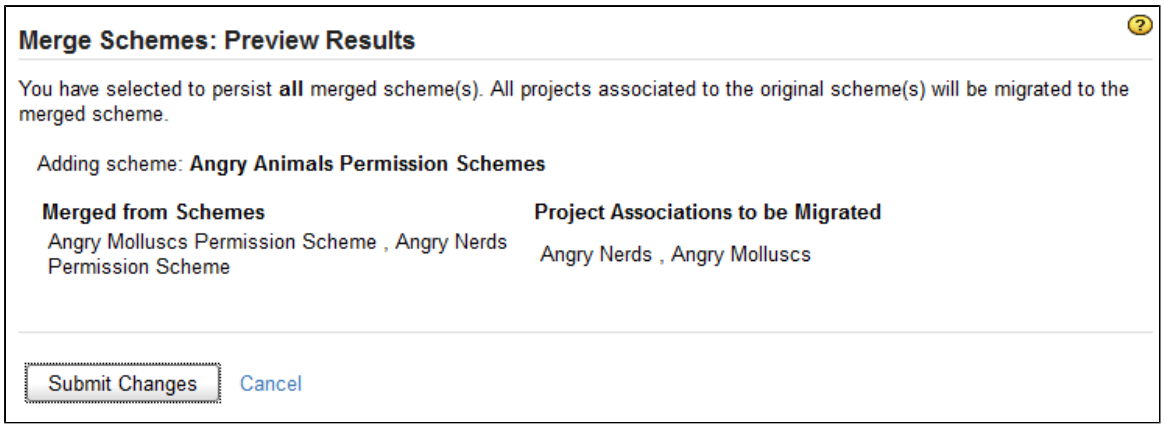
8. Click the '**Merge Duplicate Schemes Tool**' link. (Note: this link is also available from the 'Scheme Tools' page shown above).
9. You will now see the 'Merge Schemes: Choose Schemes to Merge' page:
Screenshot 10: 'Merge Schemes: Choose Schemes to Merge'



If you decide it is appropriate to merge the displayed schemes:

- Check the box next to the scheme names.
- Type a name for the new scheme in the '**New Scheme Name**' box.
- Click the '**Preview Changes**' button.

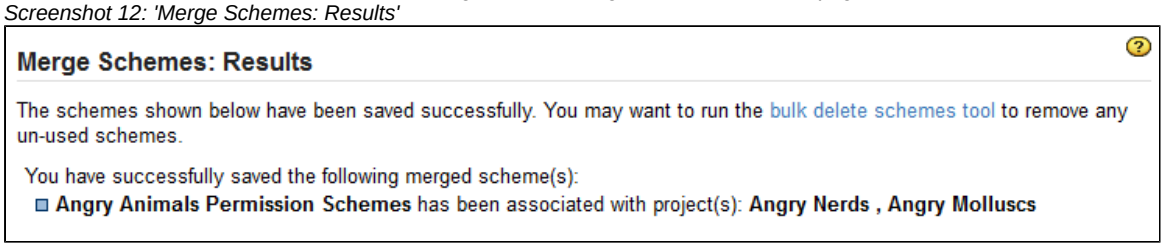
10. You will now see the 'Merge Schemes: Preview Results' page:
Screenshot 11: 'Merge Schemes: Preview Results'



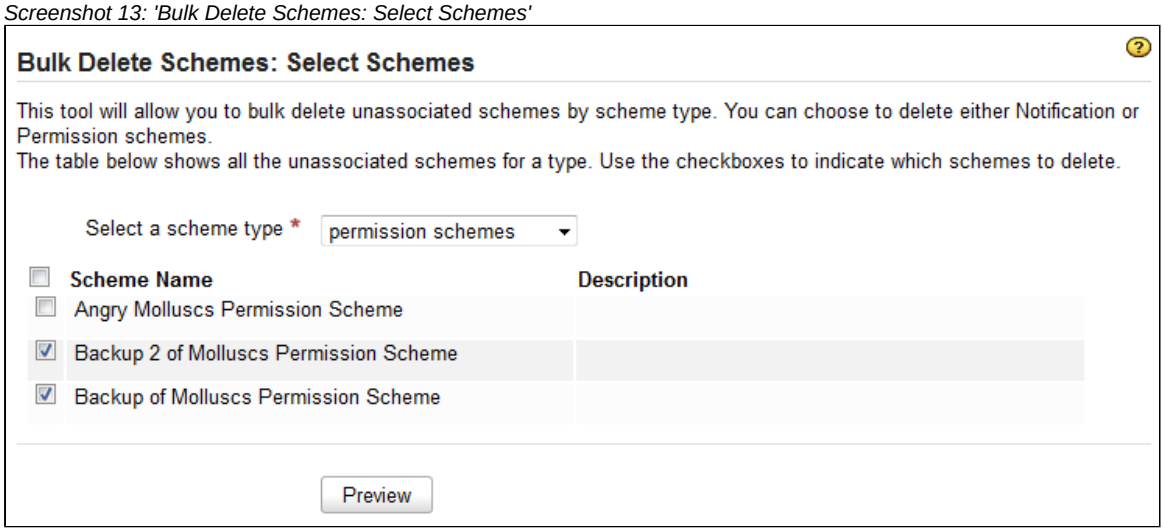
If you are satisfied that the information shown on this page is correct, click the '**Submit Changes**' button to:

- create the new scheme whose name is shown in bold.
- associate the projects (in the right column of the table) with the new scheme.
- disassociate the existing schemes (in the left column of the table) from the projects. These schemes can then be deleted using the 'Bulk Delete Schemes Tool' (see below).

11. You will now see confirmation of the above changes on the 'Merge Schemes: Results' page:
Screenshot 12: 'Merge Schemes: Results'



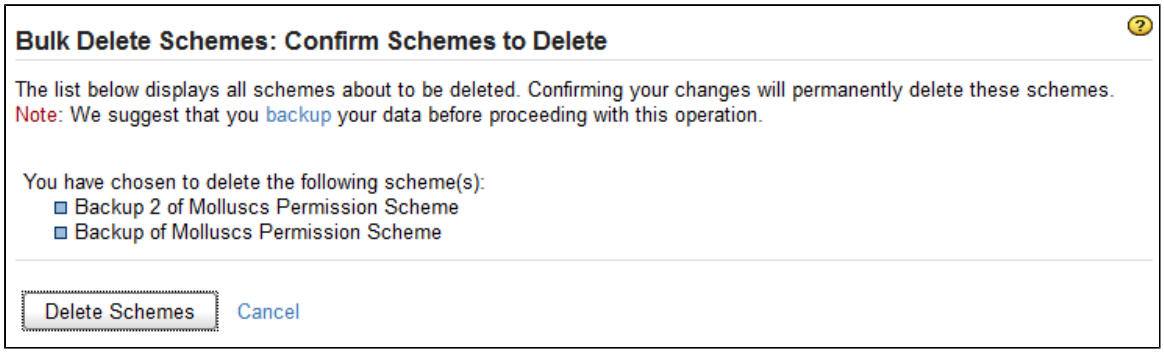
12. Click the '**Bulk Delete Schemes Tool**' link. (Note: this link is also available from the 'Scheme Tools' page shown above).
13. You will now see the '**Bulk Delete Schemes: Select Schemes**' page:
Screenshot 13: 'Bulk Delete Schemes: Select Schemes'



If you decide it is appropriate to delete the displayed schemes:

- Check the box next to the scheme names.
- Type a name for the new scheme in the "New Scheme Name" box.
- Click the '**"Preview**' button. Note that deleting these schemes will not affect any projects, as this page only displays schemes that are not associated with projects.

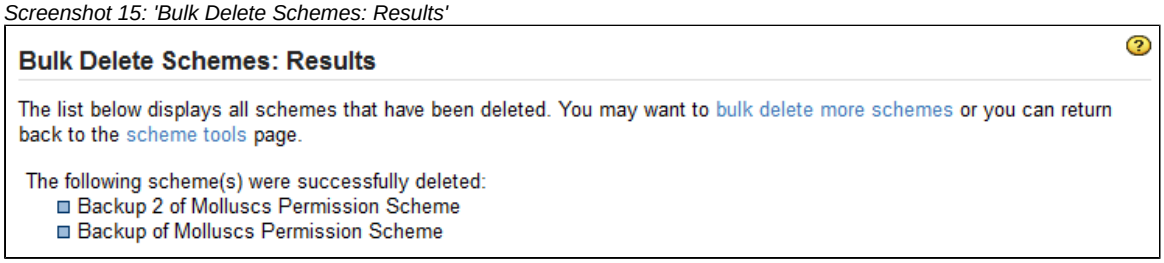
14. You will now see the 'Bulk Delete Schemes: Confirm Schemes to Delete' page:
Screenshot 14: 'Bulk Delete Schemes: Confirm Schemes to Delete'



If you are satisfied that the information shown on this page is correct, click the '**Delete Schemes**' button.

15. You will now see the 'Bulk Delete Schemes: Results' page, confirming that the unused schemes have been deleted:

Screenshot 15: 'Bulk Delete Schemes: Results'



Configuring User Directories

A user directory is a place where you store information about users and groups. User information includes the person's full name, username, password, email address and other personal information. Group information includes the name of the group, the users that belong to the group, and possibly groups that belong to other groups.

The **internal** directory stores user and group information in the JIRA database. You can also connect to **external** user directories, and to Atlassian **Crowd** and **JIRA** as directory managers.

On this page:

- Configuring User Directories in JIRA
- Connecting to a Directory
- Updating Directories

*The information on this page does not apply to **JIRA OnDemand**.*

Configuring User Directories in JIRA

To configure your user directories:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Users' > 'User Directories'**.
   Keyboard shortcut: **'g' + 'g' +** start typing **'directories'**.

Connecting to a Directory

You can add the following types of directory servers and directory managers:

- JIRA's internal directory. See Configuring the Internal Directory.
- Microsoft Active Directory. See Connecting to an LDAP Directory.
- Various other LDAP directory servers. See Connecting to an LDAP Directory.
- An LDAP directory for delegated authentication. See Connecting to an Internal Directory with LDAP Authentication.
- Atlassian Crowd. See Connecting to Crowd or Another JIRA Server for User Management.
- Another JIRA server. See Connecting to Crowd or Another JIRA Server for User Management.

You can add as many external user directories as you need. Note that you can define the **order** of the directories. This determines which directory JIRA will search first, when looking for user and group information. See Managing Multiple Directories.

Updating Directories

Limitations when Editing Directories

You cannot edit, disable or remove the directory your user belongs to. This precaution is designed to prevent administrators from locking themselves out of the application by changing the directory configuration in a way that prevents them logging in or removes their administration permissions.

This limitation applies to all directory types. For example:

- You cannot disable the internal directory if your user is an internal user.
- You cannot disable or remove an LDAP or a Crowd directory if your user comes from that directory.

In some situations, reordering the directories will change the directory that the current user comes from, if a user with the same username happens to exist in both. This behaviour can be used in some cases to create a copy of the existing configuration, move it to the top, then remove the old one. Note, however, that duplicate usernames are not a supported configuration.

You cannot remove the internal directory. This precaution aligns with the recommendation below that you always keep an administrator account active in the internal directory.

Recommendations

The recommended way to edit directory configurations is to log in as an internal user when making changes to external directory configuration.

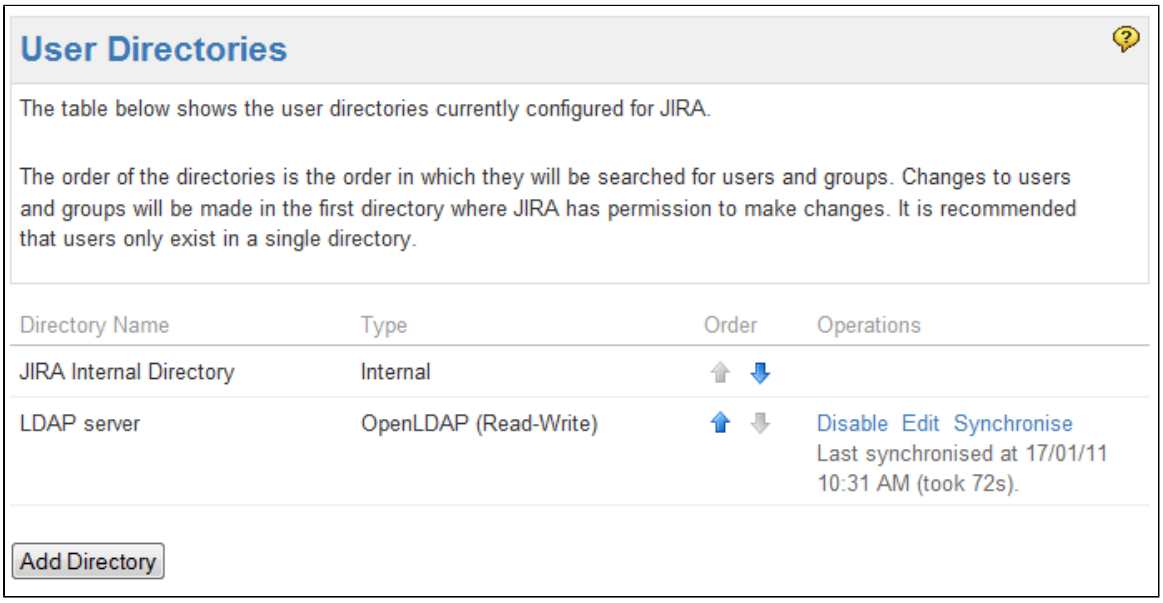
We recommend that you keep either an administrator or system administrator user active in your internal directory for troubleshooting problems with your user directories.

Enabling, Disabling and Removing Directories

You can enable or disable a directory at any time. If you disable a directory, your configuration details will remain but the application will not recognise the users and groups in that directory.

You have to disable a directory before you can remove it. Removing a directory will remove the details from the database.

Screenshot: Configuring user directories



RELATED TOPICS

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

- User and Group Management

Configuring the Internal Directory

The internal directory stores user and group information in the JIRA database.

The internal directory is enabled by default at installation. When you create the first administrator during the setup procedure, that administrator's username and other details are stored in the internal directory.

If needed, you can configure one or more additional user directories. This is useful if you want to grant access to users and groups that are stored in a corporate directory or other directory server.

On this page:

- Settings
- Diagram of Possible Configuration

*The information on this page does not apply to **JIRA OnDemand**.*

Settings

Setting	Description
Enable Nested Groups	Enable or disable support for nested groups. When nested groups are enabled, you can define a group as a member of another group. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups.

Diagram of Possible Configuration

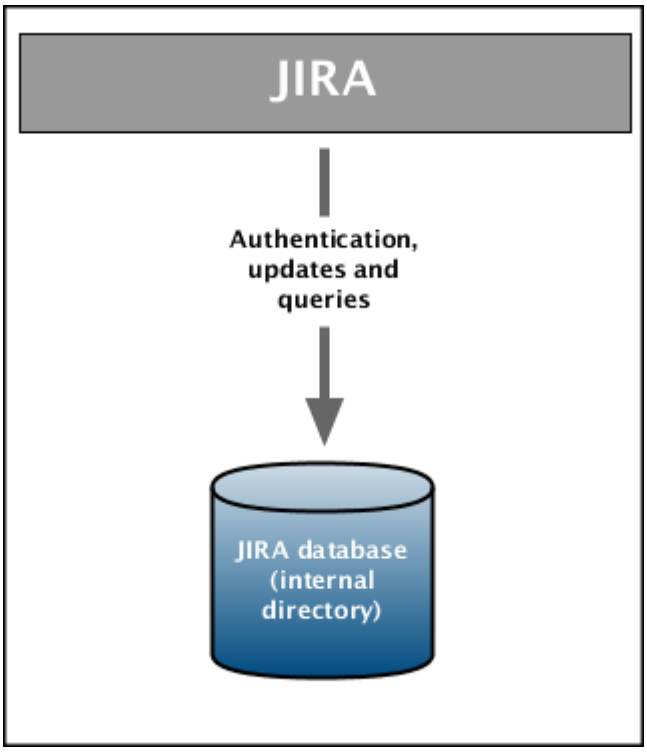


Diagram above: JIRA using its internal directory for user management.
RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Connecting to an LDAP Directory

You can connect your JIRA application to an LDAP directory for authentication, user and group management.

Overview

An LDAP directory is a collection of data about users and groups. LDAP (Lightweight Directory Access Protocol) is an Internet protocol that web applications can use to look up information about those users and groups from the LDAP server.

We provide built-in connectors for the most popular LDAP directory servers:

- Microsoft Active Directory
- Apache Directory Server (ApacheDS)
- Apple Open Directory
- Fedora Directory Server
- Novell eDirectory
- OpenDS
- OpenLDAP
- OpenLDAP Using Posix Schema
- Posix Schema for LDAP
- Sun Directory Server Enterprise Edition (DSEE)
- A generic LDAP directory server

When to use this option: Connecting to an LDAP directory server is useful if your users and groups are stored in a corporate directory. When configuring the directory, you can choose to make it read only, read only with local groups, or read/write. If you choose read/write, any changes made to user and group information in the application will also update the LDAP directory.

On this page:

- Overview
- Connecting to an LDAP Directory in JIRA
- Server Settings
- Schema Settings
- Permission Settings
  - Adding Users to Groups Automatically
- Advanced Settings
- User Schema Settings
- Group Schema Settings
- Membership Schema Settings
- Diagrams of Some Possible Configurations

*The information on this page does not apply to **JIRA OnDemand**.*

Connecting to an LDAP Directory in JIRA

To connect JIRA to an LDAP directory:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Users' > 'User Directories'**.
   Keyboard shortcut: **'g' + 'g' +** start typing **'directories'**.
3. **Add** a directory and select one of these types:
   - '**Microsoft Active Directory**' – This option provides a quick way to select AD, because it is the most popular LDAP directory type.
   - '**LDAP**' – You will be able to choose a specific LDAP directory type on the next screen.
4. Enter the values for the settings, as described below.
5. Save the directory settings.
6. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the '**User Directories**' screen. Here is a summary of how the directory order affects the processing:
   - The order of the directories is the order in which they will be searched for users and groups.
   - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

   For details see Managing Multiple Directories.

Server Settings

Setting	Description
Name	Enter a meaningful name to help you identify the LDAP directory server. Examples: • `Example Company Staff Directory` • `Example Company Corporate LDAP`
Directory Type	Select the type of LDAP directory that you will connect to. If you are adding a new LDAP connection, the value you select here will determine the default values for many of the options on the rest of screen. Examples: • `Microsoft Active Directory` • `OpenDS` • And more.
Hostname	The host name of your directory server. Examples: • `ad.example.com` • `ldap.example.com` • `opends.example.com`
Port	The port on which your directory server is listening. Examples: • `389` • `10389` • `636` (for example, for SSL)

Use SSL	Tick this check box if the connection to the directory server is an SSL (Secure Sockets Layer) connection. Note that you will need to configure an SSL certificate in order to use this setting.
Username	The distinguished name of the user that the application will use when connecting to the directory server. Examples: • `cn=administrator,cn=users,dc=ad,dc=example,dc=com` • `cn=user,dc=domain,dc=name` • `user@domain.name`
Password	The password of the user specified above.

Schema Settings

Setting	Description
Base DN	The root distinguished name (DN) to use when running queries against the directory server. Examples: • `o=example,c=com` • `cn=users,dc=ad,dc=example,dc=com` • For Microsoft Active Directory, specify the base DN in the following format: `dc=domain1,dc=local`. You will need to replace the `domain1` and `local` for your specific configuration. Microsoft Server provides a tool called `ldp.exe` which is useful for finding out and configuring the the LDAP structure of your server.
Additional User DN	This value is used in addition to the base DN when searching and loading users. If no value is supplied, the subtree search will start from the base DN. Example: • `ou=Users`
Additional Group DN	This value is used in addition to the base DN when searching and loading groups. If no value is supplied, the subtree search will start from the base DN. Example: • `ou=Groups`

Permission Settings

Setting	Description
Read Only	LDAP users, groups and memberships are retrieved from your directory server and can only be modified via your directory server. You cannot modify LDAP users, groups or memberships via the application administration screens.
Read Only, with Local Groups	LDAP users, groups and memberships are retrieved from your directory server and can only be modified via your directory server. You cannot modify LDAP users, groups or memberships via the application administration screens. However, you can add groups to the internal directory and add LDAP users to those groups.
Read/Write	LDAP users, groups and memberships are retrieved from your directory server. When you modify a user, group or membership via the application administration screens, the changes will be applied directly to your LDAP directory server. Please ensure that the LDAP user specified for the application has modification permissions on your LDAP directory server.

Adding Users to Groups Automatically

Setting	Description

Default Group Memberships	*Option available in Confluence 3.5 and later, and JIRA 4.3.3 and later.* This field appears if you select the 'Read Only, with Local Groups' permission. If you would like users to be automatically added to a group or groups, enter the group name(s) here. To specify more than one group, separate the group names with commas. *In Confluence 3.5 to Confluence 3.5.1:* Each time a user logs in, their group memberships will be checked. If the user does not belong to the specified group(s), their username will be added to the group(s). If a group does not yet exist, it will be added locally. *In Confluence 3.5.2 and later, and JIRA 4.3.3 and later:* The first time a user logs in, their group memberships will be checked. If the user does not belong to the specified group(s), their username will be added to the group(s). If a group does not yet exist, it will be added locally. On subsequent logins, the username will *not* be added automatically to any groups. This change in behaviour allows users to be removed from automatically-added groups. In Confluence 3.5 and 3.5.1, they would be re-added upon next login. Please note that there is no validation of the group names. If you mis-type the group name, authorisation failures will result – users will not be able to access the applications or functionality based on the intended group name. Examples: • `confluence-users` • `confluence-users,jira-users,jira-developers`

Advanced Settings

Setting	Description
Enable Nested Groups	Enable or disable support for nested groups. Some directory servers allow you to define a group as a member of another group. Groups in such a structure are called 'nested groups'. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups.
Use Paged Results	Enable or disable the use of the LDAP control extension for simple paging of search results. If paging is enabled, the search will retrieve sets of data rather than all of the search results at once. Enter the desired page size – that is, the maximum number of search results to be returned per page when paged results are enabled. The default is 1000 results.
Follow Referrals	Choose whether to allow the directory server to redirect requests to other servers. This option uses the node referral (JNDI lookup `java.naming.referral`) configuration setting. It is generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Naive DN Matching	If your directory server will always return a consistent string representation of a DN, you can enable naive DN matching. Using naive DN matching will result in a significant performance improvement, so we recommend enabling it where possible. This setting determines how your application will compare DNs to determine if they are equal. • If this check box is ticked, the application will do a direct, case-insensitive, string comparison. This is the default and recommended setting for Active Directory, because Active Directory guarantees the format of DNs. • If this check box is not ticked, the application will parse the DN and then check the parsed version.
Enable Incremental Synchronisation	Enable incremental synchronisation if you only want changes since the last synchronisation to be queried when synchronising a directory. ⚠ Please be aware that when using this option, the user account configured for synchronisation must have read access to: • The `uSNChanged` attribute of all users and groups in the directory that need to be synchronised. • The objects and attributes in the Active Directory deleted objects container (see Microsoft's Knowledge Base Article No. 892806 for details). If at least one of these conditions is not met, you may end up with users who are added to (or deleted from) the Active Directory not being respectively added (or deleted) in JIRA.
Synchronisation Interval (minutes)	Synchronisation is the process by which the application updates its internal store of user data to agree with the data on the directory server. The application will send a request to your directory server every x minutes, where 'x' is the number specified here. The default value is 60 minutes.
Read Timeout (seconds)	The time, in seconds, to wait for a response to be received. If there is no response within the specified time period, the read attempt will be aborted. A value of 0 (zero) means there is no limit. The default value is 120 seconds.

Search Timeout (seconds)	The time, in seconds, to wait for a response from a search operation. A value of 0 (zero) means there is no limit. The default value is 60 seconds.
Connection Timeout (seconds)	This setting affects two actions. The default value is 0. • The time to wait when getting a connection from the connection pool. A value of 0 (zero) means there is no limit, so wait indefinitely. • The time, in seconds, to wait when opening new server connections. A value of 0 (zero) means that the TCP network timeout will be used, which may be several minutes.

User Schema Settings

Setting	Description
User Object Class	This is the name of the class used for the LDAP user object. Example: • `user`
User Object Filter	The filter to use when searching user objects. Example: • `(&(objectCategory=Person)(sAMAccountName=*))`
User Name Attribute	The attribute field to use when loading the username. Examples: • `cn` • `sAMAccountName`
User Name RDN Attribute	The RDN (relative distinguished name) to use when loading the username. The DN for each LDAP entry is composed of two parts: the RDN and the location within the LDAP directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure. Example: • `cn`
User First Name Attribute	The attribute field to use when loading the user's first name. Example: • `givenName`
User Last Name Attribute	The attribute field to use when loading the user's last name. Example: • `sn`
User Display Name Attribute	The attribute field to use when loading the user's full name. Example: • `displayName`
User Email Attribute	The attribute field to use when loading the user's email address. Example: • `mail`
User Password Attribute	The attribute field to use when loading a user's password. Example: • `unicodePwd`

Group Schema Settings

Setting	Description
Group Object Class	This is the name of the class used for the LDAP group object. Examples: • `groupOfUniqueNames` • `group`

Group Object Filter	The filter to use when searching group objects. Example: • `(objectCategory=Group)`
Group Name Attribute	The attribute field to use when loading the group's name. Example: • `cn`
Group Description Attribute	The attribute field to use when loading the group's description. Example: • `description`

Membership Schema Settings

Setting	Description
Group Members Attribute	The attribute field to use when loading the group's members. Example: • `member`
User Membership Attribute	The attribute field to use when loading the user's groups. Example: • `memberOf`
Use the User Membership Attribute, when finding the user's group membership	Put a tick in the checkbox if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) • If this checkbox is ticked, your application will use the group membership attribute on the user when **retrieving the members of a given group**. This will result in a more efficient retrieval. • If this checkbox is not ticked, your application will use the members attribute on the group ('member' by default) for the search. • If the 'Enable Nested Groups' checkbox is ticked, your application will ignore the 'Use memberOf Attribute on the User' option and will use the members attribute on the group for the search.
Use the User Membership Attribute, when finding the members of a group	Put a tick in the checkbox if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) • If this checkbox is ticked, your application will use the group membership attribute on the user when **retrieving the list of groups to which a given user belongs**. This will result in a more efficient search. • If this checkbox is not ticked, your application will use the members attribute on the group ('member' by default) for the search.

Diagrams of Some Possible Configurations

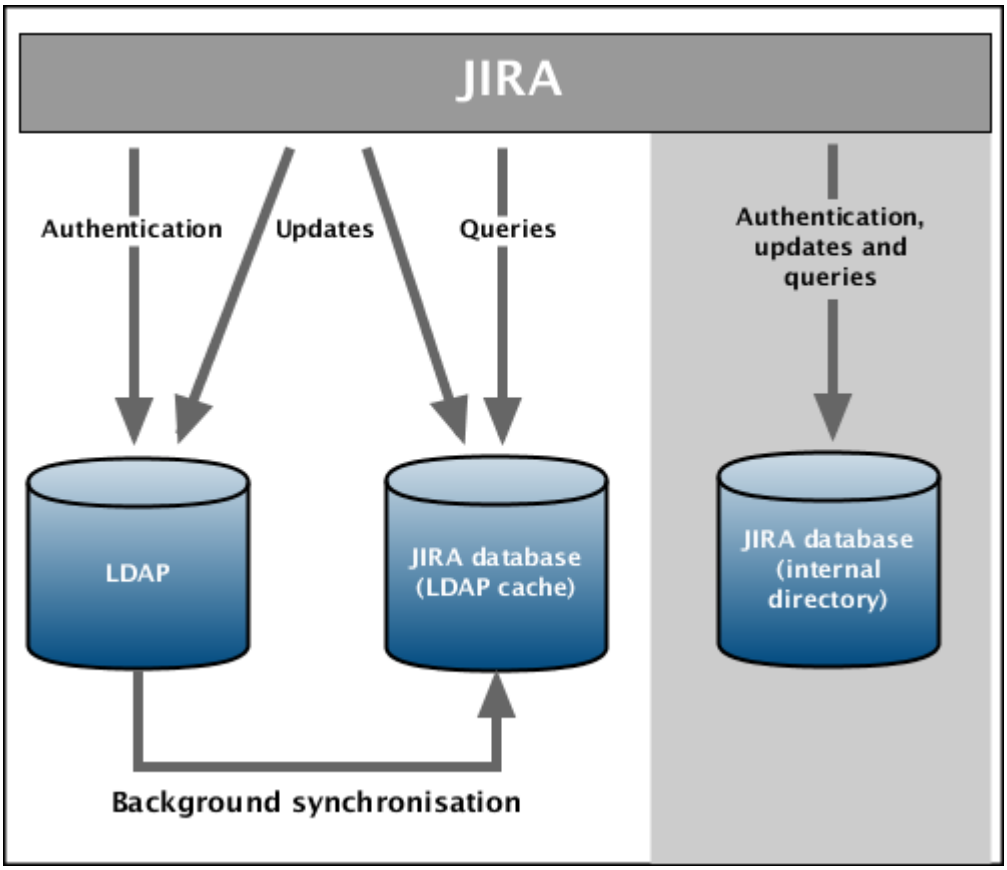


Diagram above: JIRA connecting to an LDAP directory.

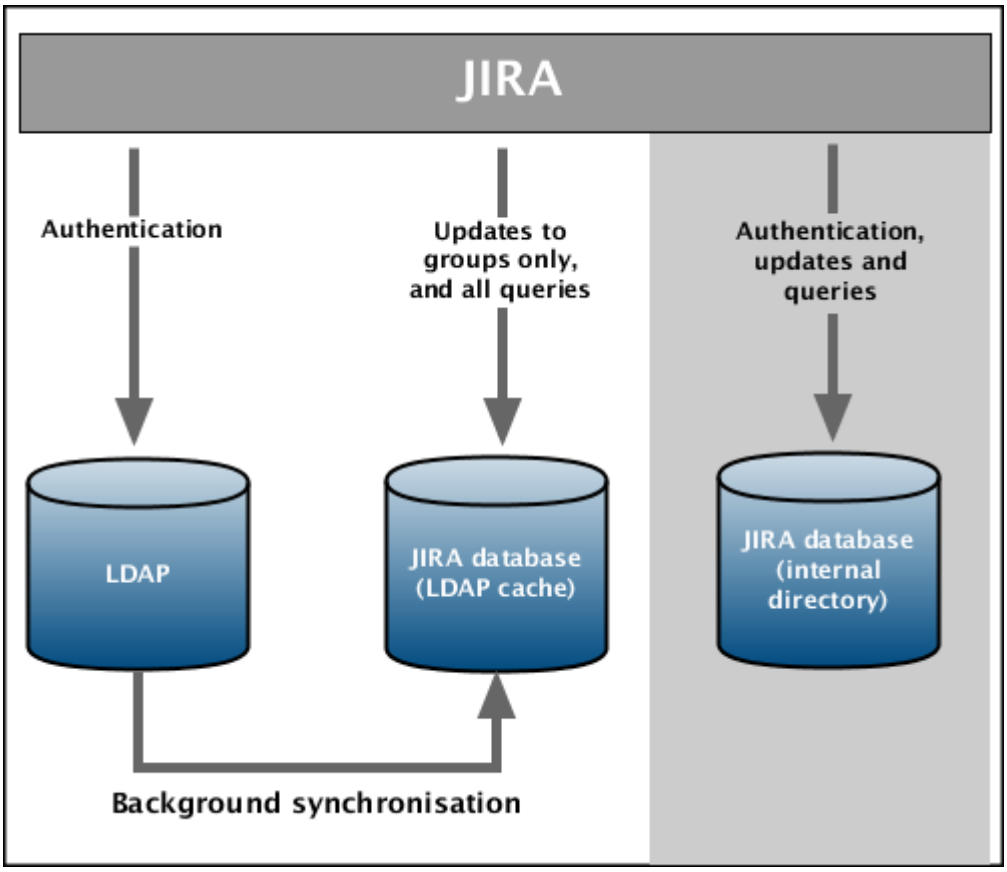


Diagram above: JIRA connecting to an LDAP directory with permissions set to read only and local groups.

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory

- Connecting to an LDAP Directory
  - Configuring an SSL Connection to Active Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Configuring an SSL Connection to Active Directory

If you want to configure a read/write connection with Microsoft Active Directory, you will need to install an SSL certificate, generated by your Active Directory server, onto your JIRA server and then install the certificate into your JVM keystore.

On this page:

- Prerequisites
- Step 1. Install the Active Directory Certificate Services
- Step 2. Obtain the Server Certificate
- Step 3. Import the Server Certificate

*The information on this page does not apply to **JIRA OnDemand**.*

Updating user, group, and membership details in Active Directory requires that your Atlassian application be running in a JVM that trusts the AD server. To do this, we generate a certificate on the Active Directory server, then import it into Java's `keystore`.

Prerequisites

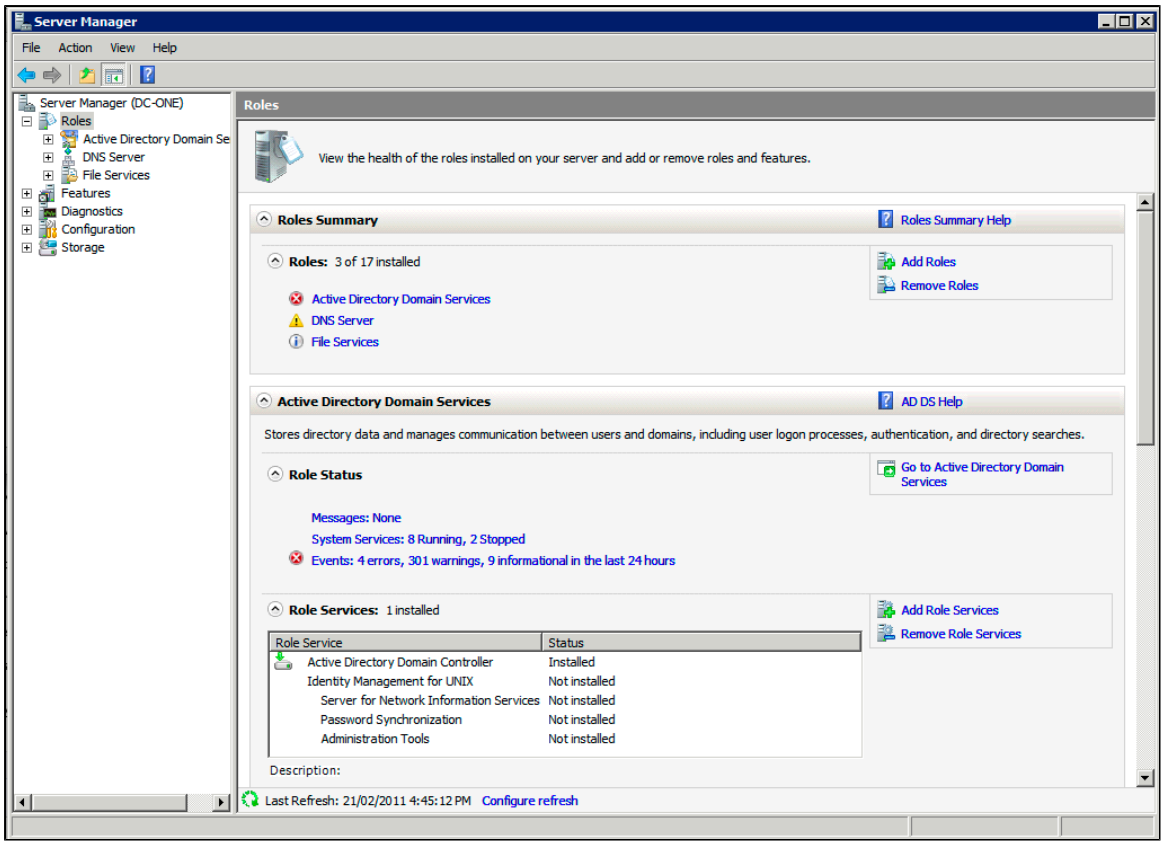
To generate a certificate, you need the following components installed on the Windows Domain Controller to which you're connecting.

Required Component	Description
Internet Information Services (IIS)	This is required before you can install Windows Certificate Services.
Windows Certificate Services	This installs a certification authority (CA) which is used to issue certificates. Step 1, below, explains this process.
Windows 2000 Service Pack 2	Required if you are using Windows 2000
Windows 2000 High Encryption Pack (128-bit)	Required if you are using Windows 2000. Provides the highest available encryption level (128-bit).

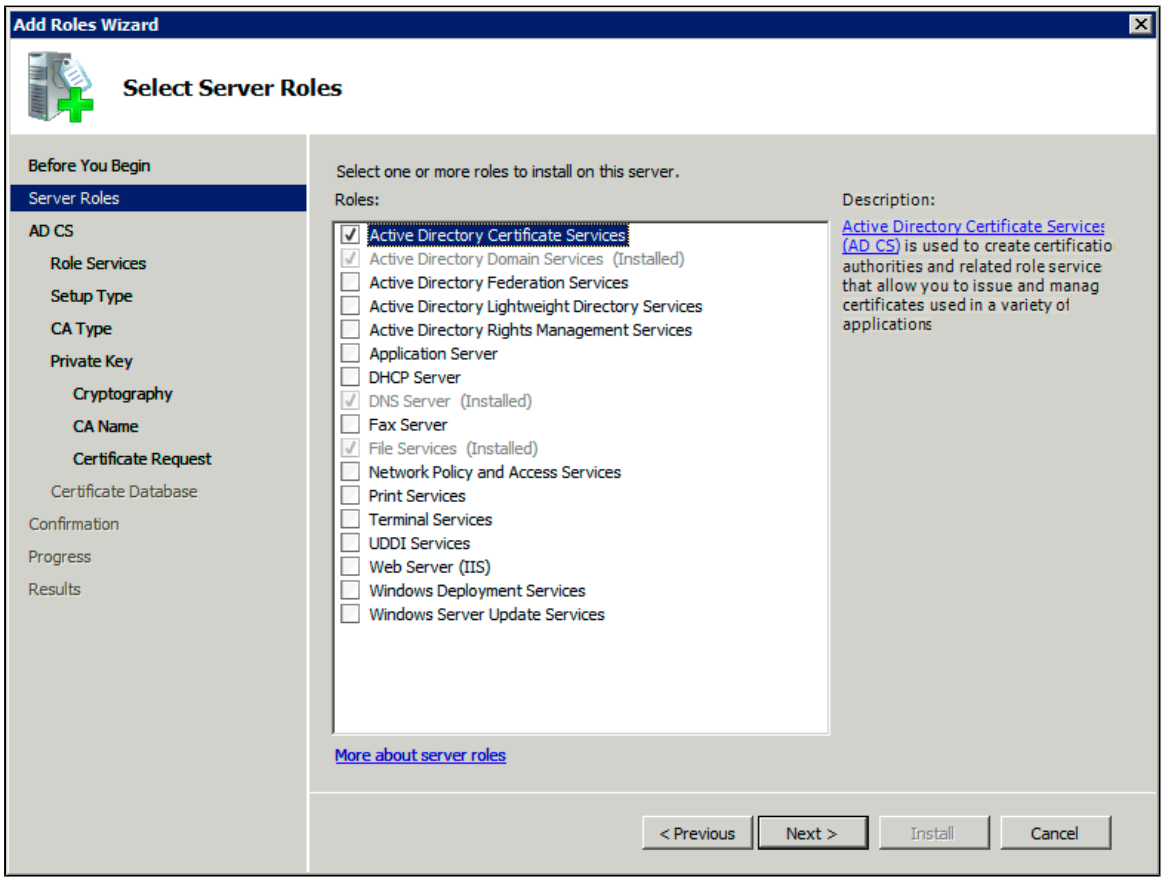
Step 1. Install the Active Directory Certificate Services

If Certificate Services are already installed, skip to step 2, below. The screenshots below are from Server 2008, but the process is similar for Server 2000 and 2003.

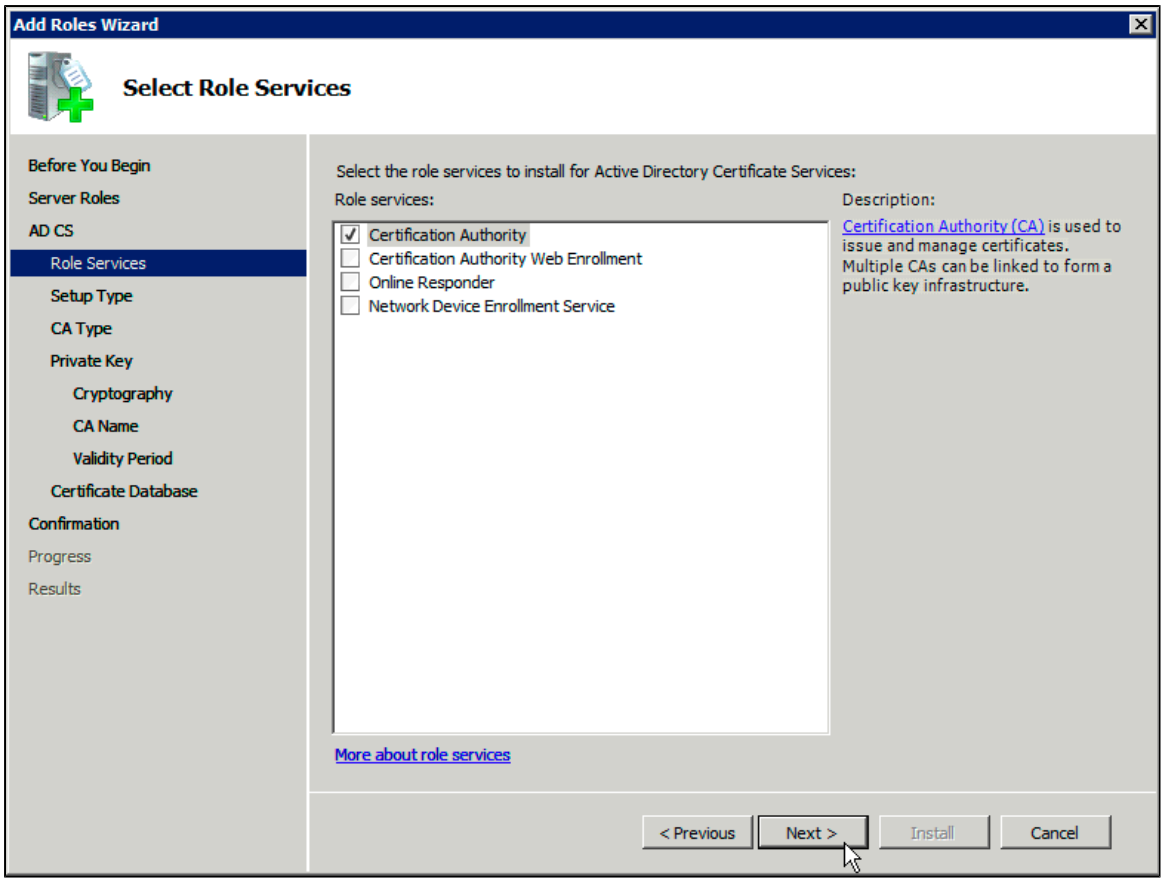
1. Log in to your Active Directory server as an administrator.
2. Click **Start**, point to **Administrative Tools**, and then click **Server Manager**.
3. In the **Roles Summary** section, click **Add Roles**.



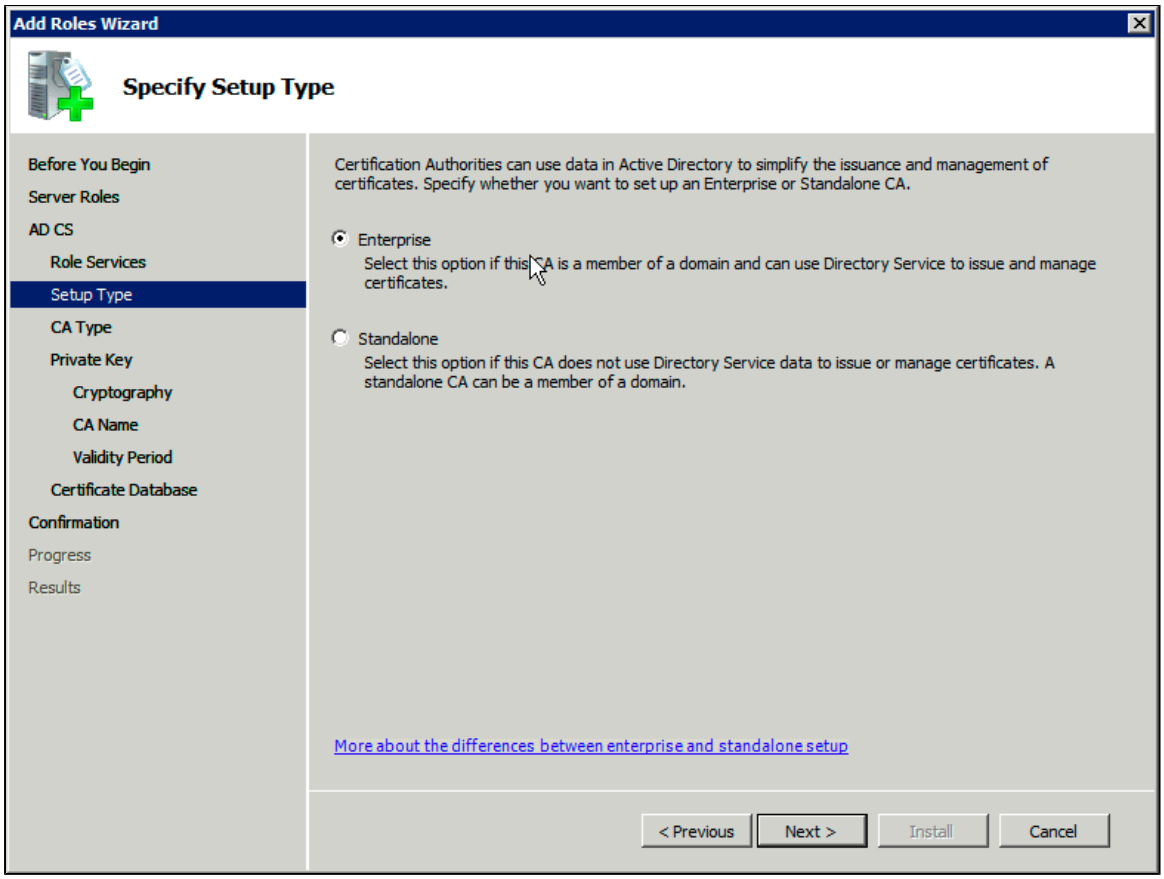
4. On the **Select Server Roles** page, select the **Active Directory Certificate Services** check box. Click **Next** twice.



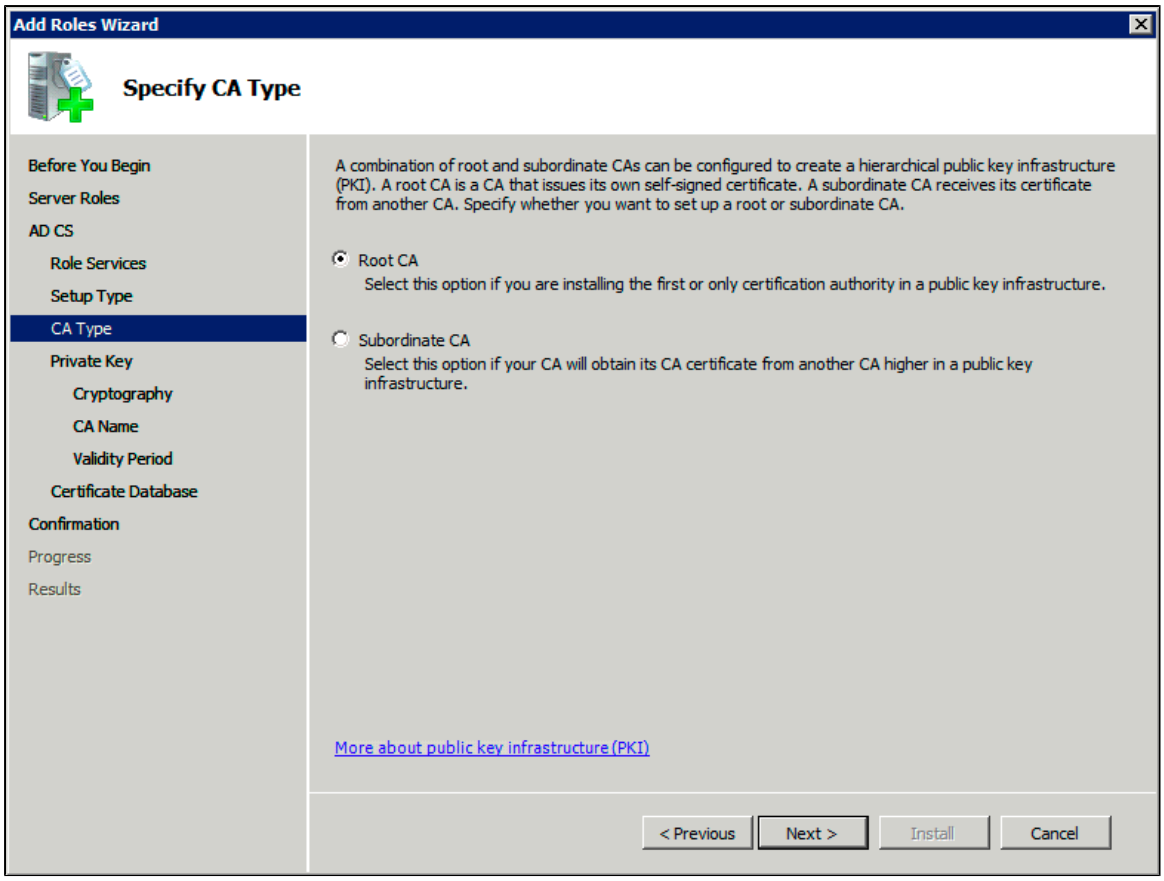
5. On the **Select Role Services** page, select the **Certification Authority** check box, and then click **Next**.



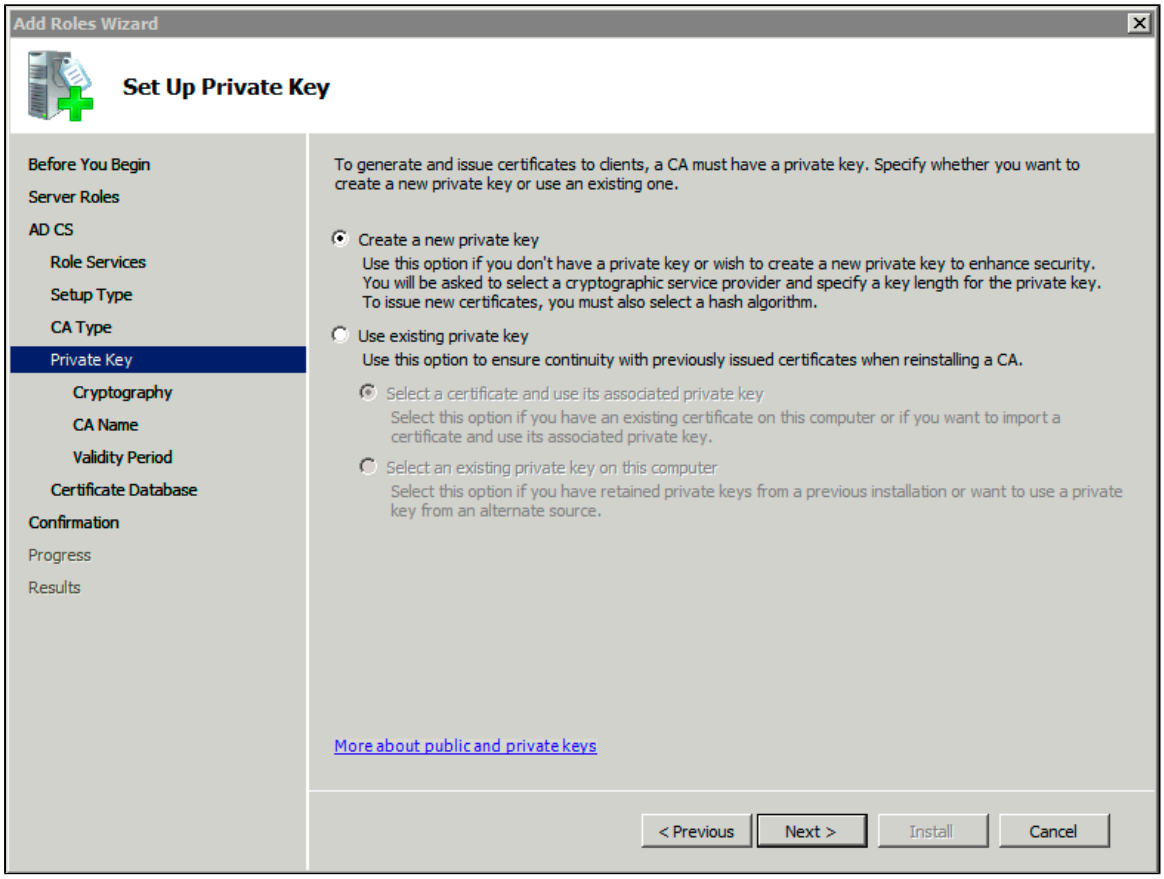
6. On the **Specify Setup Type** page, click **Enterprise**, and then click **Next**.



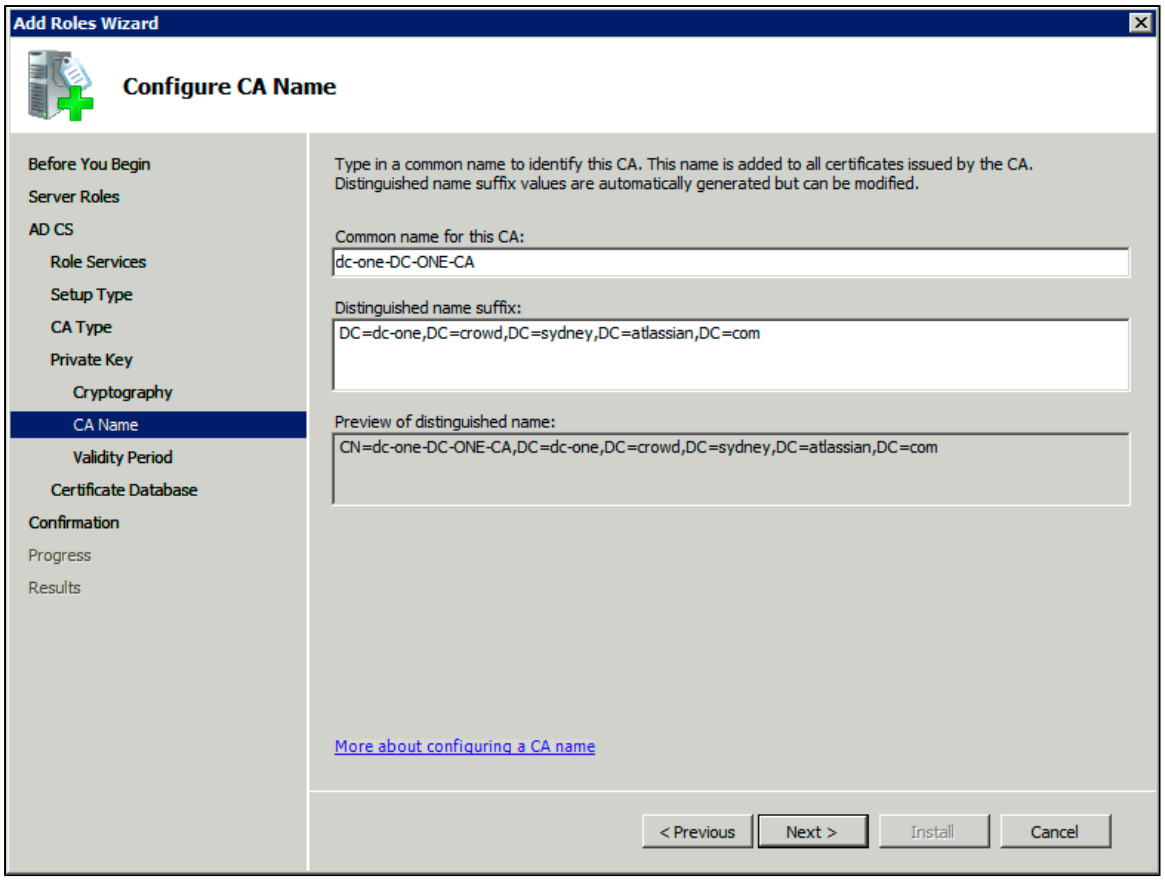
7. On the **Specify CA Type** page, click **Root CA**, and then click **Next**.



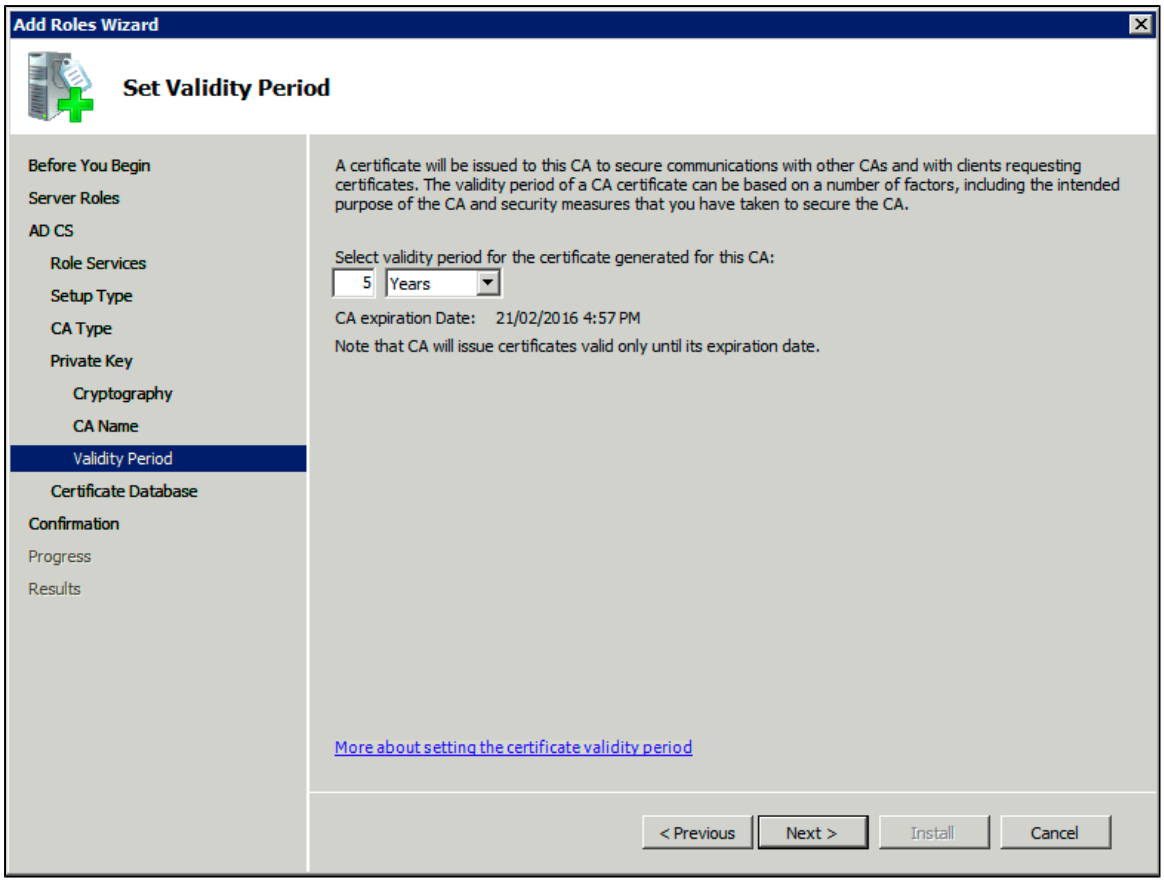
8. On the **Set Up Private Key** and **Configure Cryptography for CA** pages, you can configure optional configuration settings, including cryptographic service providers. However, the default values should be fine. Click **Next** twice.

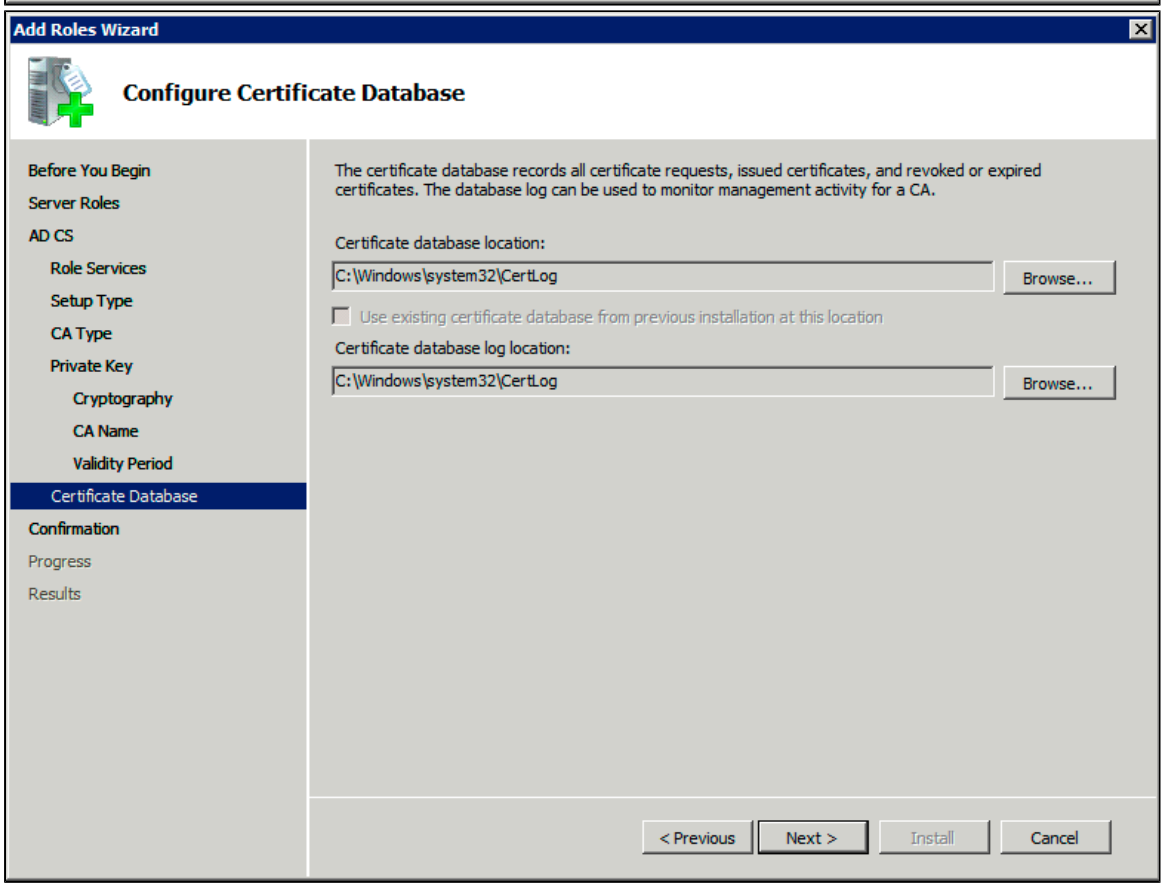


9. In the **Common name for this CA** box, type the common name of the CA, and then click **Next**.

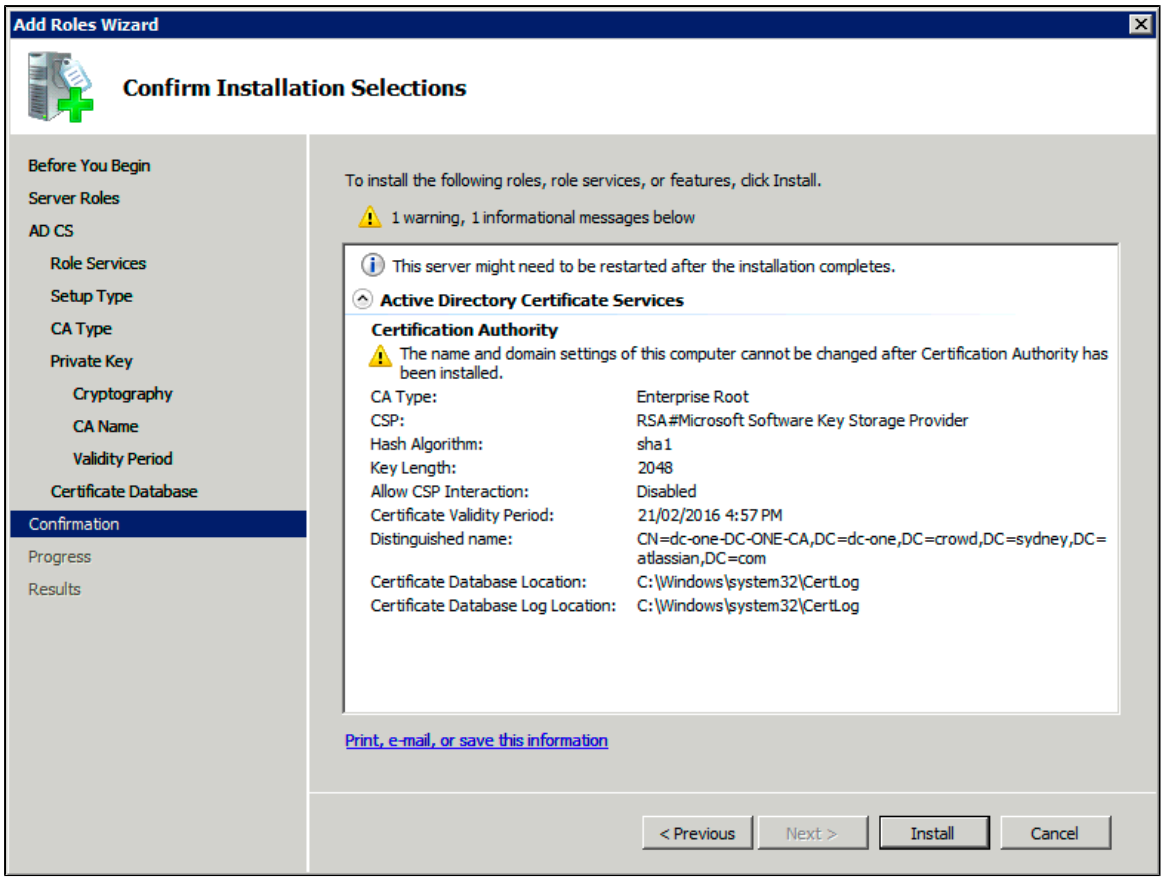


10. On the **Set Validity Period** page, accept the default values or specify other storage locations for the certificate database and the certificate database log, and then click **Next**.

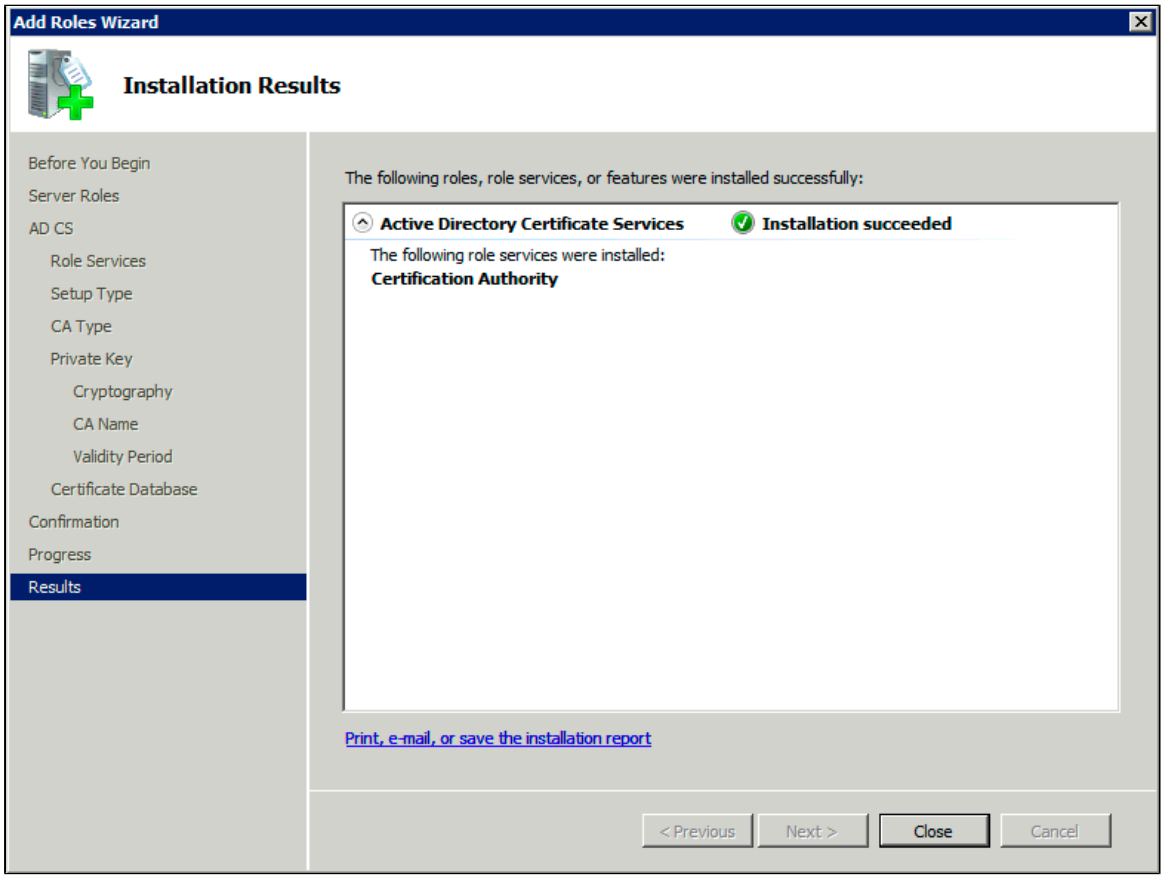




11. After verifying the information on the **Confirm Installation Selections** page, click **Install**.



12. Review the information on the results screen to verify that the installation was successful.



Step 2. Obtain the Server Certificate

The steps above describe how to install the certification authority (CA) on your Microsoft Active Directory server. Next, you will need to add the Microsoft Active Directory server's SSL certificate to the list of accepted certificates used by the JDK that runs your application server.

The Active Directory certificate is automatically generated and placed in root of the C:\ drive, matching a file format similar to the tree structure of your Active Directory server. For example: `c:\ad2008.ad01.atlassian.com_ad01.crt`.

You can also export the certificate by executing this command on the Active Directory server:

```
certutil -ca.cert client.crt
```

Step 3. Import the Server Certificate

For an application server to trust your directory's certificate, the certificate must be imported into your Java runtime environment. The JDK stores trusted certificates in a file called a keystore. The default keystore file is called `cacerts` and it lives in the `jre\lib\security` sub-directory of your Java installation.

In the following examples, we use `server-certificate.crt` to represent the certificate file exported by your directory server. You will need to alter the instructions below to match the name actually generated.

Windows

1. Navigate to the directory in which Java is installed. It's probably called something like `C:\Program Files\Java\jdk1.5.0_12`.
2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

```
keytool -import -keystore .\jre\lib\security\cacerts -file server-certificate.crt
```

3. `keytool` will prompt you for a password. The default keystore password is `changeit`.
4. When prompted `Trust this certificate? [no]:` enter `yes` to confirm the key import:

```
Enter keystore password:  changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 15563d6677a4e9e4582d8a84be683f9
Valid from: Tue Aug 21 01:10:46 ACT 2007 until: Tue Aug 21 01:13:59 ACT 2012
Certificate fingerprints:
         MD5:  D6:56:F0:23:16:E3:62:2C:6F:8A:0A:37:30:A1:84:BE
         SHA1: 73:73:4E:A6:A0:D1:4E:F4:F3:CD:CE:BE:96:80:35:D2:B4:7C:79:C1
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

You may now use the '**Secure SSL**' option when connecting your application to your directory server.

UNIX

1. Navigate to the directory in which Java is installed. `cd $JAVA_HOME` will usually get you there.
2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

```
sudo keytool -import -keystore ./jre/lib/security/cacerts -file server-certificate.crt
```

3. `keytool` will prompt you for a password. The default keystore password is `changeit`.
4. When prompted `Trust this certificate? [no]:` enter `yes` to confirm the key import:

```
Password:
Enter keystore password:  changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 15563d6677a4e9e4582d8a84be683f9
Valid from: Tue Aug 21 01:10:46 ACT 2007 until: Tue Aug 21 01:13:59 ACT 2012
Certificate fingerprints:
         MD5:  D6:56:F0:23:16:E3:62:2C:6F:8A:0A:37:30:A1:84:BE
         SHA1: 73:73:4E:A6:A0:D1:4E:F4:F3:CD:CE:BE:96:80:35:D2:B4:7C:79:C1
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

You may now use the '**Secure SSL**' option when connecting your application to your directory server.

Mac OS X

1. Navigate to the directory in which Java is installed. This is usually `/Library/Java/Home`.
2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

```
sudo keytool -import -keystore ./jre/lib/security/cacerts -file server-certificate.crt
```

3. `keytool` will prompt you for a password. The default keystore password is `changeit`.
4. When prompted `Trust this certificate? [no]:` enter `yes` to confirm the key import:

```
Password:
Enter keystore password:  changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 15563d6677a4e9e4582d8a84be683f9
Valid from: Tue Aug 21 01:10:46 ACT 2007 until: Tue Aug 21 01:13:59 ACT 2012
Certificate fingerprints:
         MD5:  D6:56:F0:23:16:E3:62:2C:6F:8A:0A:37:30:A1:84:BE
         SHA1: 73:73:4E:A6:A0:D1:4E:F4:F3:CD:CE:BE:96:80:35:D2:B4:7C:79:C1
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

You may now use the '**Secure SSL**' option when connecting your application to your directory server.
RELATED TOPICS

Connecting to an LDAP Directory
Configuring User Directories

Connecting to an Internal Directory with LDAP Authentication

You can connect your JIRA application to an LDAP directory for delegated authentication. This means that JIRA will have an internal directory that uses LDAP for authentication only. There is an option to create users in the internal directory automatically when they attempt to log in, as described in the settings section.

Overview

An internal directory with LDAP authentication offers the features of an internal directory while allowing you to store and check users' passwords in LDAP only. Note that the 'internal directory with LDAP authentication' is separate from the default 'internal directory'. On LDAP, all that the application does is to check the password. The LDAP connection is read only. Every user in the internal directory with LDAP authentication must map to a user on LDAP, otherwise they cannot log in.

When to use this option: Choose this option if you want to set up a user and group configuration within your application that suits your needs, while checking your users' passwords against the corporate LDAP directory. This option also helps to avoid the performance issues that may result from downloading large numbers of groups from LDAP.

On this page:

- Overview
- Connecting JIRA to an Internal Directory with LDAP Authentication
- Server Settings
  - Copying Users on First Login
- Schema Settings
- User Schema Settings (Used when Copying Users on First Login)
- Diagrams of Possible Configurations

*The information on this page does not apply to **JIRA OnDemand**.*

Connecting JIRA to an Internal Directory with LDAP Authentication

To connect to an internal directory but check logins via LDAP:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Users' > 'User Directories'**.
   Keyboard shortcut: **'g' + 'g' +** start typing **'directories'**.
3. **Add** a directory and select type '**Internal with LDAP Authentication**'.
4. Enter the values for the settings, as described below.
5. Save the directory settings.
6. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the '**User Directories**' screen. We

recommend that the 'Internal Directory with LDAP Authentication' is at the top of the list. Here is a summary of how the directory order affects the processing:
- The order of the directories is the order in which they will be searched for users and groups.
- Changes to users and groups will be made only in the first directory where the application has permission to make changes.

For details see Managing Multiple Directories.

7. Add your users and groups in JIRA. See Managing Users and Managing Groups.

Server Settings

Note: The option to **select a directory type** is available only in JIRA 4.3.3 and later.

Setting	Description
Name	A descriptive name that will help you to identify the directory. Examples: • `Internal directory with LDAP Authentication` • `Corporate LDAP for Authentication Only`
Directory Type	Select the type of LDAP directory that you will connect to. If you are adding a new LDAP connection, the value you select here will determine the default values for some of the options on the rest of screen. Examples: • `Microsoft Active Directory` • `OpenDS` • And more.
Hostname	The host name of your directory server. Examples: • `ad.example.com` • `ldap.example.com` • `opends.example.com`
Port	The port on which your directory server is listening. Examples: • `389` • `10389` • `636` (for example, for SSL)
Use SSL	Tick this check box if the connection to the directory server is an SSL (Secure Sockets Layer) connection. Note that you will need to configure an SSL certificate in order to use this setting.
Username	The distinguished name of the user that the application will use when connecting to the directory server. Examples: • `cn=administrator,cn=users,dc=ad,dc=example,dc=com` • `cn=user,dc=domain,dc=name` • `user@domain.name`
Password	The password of the user specified above.

Copying Users on First Login

Note: The option to **copy users on first login** is available only in JIRA 4.3.3 and later.

Setting	Description
Copy User on First Login	This option affects what will happen when a user attempts to log in, if their username does not yet exist in the internal directory that is using LDAP for authentication. If this check box is ticked, the user will be created automatically in the internal directory when the user logs in. If this check box is not ticked, the user's login will fail. If you tick this check box the following additional fields will appear on the screen, both described in more detail below: • Default Group Memberships • User Schema Settings

Default Group Memberships	This field appears if you tick the 'Copy User on First Login' check box. If you would like users to be automatically added to a group or groups, enter the group name(s) here. To specify more than one group, separate the group names with commas. Each time a user logs in, their group memberships will be checked. If the user does not belong to the specified group(s), their username will be added to the group(s). If a group does not yet exist, it will be added to the internal directory that is using LDAP for authentication. Please note that there is no validation of the group names. If you mis-type the group name, authorisation failures will result – users will not be able to access the applications or functionality based on the intended group name. Examples: • `confluence-users` • `confluence-users,jira-users,jira-developers`

Schema Settings

Setting	Description
Base DN	The root distinguished name (DN) to use when running queries against the directory server. Examples: • `o=example,c=com` • `cn=users,dc=ad,dc=example,dc=com` • For Microsoft Active Directory, specify the base DN in the following format: `dc=domain1,dc=local`. You will need to replace the `domain1` and `local` for your specific configuration. Microsoft Server provides a tool called `ldp.exe` which is useful for finding out and configuring the the LDAP structure of your server.
User Name Attribute	The attribute field to use when loading the username. Examples: • `cn` • `sAMAccountName`

User Schema Settings (Used when Copying Users on First Login)

Note: The **user schema settings** are available only in JIRA 4.3.3 and later.

Setting	Description
User Schema Settings	This section appears if you tick the 'Copy User on First Login' check box. If the fields below this heading are hidden, click the heading to reveal the fields.
Additional User DN	This value is used in addition to the base DN when searching and loading users. If no value is supplied, the subtree search will start from the base DN. Example: • `ou=Users`
User Object Class	This is the name of the class used for the LDAP user object. Example: • `user`
User Object Filter	The filter to use when searching user objects. Example: • `(&(objectCategory=Person)(sAMAccountName=*))`
User Name RDN Attribute	The RDN (relative distinguished name) to use when loading the username. The DN for each LDAP entry is composed of two parts: the RDN and the location within the LDAP directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure. Example: • `cn`
User First Name Attribute	The attribute field to use when loading the user's first name. Example: • `givenName`

User Last Name Attribute	The attribute field to use when loading the user's last name. Example: • `sn`
User Display Name Attribute	The attribute field to use when loading the user's full name. Example: • `displayName`
User Email Attribute	The attribute field to use when loading the user's email address. Example: • `mail`

Diagrams of Possible Configurations

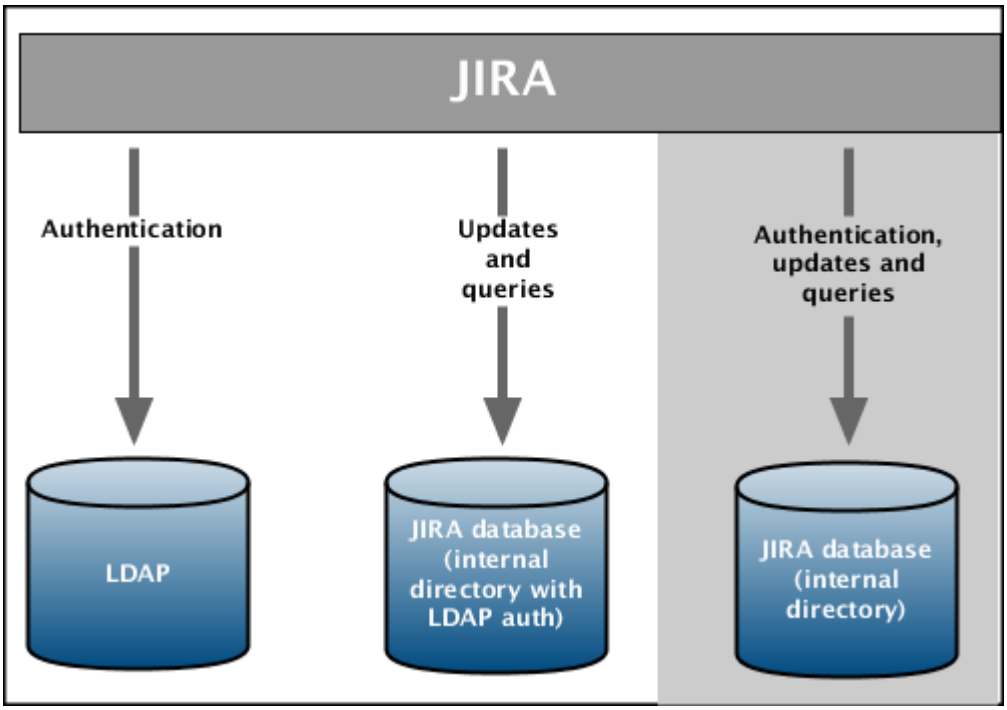


Diagram above: JIRA connecting to an LDAP directory for authentication only.

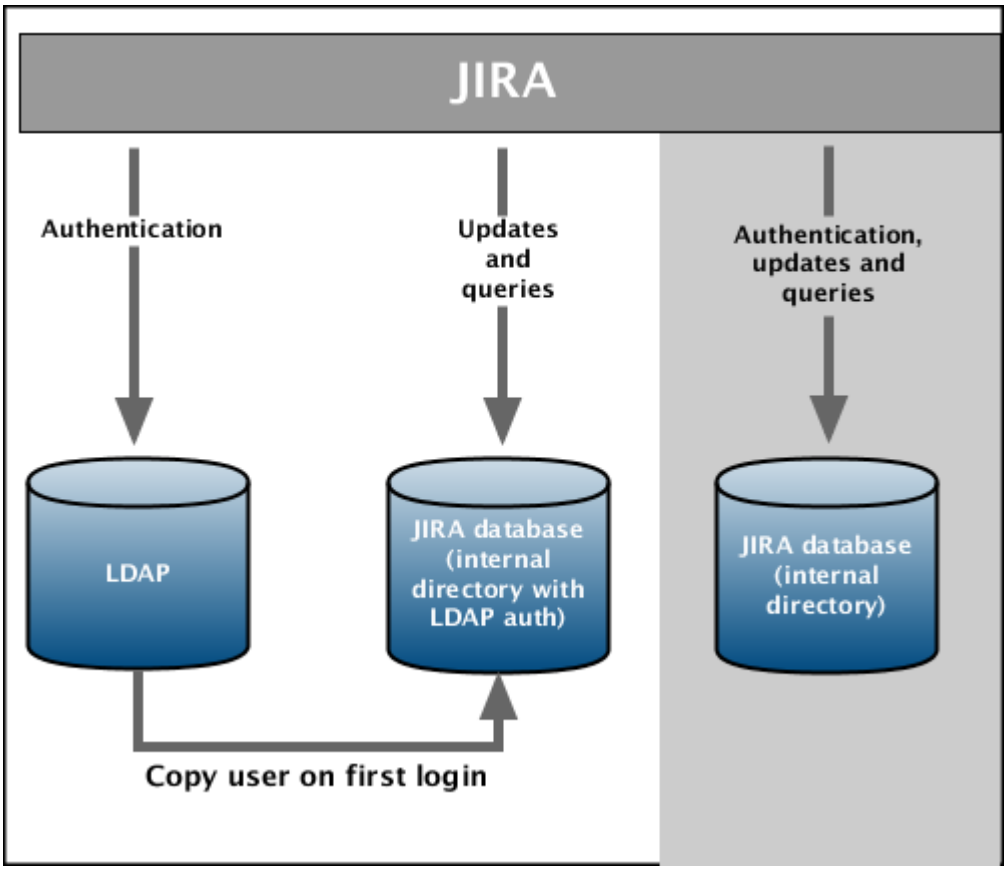


Diagram above: JIRA connecting to an LDAP directory for authentication only, with each user copied to the internal directory when they first log in to JIRA.

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Connecting to Crowd or Another JIRA Server for User Management

You can connect your JIRA application to Atlassian Crowd or to another JIRA server (version 4.3 or later) for management of users and groups, and for authentication (verification of a user's login).

On this page:

- Connecting JIRA to Crowd
- Connecting JIRA to Another JIRA Server
- Diagrams of Some Possible Configurations

*The information on this page does not apply to **JIRA OnDemand**.*

Connecting JIRA to Crowd

Atlassian Crowd is an application security framework that handles authentication and authorisation for your web-based applications. With Crowd you can integrate multiple web applications and user directories, with support for single sign-on (SSO) and centralised identity management. The Crowd Administration Console provides a web interface for managing directories, users and their permissions. See the Crowd Administration Guide.

When to use this option: Connect to Crowd if you want to use the full Crowd functionality to manage your directories, users and groups. You can connect your Crowd server to a number of directories of all types that Crowd supports, including custom directory connectors.

To connect JIRA to Crowd:

1. Go to your **Crowd Administration Console** and define the JIRA application to Crowd. See the Crowd documentation: Adding an Application.
2. Log in to JIRA as a user with the **'JIRA Administrators'** global permission.
3. Select **'Administration' > 'Users' > 'User Directories'**.
   Keyboard shortcut: **'g' + 'g' +** start typing **'directories'**.
4. **Add** a directory and select type '**Atlassian Crowd**'. Enter the settings as described below.
5. Save the directory settings.
6. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the '**User Directories**' screen. Here is a summary of how the directory order affects the processing:
   - The order of the directories is the order in which they will be searched for users and groups.
   - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

   For details see Managing Multiple Directories.
7. If required, configure JIRA to use Crowd for single sign-on (SSO) too. See the Crowd documentation: Integrating Crowd with Atlassian JIRA .

Settings in JIRA for the Crowd Directory Type

Setting	Description
Name	A meaningful name that will help you to identify this Crowd server amongst your list of directory servers. Examples: • `Crowd Server` • `Example Company Crowd`
Server URL	The web address of your Crowd console server. Examples: • `http://www.example.com:8095/crowd/` • `http://crowd.example.com`
Application Name	The name of your application, as recognised by your Crowd server. Note that you will need to define the application in Crowd too, using the Crowd administration Console. See the Crowd documentation on adding an application.
Application Password	The password which the application will use when it authenticates against the Crowd framework as a client. This must be the same as the password you have registered in Crowd for this application. See the Crowd documentation on adding an application.

Crowd Permissions

Setting	Description
Read Only	The users, groups and memberships in this directory are retrieved from Crowd and can only be modified via Crowd. You cannot modify Crowd users, groups or memberships via the application administration screens.
Read/Write	The users, groups and memberships in this directory are retrieved from Crowd. When you modify a user, group or membership via the application administration screens, the changes will be applied directly to Crowd. Please ensure that the application has modification permissions for the relevant directories in Crowd. See the Crowd documentation: Specifying an Application's Directory Permissions.

Advanced Crowd Settings

Setting	Description
Enable Nested Groups	Enable or disable support for nested groups. Before enabling nested groups, please check to see if the user directory or directories in Crowd support nested groups. When nested groups are enabled, you can define a group as a member of another group. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups.
Synchronisation Interval (minutes)	Synchronisation is the process by which the application updates its internal store of user data to agree with the data on the directory server. The application will send a request to your directory server every x minutes, where 'x' is the number specified here. The default value is 60 minutes.

Connecting JIRA to Another JIRA Server

Subject to certain limitations, you can connect a number of Atlassian web applications to a single JIRA server for centralised user management.

When to use this option: You can only connect to a server running **JIRA 4.3 or later**. Choose this option as an alternative to Atlassian Crowd, for simple configurations with a limited number of users.

Let's assume that you have two JIRA servers, called for example '**JIRA site 1**' and '**JIRA site 2**'. You want JIRA site 2 to manage your users and groups. JIRA site 1 will delegate user management to JIRA site 2.

To connect JIRA site 1 to use JIRA site 2 for user management:

1. Configure JIRA site 2 to recognise JIRA site 1:
    - Log in to JIRA site 2 as a user with the **'JIRA Administrators'** global permission.
    - Select **'Administration' > 'Users' > 'JIRA User Server'**.
      Keyboard shortcut: **'g' + 'g' +** start typing **'jira user'**.
    - **Add** an application.
    - Enter the **application name** and **password** that JIRA site 1 will use when accessing JIRA site 2.
    - Enter the **IP address** or addresses of JIRA site 1. Valid values are:
        - A full IP address, e.g. `192.168.10.12`.
        - A wildcard IP range, using CIDR notation, e.g. `192.168.10.1/16`. For more information, see the introduction to CIDR notation on Wikipedia and RFC 4632.
    - **Save** the new application.
2. Configure JIRA site 1 to delegate user management:
    - Log in to JIRA site 1 as a user with the **'JIRA Administrators'** global permission.
    - Select **'Administration' > 'Users' > 'User Directories'**.
      Keyboard shortcut: **'g' + 'g' +** start typing **'directories'**.
    - **Add** a directory and select type '**Atlassian JIRA**'.
    - Enter the settings as described below. When asked for the **application name** and **password**, enter the values that you defined in the settings on JIRA site 2.
    - Save the directory settings.
    - Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the '**User Directories**' screen. Here is a summary of how the directory order affects the processing:
        - The order of the directories is the order in which they will be searched for users and groups.
        - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

      For details see Managing Multiple Directories.

Settings for the JIRA Directory Type

Setting	Description
Name	A meaningful name that will help you to identify this JIRA server amongst your list of directory servers. Examples: • `JIRA Server` • `My Company JIRA`
Server URL	The web address of your JIRA server. Examples: • `http://www.example.com/8080` • `http://jira.example.com`
Application Name	The name used by your application when accessing the JIRA server that acts as user manager. Note that you will also need to define your application to that JIRA server, via the '**Other Applications**' option in the 'Users, Groups & Roles' section of the 'Administration' menu.
Application Password	The password used by your application when accessing the JIRA server that acts as user manager.

Permissions for the JIRA Directory Type

Setting	Description
Read Only	The users, groups and memberships in this directory are retrieved from the JIRA server that is acting as user manager. They can only be modified via that JIRA server.
Read/Write	The users, groups and memberships in this directory are retrieved from the JIRA server that is acting as user manager. When you modify a user, group or membership, the changes will be applied directly to your application and to the JIRA server that is acting as user manager.

Advanced Settings for the JIRA Directory Type

Setting	Description
Enable Nested Groups	Enable or disable support for nested groups. Before enabling nested groups, please check to see if nested groups are enabled on the JIRA server that is acting as user manager. When nested groups are enabled, you can define a group as a member of another group. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups.
Synchronisation Interval (minutes)	Synchronisation is the process by which the application updates its internal store of user data to agree with the data on the directory server. The application will send a request to your directory server every x minutes, where 'x' is the number specified here. The default value is 60 minutes.

Diagrams of Some Possible Configurations

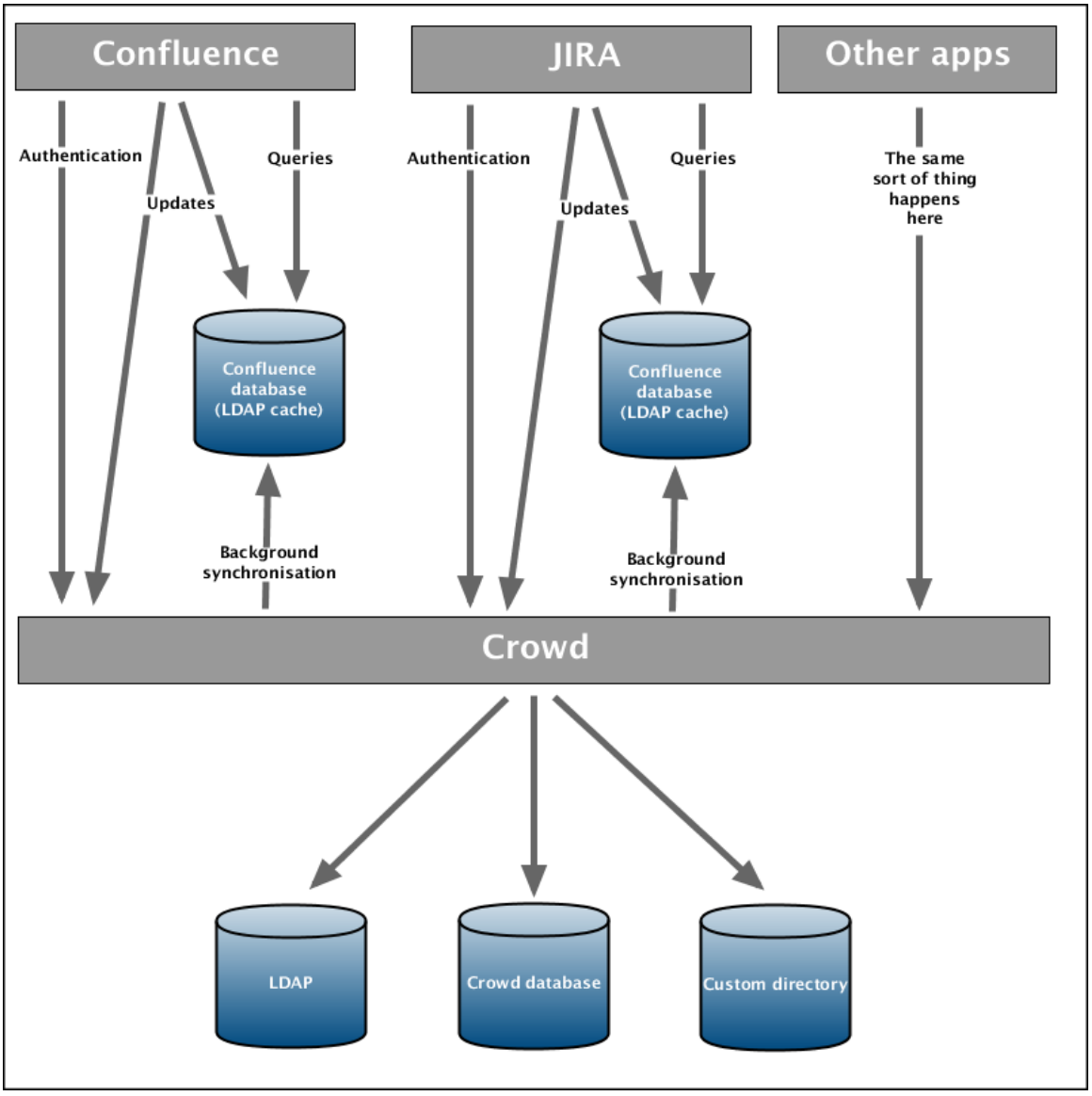


Diagram above: Confluence, JIRA and other applications connecting to Crowd for user management.

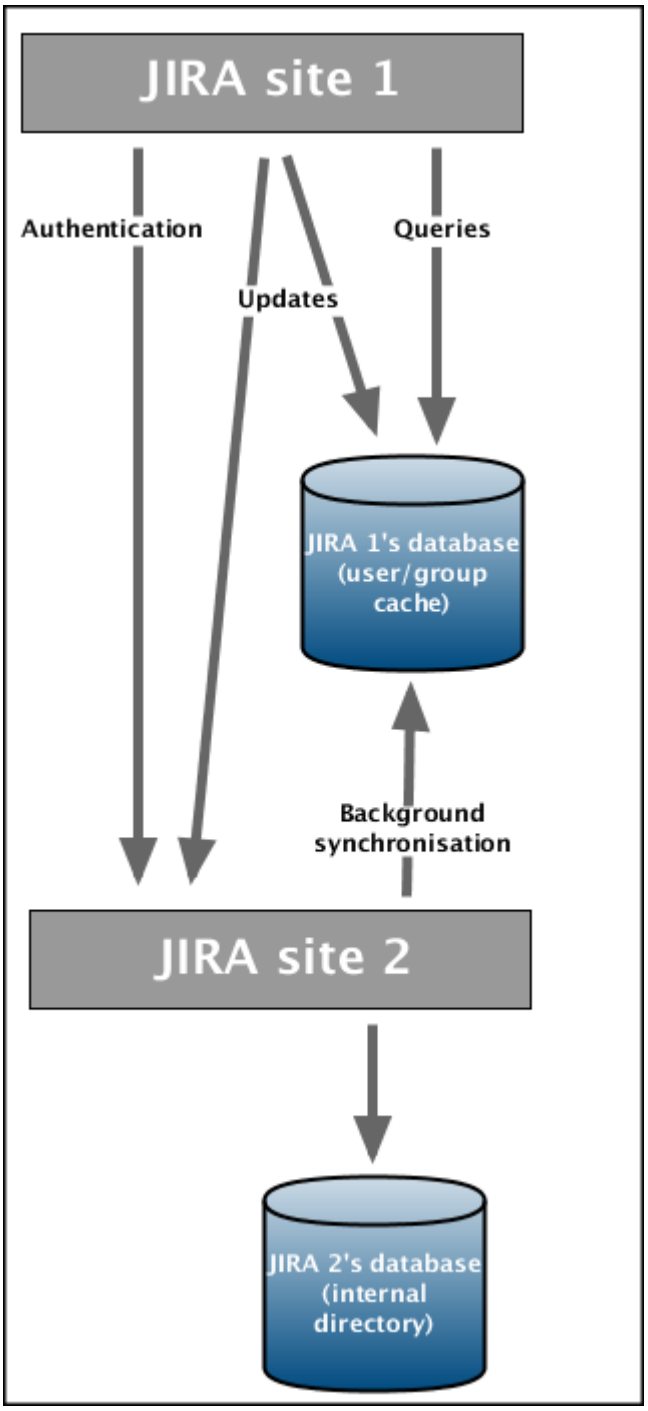


Diagram above: One JIRA site connecting to another for user management. JIRA site 2 does the user management, storing the user data in its internal directory.

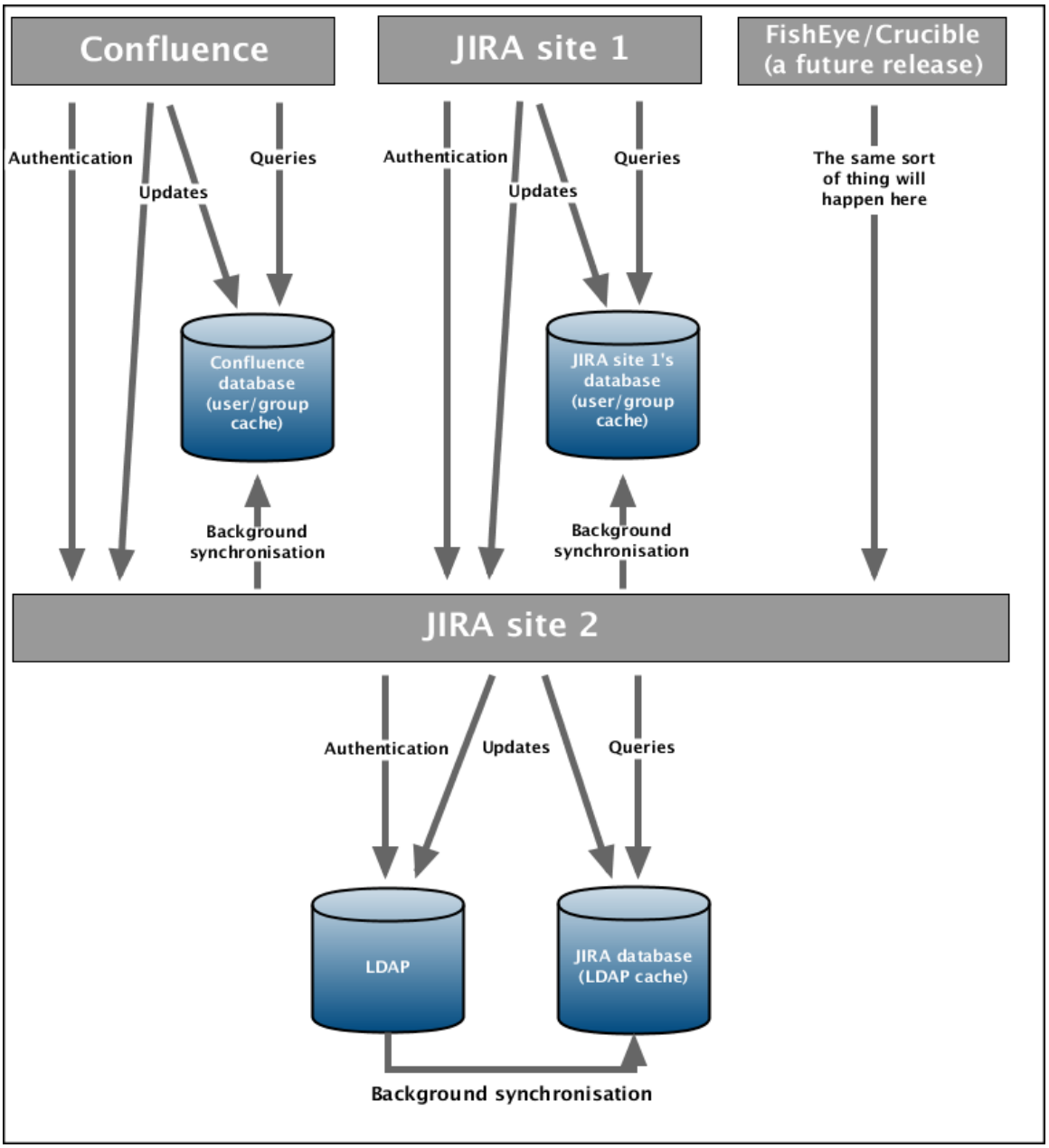


Diagram above: A number of applications connecting to JIRA (site 2) for user management, with JIRA in turn connecting to an LDAP server.

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Managing Multiple Directories

This page describes what happens when you have defined more than one user directory in JIRA. For example, you may have an internal directory and you may also connect to an LDAP directory server and/or other types of user directories. When you connect to a new directory server, you also need to define the **directory order**.

Duplicate usernames across directories are not supported. If you are connecting to more than one user directory, please ensure that the usernames are unique to one directory. For example, if you have a user `jsmith` in both 'Directory1' and 'Directory2', that is an unsupported configuration.

Here is a summary of how the directory order affects the processing:

- The order of the directories is the order in which they will be searched for users and groups.
- Changes to users and groups will be made only in the first directory where the application has permission to make changes.

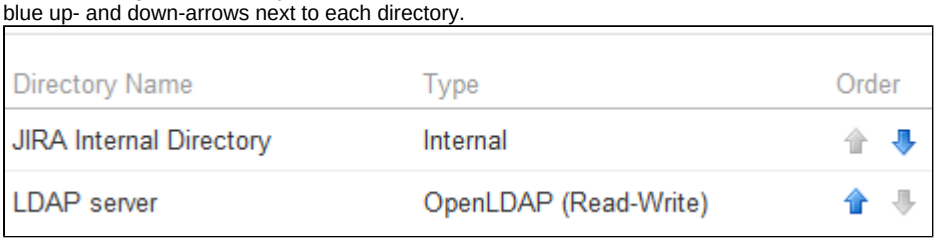
On this page:

- Configuring the Directory Order
- Effect of Directory Order
  - Login
  - Permissions
  - Updating Users and groups

*The information on this page does not apply to **JIRA OnDemand**.*

Configuring the Directory Order

You can change the order of your directories as defined to JIRA. Select '**User Directories**' from the JIRA administration menu and click the blue up- and down-arrows next to each directory.



Notes:

- Please read the rest of this page to understand what effect the directory order will have on authentication (login) and permissions in JIRA, and what happens when you update users and groups in JIRA.

Effect of Directory Order

This section summarises the effect the order of the directories will have on login and permissions, and on the updating of users and groups.

Login

The directory order is significant during the authentication of the user, in cases where the same user exists in multiple directories. When a user attempts to log in, the application will search the directories in the order specified, and will use the credentials (password) of the *first occurrence of the user* to validate the login attempt.

Permissions

The directory order is significant when granting the user permissions based on group membership. If the same username exists in more than one directory, the application will look for group membership only in the first directory where the username appears, based on the directory order.

Example:

- You have connected two directories: The Customers directory and the Partners directory.
- The Customers directory is first in the directory order.
- A username `jsmith` exists in both the Customers directory and the Partners directory.
- The user `jsmith` is a member of group `G1` in the Customers directory and group `G2` in the Partners directory.
- The user `jsmith` will have permissions based on membership of `G1` only, not `G2`.

Updating Users and groups

If you update a user or group via the application's administration screens, the update will be made in the first directory where the application has write permissions.

Example 1:

- You have connected two directories: The Customers directory and the Partners directory.
- The application has permission to update both directories.
- The Customers directory is first in the directory order.
- A username `jsmith` exists in both the Customers directory and the Partners directory.
- You update the email address of user `jsmith` via the application's administration screens.
- The email address will be updated in the Customers directory only, not the Partners directory.

Example 2:

- You have connected two directories: A read/write LDAP directory and the internal directory.
- The LDAP directory is first in the directory order.
- All new users will be added to the LDAP directory. It is not possible to add a new user to the internal directory.

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Synchronising Data from External Directories

For certain directory types, JIRA stores a cache of directory information (users and groups) in the application database, to ensure fast recurrent access to user and group data. A synchronisation task runs periodically to update the internal cache with changes from the externa directory.

On this page:

- Affected Directory Types
- How it Works
- Finding the Time Taken to Synchronise
- Manually Synchronising the Cache
- Configuring the Synchronisation Interval

*The information on this page does not apply to **JIRA OnDemand**.*

Affected Directory Types

Data caching and synchronisation apply to the following user directory types:

- **LDAP** (Microsoft Active Directory and all supported LDAP directories) where permissions are set to **read only**.
- **LDAP** (Microsoft Active Directory and all supported LDAP directories) where permissions are set to **read only, with local groups**.
- **LDAP** (Microsoft Active Directory and all supported LDAP directories) where permissions are set to **read/write**.
- **Atlassian Crowd**.
- **Atlassian JIRA**.

Data caching and synchronisation do not occur for the following user directory types:

- **LDAP** (Microsoft Active Directory and all supported LDAP directories) where permissions are set to **authentication only, with local groups**.
- **Internal Directory with LDAP Authentication**.
- **Internal Directory**.

How it Works

Here is a summary of the caching functionality:

- The caches are held in the application database.
- When you connect a new external user directory to the application, a synchronisation task will start running in the background to copy all the required users, groups and membership information from the external directory to the application database. This task may take a while to complete, depending on the size and complexity of your user base.
- Note that a user will not be able to log in until the synchronisation task has copied that user's details into the cache.
- A periodic synchronisation task will run to update the database with any changes made to the external directory. The default synchronisation interval, or polling interval, is one hour (60 minutes). You can change the synchronisation interval on the directory configuration screen.
- You can manually synchronise the cache if necessary.
- If the external directory permissions are set to read/write: Whenever an update is made to the users, groups or membership information via the application, the update will also be applied to the cache and the external directory immediately.
- All authentication is happens via calls to the external directory. When caching information from an external directory, the application database does not store user passwords.
- All other queries run against the internal cache.

Finding the Time Taken to Synchronise

The '**User Directories**' screen shows information about the last synchronisation operation, including the length of time it took.

Manually Synchronising the Cache

You can manually synchronise the cache by clicking '**Synchronise**' on the '**User Directories**' screen. If a synchronisation operation is already in progress, you cannot start another until the first has finished.

Screen snippet: User directories, showing information about synchronisation

Configuring the Synchronisation Interval

You can set the '**Synchronisation Interval**' on the directory configuration screen. The synchronisation interval is the period of time to wait between requests for updates from the directory server.

The length you choose for your synchronisation interval depends on:

- The length of time you can tolerate stale data.
- The amount of load you want to put on the application and the directory server.
- The size of your user base.

If you synchronise more frequently, then your data will be more up to date. The downside of synchronising more frequently is that you may overload your server with requests.

If you are not sure what to do, we recommend that you start with an interval of 60 minutes (this is the default setting) and reduce the value incrementally. You will need to experiment with your setup.
RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Managing Nested Groups

Some directory servers allow you to define a group as a member of another group. Groups in such a structure are called 'nested groups'. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups.
This page describes how JIRA handles nested groups that exist in one or more of your directory servers.

Enabling Nested Groups

You can enable or disable support for nested groups on each directory individually. Select '**User Directories**' from the JIRA administration menu, **edit** the directory and select '**Enable Nested Groups'**. See Configuring User Directories.

Notes:

- Before enabling nested groups for a specific directory type in JIRA, please make sure that your directory server supports nested groups.
- Please read the rest of this page to understand what effect nested groups will have on authentication (login) and permissions in JIRA, and what happens when you update users and groups in JIRA.

On this page:

- Enabling Nested Groups
- Effect of Nested Groups
  - Login
  - Permissions
  - Viewing Lists of Group Members
  - Adding and Updating Group Memberships
- Examples
  - Example 1: User is Member of Sub-Group
  - Example 2: Sub-Groups as Members of the 'jira-developers' group
  - Example 3: Sub-Groups as Members of the 'confluence-users' group
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Effect of Nested Groups

This section summarises the effect nested groups will have on login and permissions, and on the viewing and updating of users and groups.

Login

When a user logs in, they will be allowed access to the application if they belong to an authorised group or any of its sub-groups.

Permissions

The user will be allowed access to a function if they belong to a group that has the necessary permissions, or if they belong to any of its sub-groups.

Viewing Lists of Group Members

If you ask to view the members of a group, you will see all users who are members of the group and all users belonging its sub-groups, consolidated into one list. We call this a '**flattened**' list.

You cannot view or edit the nested groups themselves. You will not be able to see that one group is a member of another group.

Adding and Updating Group Memberships

If you add a user to a group, the user is added to the named group and not to any other groups.

If you try to remove a user from a flattened list, the following will happen:

- If the user is a member of the top group in the hierarchy (tree) of groups contained in the flattened list, the user will be removed from the group.
- Otherwise, you will see an error message stating that the user is not a direct member of the group.

Examples

Example 1: User is Member of Sub-Group

Let's assume that the following two groups exist in your directory server:

- `staff`
- `marketing`

Memberships:

- The `marketing` group is a member of the `staff` group.
- User `jsmith` is a member of `marketing`.

You will see that `jsmith` is a member of both `marketing` and `staff`. You will not see that the two groups are nested. If you assign permissions to the `staff` group, then `jsmith` will get those permissions.

Example 2: Sub-Groups as Members of the 'jira-developers' group

In an LDAP directory server, we have groups '**engineering-group**' and '**techwriters-group**'. We want to grant both groups developer-level access to our JIRA site.

- Add a group called '**jira-developers**'.
- Add the '**engineering-group**' as a sub-group of '**jira-developers**'.
- Add the '**techwriters-group**' as a sub-group of '**jira-developers**'.

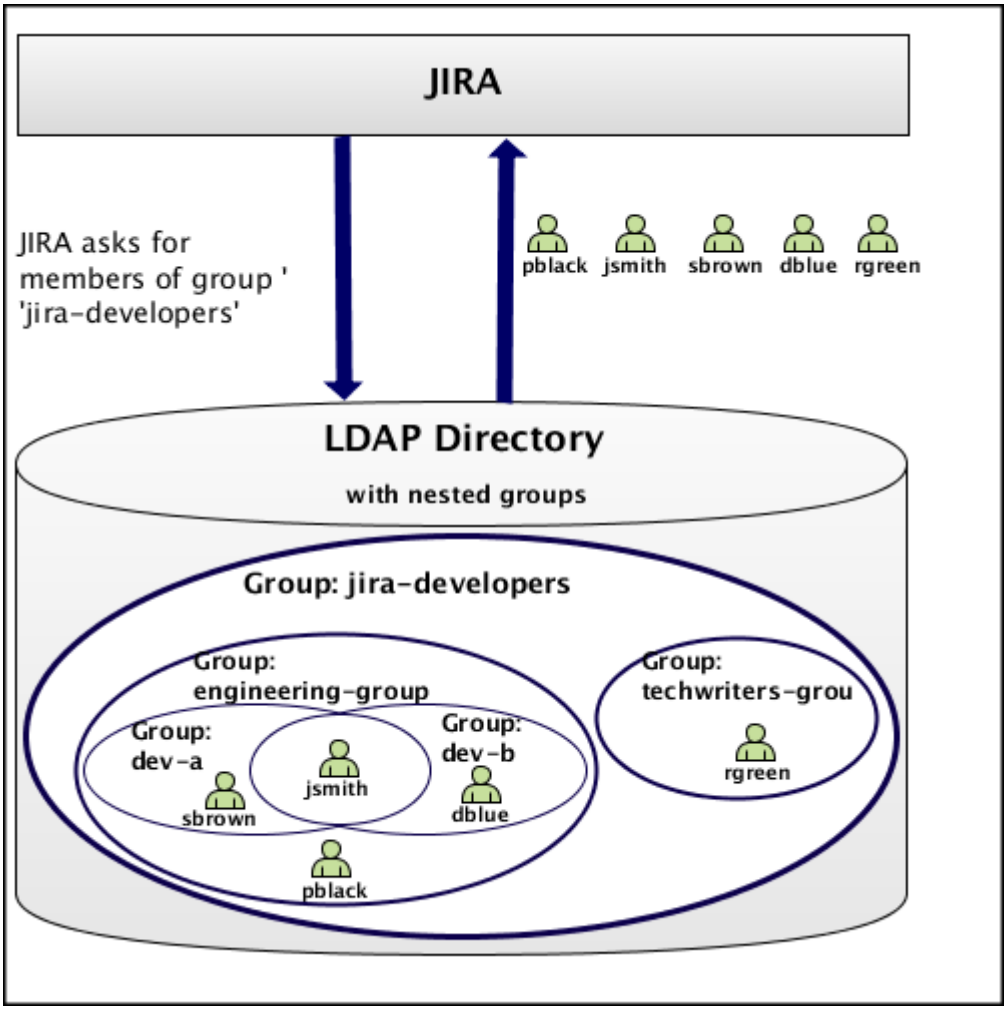
Group memberships are now:

- **jira-developers** — sub-groups: **engineering-group**, **techwriters-group**
- **engineering-group** — sub-groups: **dev-a**, **dev-b**; users: **pblack**
- **dev-a** — users: **jsmith**, **sbrown**
- **dev-b** — users: **jsmith**, **dblue**
- **techwriters-group** — users: **rgreen**

When JIRA requests a list of users in the '**jira-developers**' group, it will receive the following list:

- **pblack**
- **jsmith**
- **sbrown**
- **dblue**
- **rgreen**

Diagram: Sub-groups as members of the 'jira-developers' group



Example 3: Sub-Groups as Members of the 'confluence-users' group

In an LDAP directory server, we have groups '**engineering-group**' and '**payroll-group**'. We want to grant both groups access to our Confluence site.

- Add a group called '**confluence-users**'.
- Add the '**engineering-group**' as a sub-group of '**confluence-users**'.
- Add the '**payroll-group**' as a sub-group of '**confluence-users**'.

Group memberships are now:

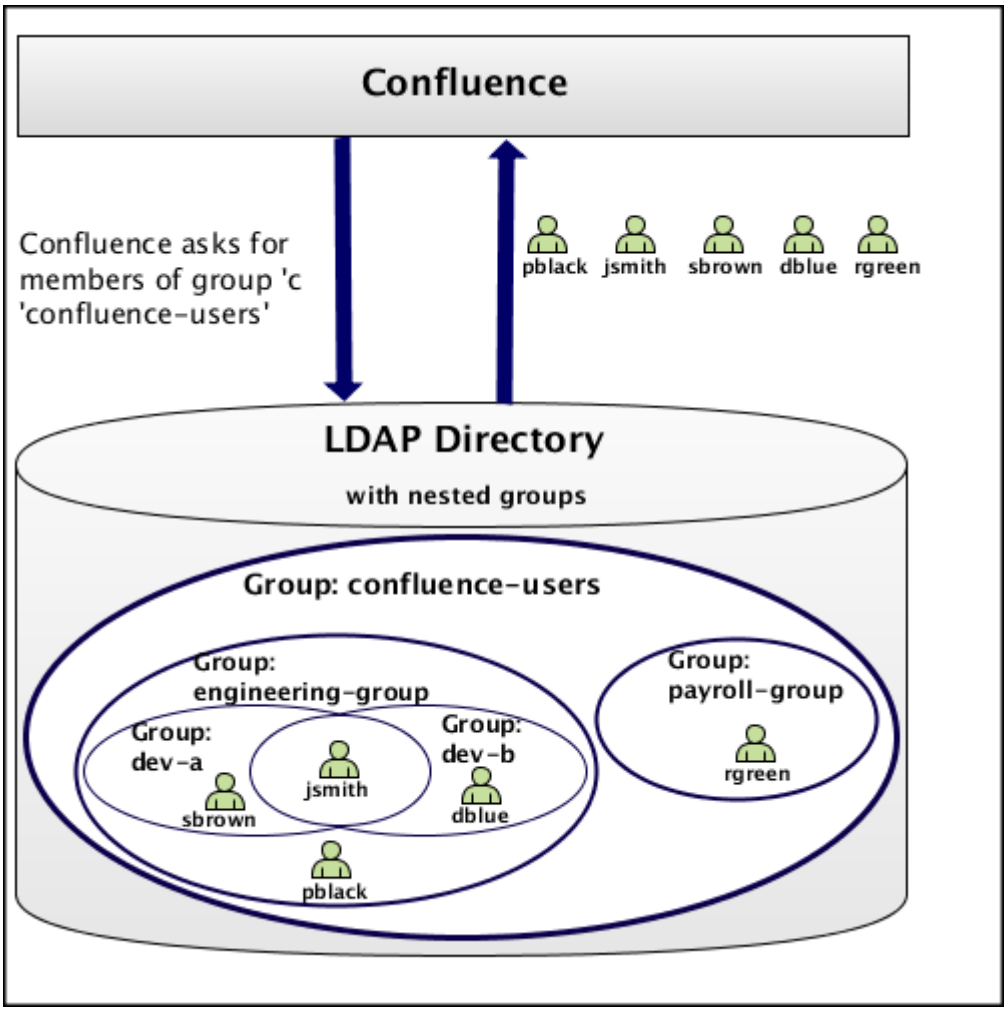
- **confluence-users** — sub-groups: **engineering-group**, **payroll-group**
- **engineering-group** — sub-groups: **dev-a**, **dev-b**; users: **pblack**
- **dev-a** — users: **jsmith**, **sbrown**
- **dev-b** — users: **jsmith**, **dblue**
- **payroll-group** — users: **rgreen**

When Confluence requests a list of users in the '**confluence-users**' group, it will receive the following list:

- **pblack**

- **jsmith**
- **sbrown**
- **dblue**
- **rgreen**

Diagram: Sub-groups as members of the 'confluence-users' group



Notes

- **Possible impact on performance.** Enabling nested groups may result in slower user searches.
- **Definition of nested groups in LDAP.** In an LDAP directory, a nested group is defined as a child group entry whose DN (Distinguished Name) is referenced by an attribute contained within a parent group entry. For example, a parent group '**Group One**' might have an `objectClass=group` attribute and one or more `member=DN` attributes, where the DN can be that of a user *or* that of a group elsewhere in the LDAP tree:

```
member=CN=John Smith,OU=Users,OU=OrgUnitA,DC=sub,DC=domain
member=CN=Group Two,OU=OrgUnitBGroups,OU=OrgUnitB,DC=sub,DC=domain
```

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Diagrams of Possible Configurations for User Management

The aim of these diagrams is to help people understand each directory type at a glance. We have kept the diagrams simple and conceptual, with just enough information to be correct.

Some things that we do **not** attempt to show:

- In most cases, we do not attempt to show that you can have multiple directory types mapped to JIRA at the same time. We illustrate that fact in just the first two LDAP diagrams.
- We have not included a diagram for Confluence's legacy connection to JIRA database.
- We do not attempt to show all of the possible configurations and layered connections that are available now that you can use JIRA as a directory manager.

On this page:

- JIRA Internal Directory
- JIRA with Read/Write Connection to LDAP
- JIRA with Read-Only Connection to LDAP, with Local Groups
- JIRA Internal Directory with LDAP Authentication
- JIRA with LDAP Authentication, Copy Users on First Login
- One JIRA Site Connecting to Another
- Confluence and JIRA Connecting to Crowd
- A Number of Applications Connecting to JIRA

*The information on this page does not apply to **JIRA OnDemand**.*

JIRA Internal Directory

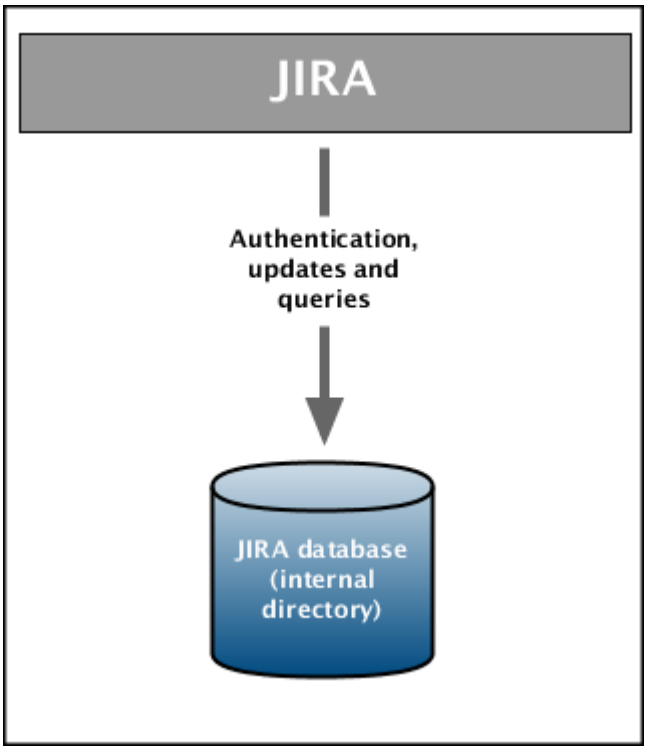


Diagram above: JIRA using its internal directory for user management.

JIRA with Read/Write Connection to LDAP

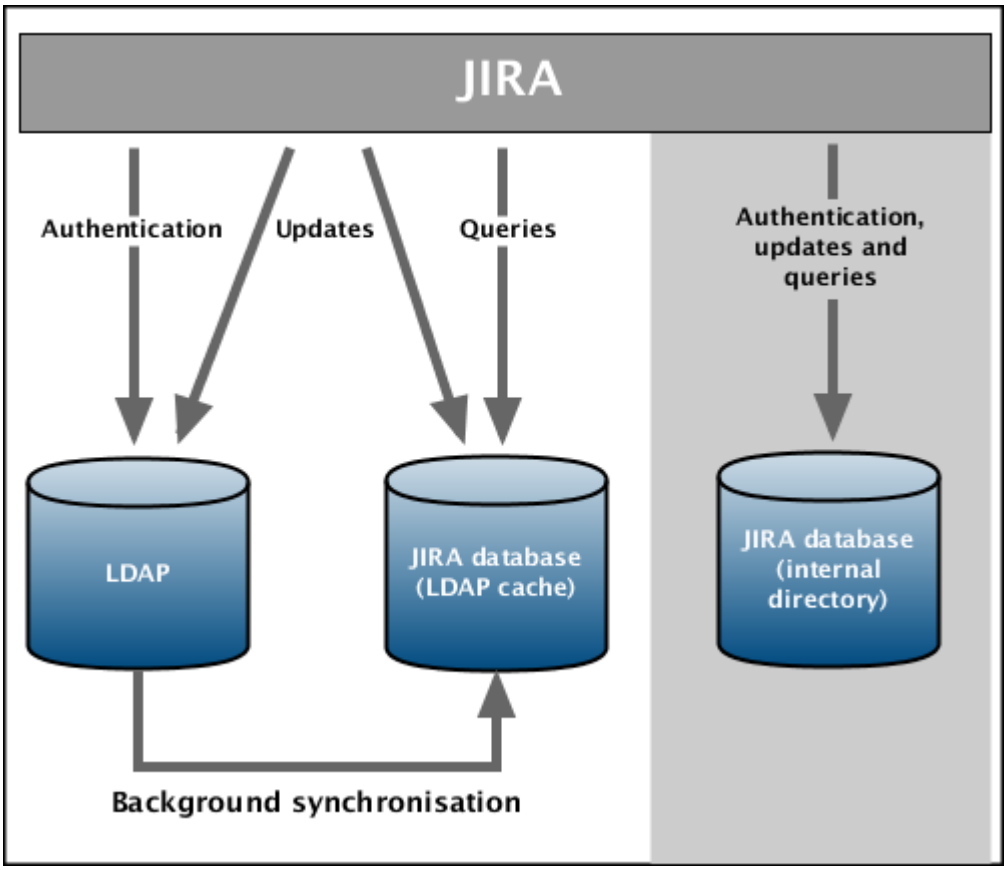


Diagram above: JIRA connecting to an LDAP directory.

JIRA with Read-Only Connection to LDAP, with Local Groups

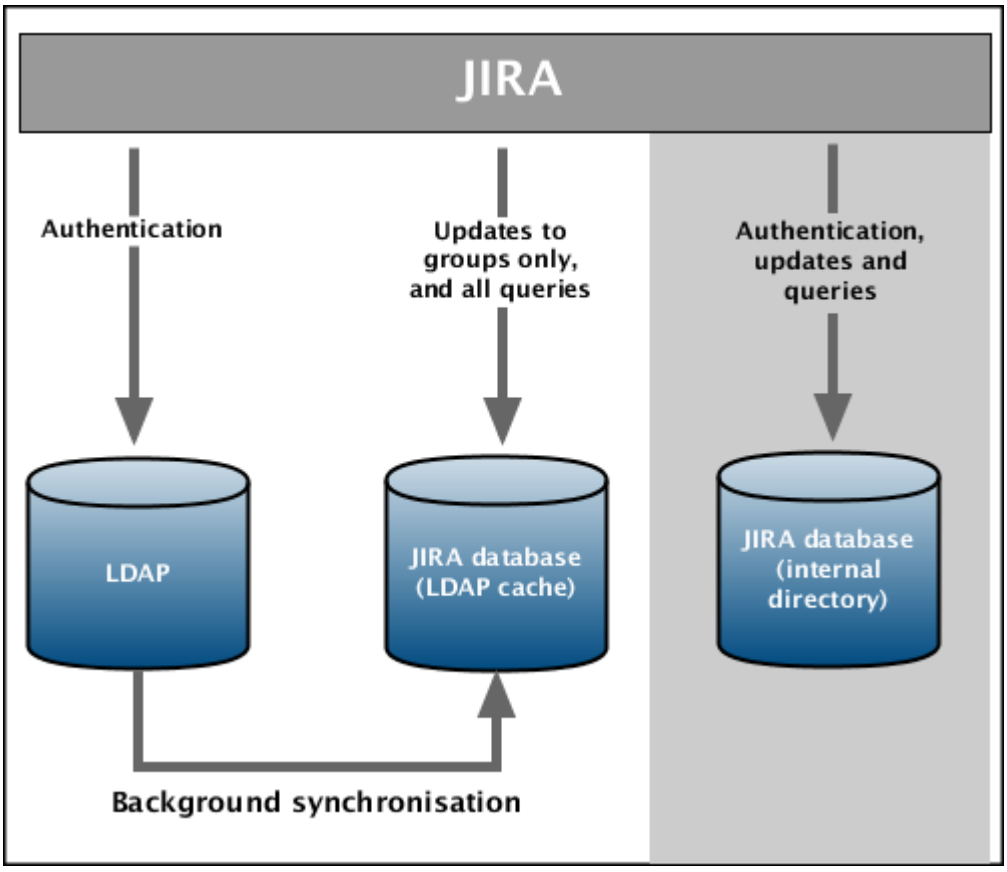


Diagram above: JIRA connecting to an LDAP directory with permissions set to read only and local groups.

JIRA Internal Directory with LDAP Authentication

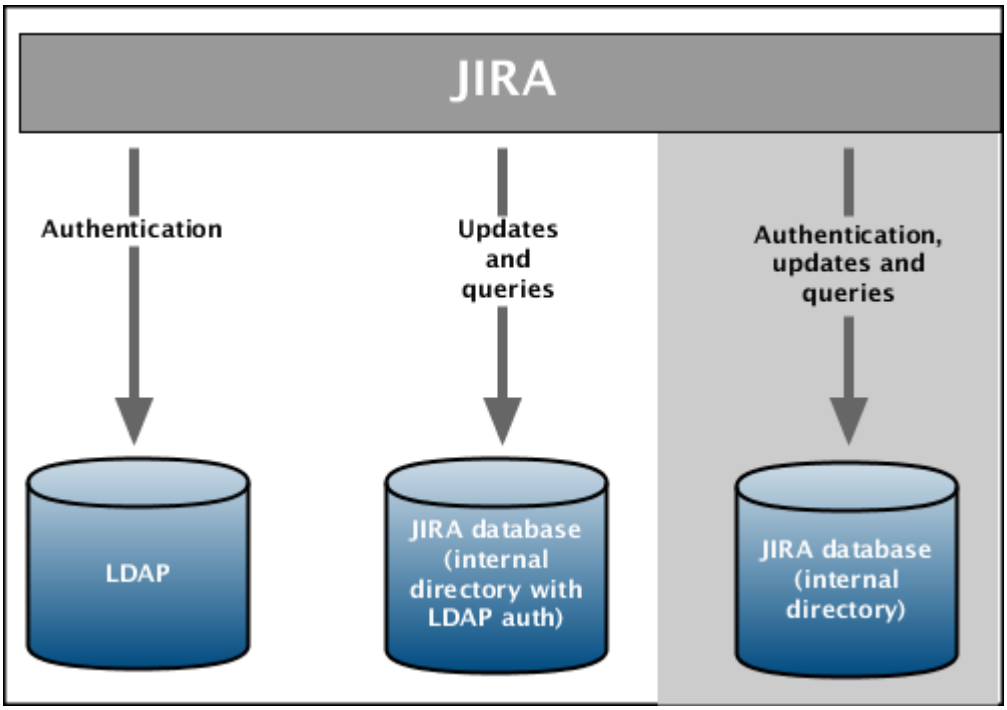


Diagram above: JIRA connecting to an LDAP directory for authentication only.

JIRA with LDAP Authentication, Copy Users on First Login

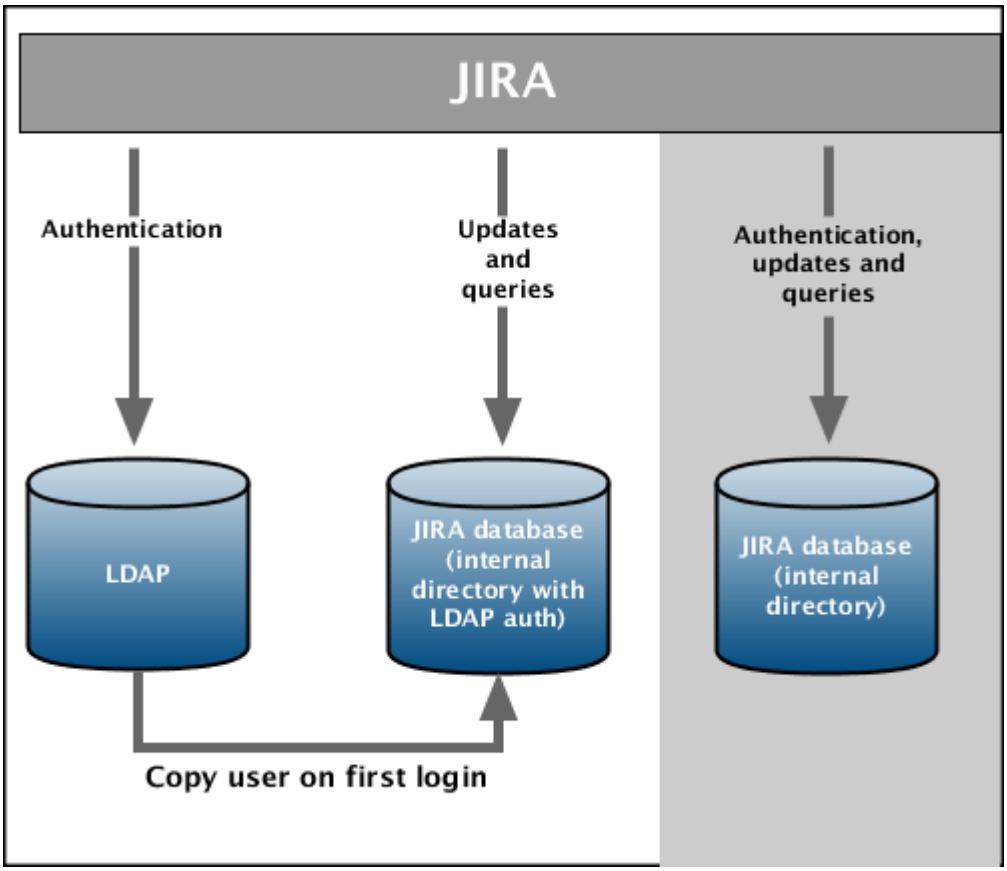


Diagram above: JIRA connecting to an LDAP directory for authentication only, with each user copied to the internal directory when they first log in to JIRA.

One JIRA Site Connecting to Another

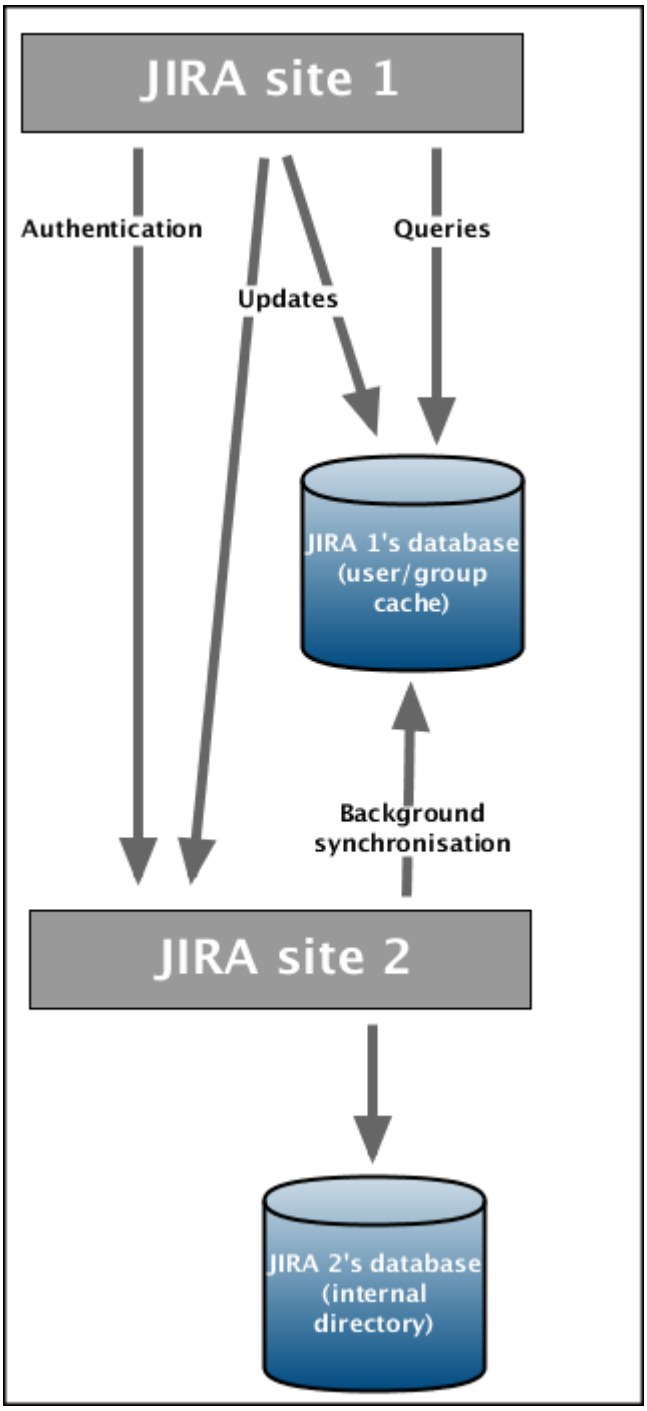


Diagram above: One JIRA site connecting to another for user management. JIRA site 2 does the user management, storing the user data in its internal directory.

Confluence and JIRA Connecting to Crowd

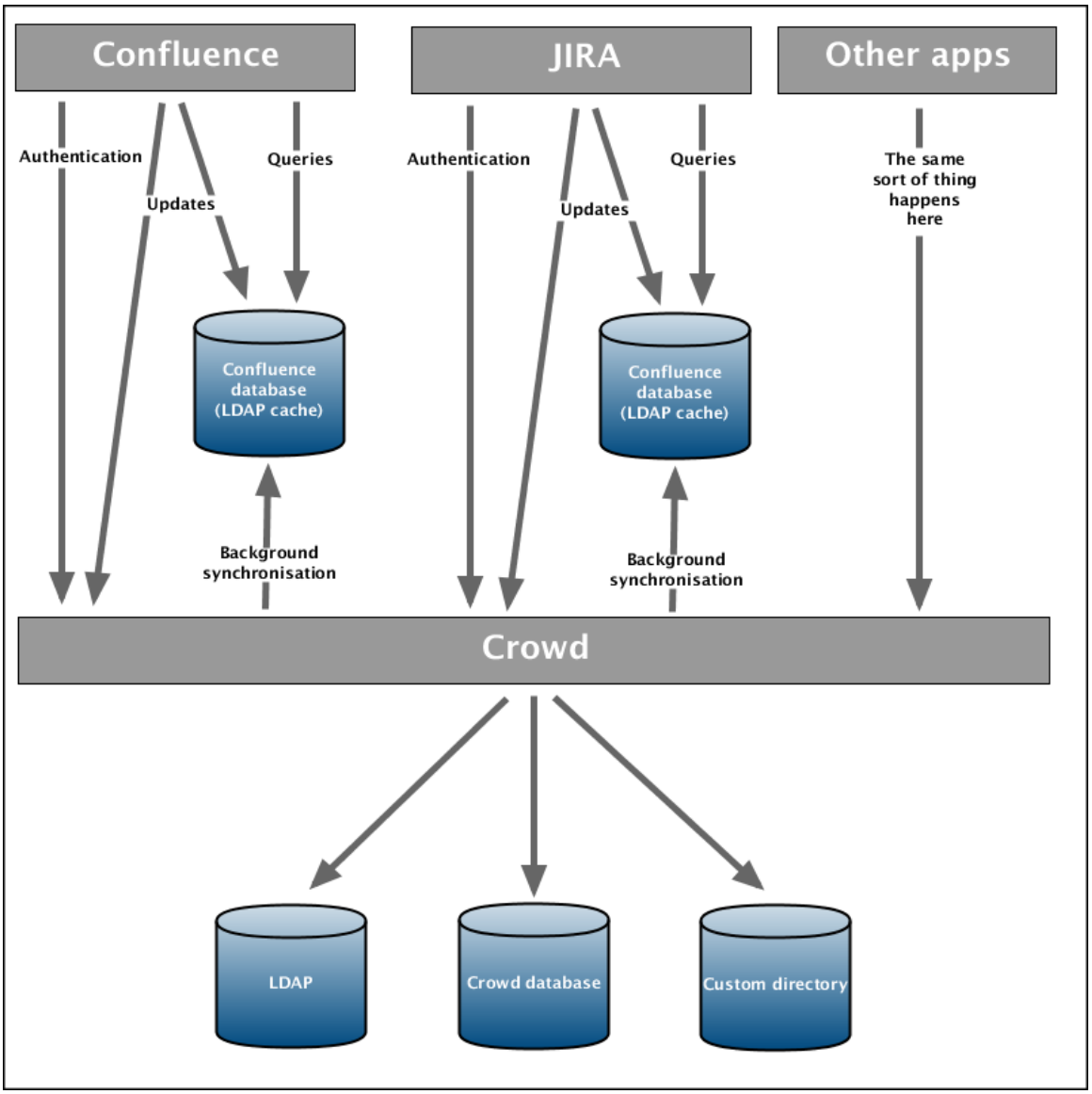


Diagram above: Confluence, JIRA and other applications connecting to Crowd for user management.

A Number of Applications Connecting to JIRA

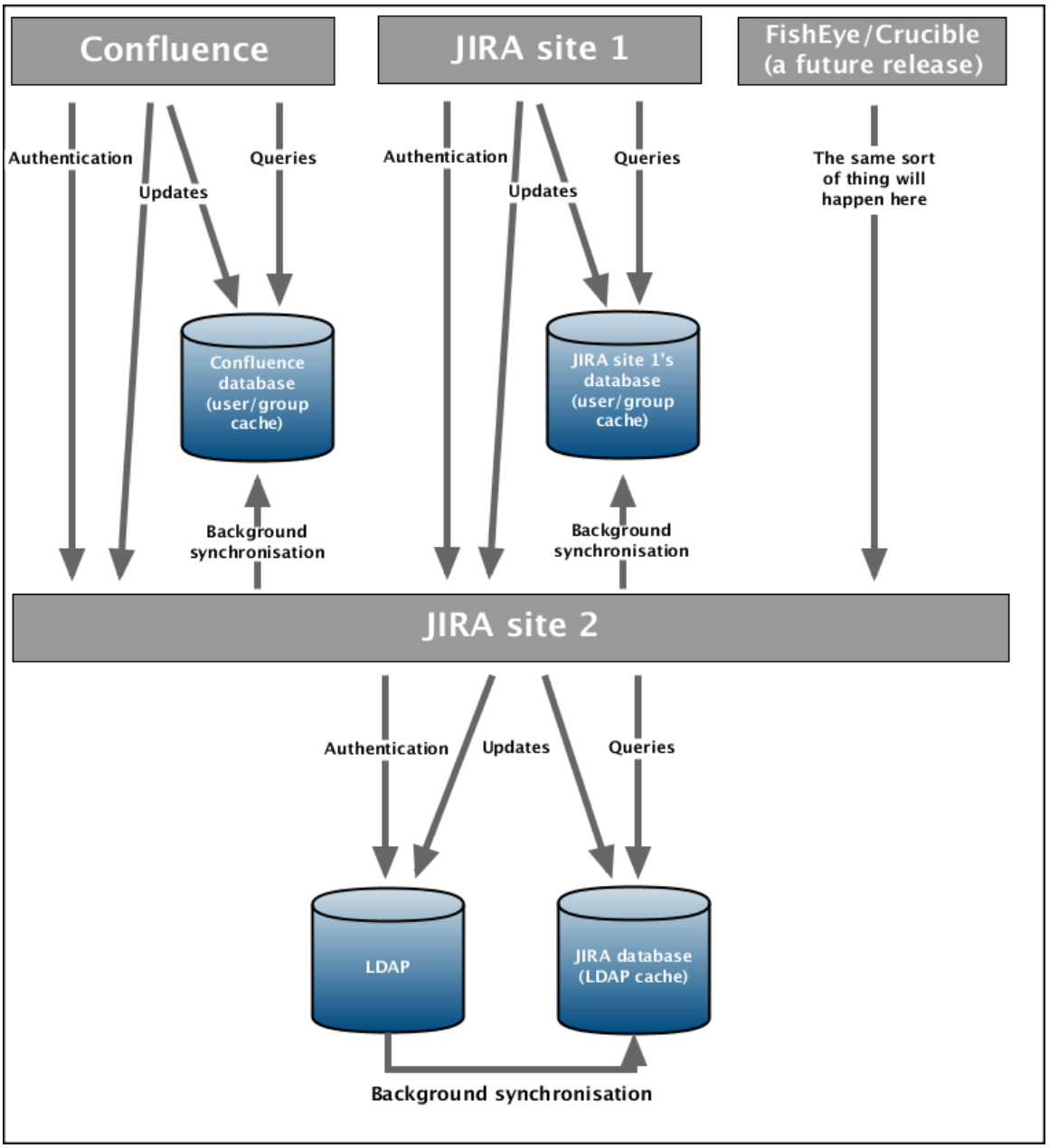


Diagram above: A number of applications connecting to JIRA (site 2) for user management, with JIRA in turn connecting to an LDAP server.

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

User Management Limitations and Recommendations

This page describes the optimal configurations and limitations that apply to user management in JIRA.

On this page:

- General Recommendations
- Recommendations for Connecting to LDAP
- Recommendations for Connecting to Another JIRA Server

*The information on this page does not apply to **JIRA OnDemand**.*

General Recommendations

- **Duplicate usernames across directories are not supported**. If you are connecting to more than one user directory, please ensure that the usernames are unique to one directory. For example, if you have a user `jsmith` in both 'Directory1' and 'Directory2', that is an unsupported configuration.
- **Be careful when deleting users in remote directories**. If you are connecting to an LDAP directory, a Crowd directory or a remote JIRA directory, please take care when deleting users from the remote directory. If you delete a user that is associated with data in JIRA, this will cause problems in JIRA. We recommend that you perform all user management in JIRA, because the JIRA UI will prevent the deletion of a user if there are issues assigned to the user, reported by the user or the user is a project lead.

Recommendations for Connecting to LDAP

Please consider the following limitations and recommendations when connecting to an LDAP user directory.

Optimal Number of Users and Groups in your LDAP Directory

The connection to your LDAP directory provides powerful and flexible support for connecting to, configuring and managing LDAP directory servers. To achieve optimal performance, a background synchronisation task loads the required users and groups from the LDAP server into the application's database, and periodically fetches updates from the LDAP server to keep the data in step. The amount of time needed to copy the users and groups rises with the number of users, groups, and group memberships. For that reason, we recommended a maximum number of users and groups as described below.

This recommendation affects connections to LDAP directories:

- Microsoft Active Directory
- All other LDAP directory servers

The following LDAP configurations are **not** affected:

- Internal directories with LDAP authentication
- LDAP directories configured for 'Authentication Only, Copy User On First Login'

Please choose one of the following solutions, depending on the number of users, groups and memberships in your LDAP directory.

Your environment	Recommendation
Up to 10 000 (ten thousand) users, 1000 (one thousand) groups, and 20 (twenty) groups per user	Choose the '**LDAP**' or '**Microsoft Active Directory**' directory type. You can make use of the full synchronisation option. Your application's database will contain all the users and groups that are in your LDAP server.
More than the above	Use LDAP filters to reduce the number of users and groups visible to the synchronisation task.

Our Test Results

We performed internal testing of synchronisation with an AD server on our local network consisting of 10 000 users, 1000 groups and 200 000 memberships.

We found that the initial synchronisation took about 5 minutes. Subsequent synchronisations with 100 modifications on the AD server took a couple of seconds to complete.

Please keep in mind that a number of factors come into play when trying to tune the performance of the synchronisation process, including:

- **Size of userbase.** Use LDAP filters to keep this to the minimum that suits your requirements.
- **Type of LDAP server.** We currently support change detection in AD, so subsequent synchronisations are much faster for AD than for other LDAP servers.
- **Network topology.** The further away your LDAP server is from your application server, the more latent LDAP queries will be.
- **Database performance.** As the synchronisation process caches data in the database, the performance of your database will affect the performance of the synchronisation.
- **JVM heap size.** If your heap size is too small for your userbase, you may experience heavy garbage collection during the synchronisation process which could in turn slow down the synchronisation.

Redundant LDAP is Not Supported

The LDAP connections do not support the configuration of two or more LDAP servers for redundancy (automated failover if one of the servers goes down).

Specific Notes for Connecting to Active Directory

When the application synchronises with Active Directory (AD), the synchronisation task requests only the changes from the LDAP server rather than the entire user base. This optimises the synchronisation process and gives much faster performance on the second and subsequent requests.

On the other hand, this synchronisation method results in a few limitations:

1. **Externally moving objects out of scope or renaming objects causes problems in AD.** If you move objects out of scope in AD, this will result in an inconsistent cache. We recommend that you do not use the external LDAP directory interface to move objects out of the scope of the sub-tree, as defined on the application's directory configuration screen. If you do need to make structural changes to your LDAP directory, manually synchronise the directory cache after you have made the changes to ensure cache consistency.
2. **Synchronising between AD servers is not supported.** Microsoft Active Directory does not replicate the uSNChanged attribute across instances. For that reason, we do not support connecting to different AD servers for synchronisation. (You can of course define multiple different directories, each pointing to its own respective AD server.)
3. **Synchronising with AD servers behind a load balancer is not supported.** As with synchronising between two different AD servers, Microsoft Active Directory does not replicate the uSNChanged attribute across instances. For that reason, we do not support connecting to different AD servers even when they are load balanced. You will need to select one server (preferably one that is local) to synchronise with instead of using the load balancer.
4. **You must restart the application after restoring AD from backup.** On restoring from backup of an AD server, the uSNChanged timestamps are reverted to the backup time. To avoid the resulting confusion, you will need to flush the directory cache after a Active Directory restore operation.
5. **Obtaining AD object deletions requires administrator access.** Active Directory stores deleted objects in a special container called cn=Deleted Objects. By default, to access this container you need to connect as an administrator and so, for the synchronisation task to be aware of deletions, you must use administrator credentials. Alternatively, it is possible to change the permissions on the cn=Deleted Objects container. If you wish to do so, please see this Microsoft KB Article.
6. **The User DN used to connect to AD must be able to see the uSNChanged attribute.** The synchronisation task relies on the uSNChanged attribute to detect changes, and so must be in the appropriate AD security groups to see this attribute for all LDAP objects in the subtree.

Recommendations for Connecting to Another JIRA Server

Please consider the following limitations and recommendations when connecting to a JIRA server for user management.

Single Sign-On Across Multiple Applications is Not Supported

When you connect to JIRA for user management, you will not have single sign-on across the applications connected in this way. JIRA, when acting as a directory manager, does not support SSO.

Custom Application Connectors are Not Supported

JIRA, Confluence, FishEye, Crucible and Bamboo can connect to a JIRA server for user management. Custom application connectors will need to use the new REST API.

Custom Directories are Not Supported

Earlier versions of JIRA supported OSUser Providers. It was therefore possible write a special provider to obtain user information from any external user directory. This is no longer the case.

Optimal Number of Users and Applications

Please consider the following limitations when connecting to a JIRA server for user management:

- Maximum 500 users.
- Maximum 5 connected applications.

Recommendations

Your environment	Recommendation
If **all** the following are true: • You have fewer than 500 users. • You want to share user and group management across just a few applications, such as one JIRA server and one Confluence server, or two JIRA servers. • You do not need single sign-on (SSO) between JIRA and Confluence, or between two JIRA servers. • You do not have custom application connectors. Or, if you do have them, you are happy to convert them to use the new REST API. • You are happy to shut down all your servers when you need to upgrade JIRA.	Your environment meets the optimal requirements for using JIRA for user management.

If **one or more** of the following are true: • You have more than 500 users. • You want to share user and group management across more than 5 applications. • You need single sign-on (SSO) across multiple applications. • You have custom applications integrated via the Crowd SOAP API, and you cannot convert them to use the new REST API. • You are not happy to shut down all your servers when you need to upgrade JIRA.	We recommend that you install Atlassian Crowd for user management and SSO.
If you are considering creating a custom directory connector to define your own storage for users and groups...	Please see if one of the following solutions will work for you: • If you have written a custom provider to support a specific LDAP schema, please check the supported LDAP schemas to see if you can use one of them instead. • If you have written a custom provider to support nested groups, please consider enabling nested groups in the supported directory connectors instead. • If you have written a custom provider to connect to your own database, please consider loading the data into the application's database instead. • If you need to keep the custom directory connection, please consider whether Atlassian Crowd meets your requirements. See the documentation on Creating a Custom Directory Connector.

RELATED TOPICS

Connecting to an LDAP Directory
Connecting to Crowd or Another JIRA Server for User Management
Configuring User Directories

Allowing Other Applications to Connect to JIRA for User Management

You can allow other applications to connect to your JIRA server for management of users and groups, and for authentication (verification of a user's login). Examples of such applications: Atlassian Confluence, FishEye/Crucible, Bamboo, or another JIRA server.

On this page:

- Allowing an Application to Connect to JIRA for User Management
- Diagrams of Some Possible Configurations

*The information on this page does not apply to **JIRA OnDemand**.*

Allowing an Application to Connect to JIRA for User Management

Subject to certain limitations, you can connect a number of Atlassian web applications to a single JIRA server for centralised user management.

When to use this option: You can only connect to a server running **JIRA 4.3 or later**. Choose this option as an alternative to Atlassian Crowd, for simple configurations with a limited number of users.

To configure an application to connect to JIRA as a user server:

1. Add the application in JIRA:
    a. Log in to JIRA as a user with the **'JIRA Administrators'** global permission.
    b. Select **'Administration' > 'Users' > 'JIRA User Server'**.
    Keyboard shortcut: **'g' + 'g' +** start typing **'jira user'**.
    c. **Add** an application.
    d. Enter the **application name** and **password** that the application will use when accessing your JIRA server.
    e. Enter the **IP address** or addresses of the application. Valid values are:
        - A full IP address, e.g. `192.168.10.12`.
        - A wildcard IP range, using CIDR notation, e.g. `192.168.10.1/16`. For more information, see the introduction to CIDR notation on Wikipedia and RFC 4632.
    f. **Save** the new application.
2. Set up the JIRA user directory in the application:
    For example, see Connecting Confluence to JIRA for User Management or Connecting JIRA to Another JIRA Server:
    a. Log in to the application that is going to connect to JIRA for user management.
    b. Go to the application's '**User Directories**' administration area.
    c. Add a new directory of type '**Atlassian JIRA**'.

   d. Define the directory order (see Managing Multiple Directories).
3. Create any groups in JIRA that are required by the application. For example, see Connecting Confluence to JIRA for User Management.

Diagrams of Some Possible Configurations

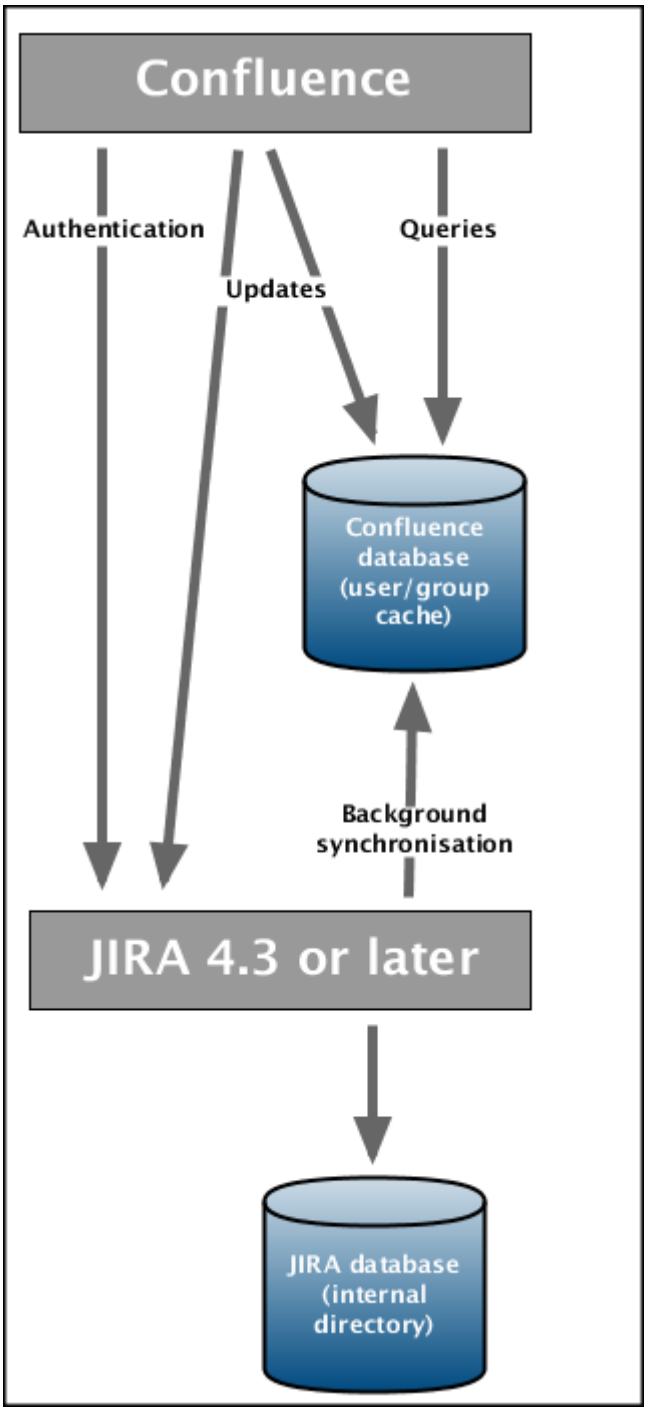


Diagram above: Confluence connecting to JIRA for user management.

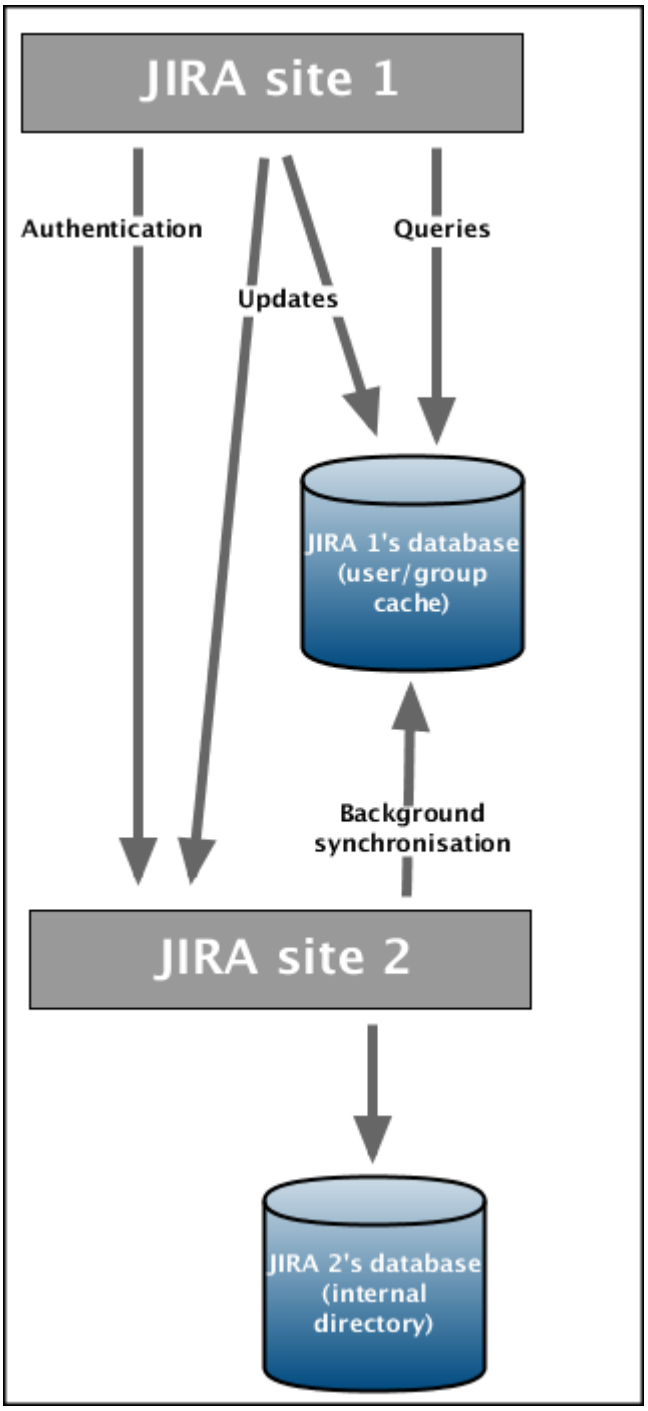


Diagram above: One JIRA site connecting to another for user management. JIRA site 2 does the user management, storing the user data in its internal directory.

RELATED TOPICS

Configuring User Directories

- Configuring the Internal Directory
- Connecting to an LDAP Directory
- Connecting to an Internal Directory with LDAP Authentication
- Connecting to Crowd or Another JIRA Server for User Management
- Managing Multiple Directories
- Synchronising Data from External Directories
- Managing Nested Groups
- Diagrams of Possible Configurations for User Management
- User Management Limitations and Recommendations
- Allowing Other Applications to Connect to JIRA for User Management

Viewing User Sessions

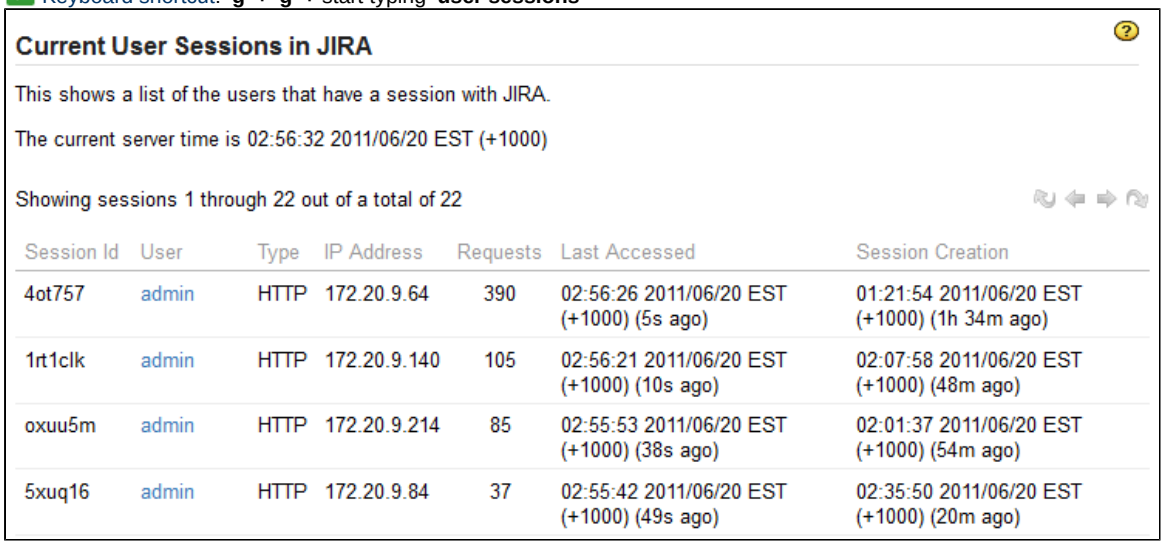
JIRA provides a list of users who are currently accessing JIRA. This is useful if you need to know who to contact before planned downtime, for example.

*The information on this page does not apply to **JIRA OnDemand**.*

Viewing Current User Sessions

To view a list of current JIRA user sessions:

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Security' > 'User Sessions'** (tab) to open the 'Current User Sessions in JIRA' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'user sessions'**



Current User Sessions in JIRA

This shows a list of the users that have a session with JIRA.

The current server time is 02:56:32 2011/06/20 EST (+1000)

Showing sessions 1 through 22 out of a total of 22

Session Id	User	Type	IP Address	Requests	Last Accessed	Session Creation
4ot757	admin	HTTP	172.20.9.64	390	02:56:26 2011/06/20 EST (+1000) (5s ago)	01:21:54 2011/06/20 EST (+1000) (1h 34m ago)
1rt1clk	admin	HTTP	172.20.9.140	105	02:56:21 2011/06/20 EST (+1000) (10s ago)	02:07:58 2011/06/20 EST (+1000) (48m ago)
oxuu5m	admin	HTTP	172.20.9.214	85	02:55:53 2011/06/20 EST (+1000) (38s ago)	02:01:37 2011/06/20 EST (+1000) (54m ago)
5xuq16	admin	HTTP	172.20.9.84	37	02:55:42 2011/06/20 EST (+1000) (49s ago)	02:35:50 2011/06/20 EST (+1000) (20m ago)

The session id shown is also used in the JIRA access log and `atlassian-jira.log`.

It is possible to have "sessions" for computers that are not logged in. For example, when someone accesses JIRA without logging in, a unique session is created without a username (this is shown as 'Not Available' in the 'User' column).

To administer a user, click a username to go to the user's Profile, then select '**Administer User**' from the '**Tools**' menu.

User access logging

Occasionally one wishes to get an overall picture of *which* users are accessing *which* pages in JIRA. Application servers are able to log the requested URL, but (it seems) they cannot determine the currently logged in user (probably because they run before the Seraph filter has a chance to set request.getRemoteUser()).

Similar to Confluence, JIRA 3.3 and above has a built-in URL logging mechanism, which shows the user and URL invoked:

Please note, as of JIRA 4.1, some of this functionality is now available via the Administration interface. See Viewing User Sessions for more details.

On this page:

- Usage in JIRA 3.x
- Usage in JIRA 4.x
- Application Server Access Logs
- Related pages

*The information on this page does not apply to **JIRA OnDemand**.*

```
2005-07-15 18:10:19,276 INFO [JIRACOM:jira.web.filters.AccessLogFilter] AccessLogFilter
initialized. Format is: <user> <url> <starting memory free (kb)> +- <difference in free mem
(kb)> <query time (ms)>
2005-07-15 18:10:20,523 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/secure/admin/jira/ViewLogging.jspa 109687-110 35
2005-07-15 18:10:20,523 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/secure/admin/jira/ViewLogging.jspa 109687-110 35
2005-07-15 18:15:59,709 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/secure/IssueNavigator.jspa 115762-3000 2385
2005-07-15 18:15:59,709 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/secure/IssueNavigator.jspa 115762-3000 2385
2005-07-15 18:26:53,916 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/logout 108756 1
2005-07-15 18:26:53,916 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/logout 108756 1
2005-07-15 18:26:54,306 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/secure/Logout!default.jspa 116660-2809 279
2005-07-15 18:26:54,306 INFO [JIRACOM:jira.web.filters.AccessLogFilter] joe
http://localhost:8090/jira/secure/Logout!default.jspa 116660-2809 279
2005-07-15 18:27:01,380 INFO [JIRACOM:jira.web.filters.AccessLogFilter] -
http://localhost:8090/jira/default.jsp 116393-512 42
2005-07-15 18:27:01,380 INFO [JIRACOM:jira.web.filters.AccessLogFilter] -
http://localhost:8090/jira/default.jsp 116393-512 42
2005-07-15 18:27:01,472 INFO [JIRACOM:jira.web.filters.AccessLogFilter] -
http://localhost:8090/jira/secure/Dashboard.jspa 115881-809 87
2005-07-15 18:27:01,472 INFO [JIRACOM:jira.web.filters.AccessLogFilter] -
http://localhost:8090/jira/secure/Dashboard.jspa 115881-809 87
```

Here you can see user 'joe' enable access logging, then log out, and view the dashboard anonymously.

Usage in JIRA 3.x

URL logging is disabled in JIRA by default. To turn it on, go to **Admin -> System -> Logging & Profiling**, and change the log level for AccessLogFilter from `WARN` to `INFO`. To make this change permanent, you would need to edit the corresponding section in `WEB-INF/classes/log4j.properties` on disk, changing:

```
log4j.category.com.atlassian.jira.web.filters.AccessLogFilter = WARN, console, filelog
log4j.additivity.com.atlassian.jira.web.filters = false
```

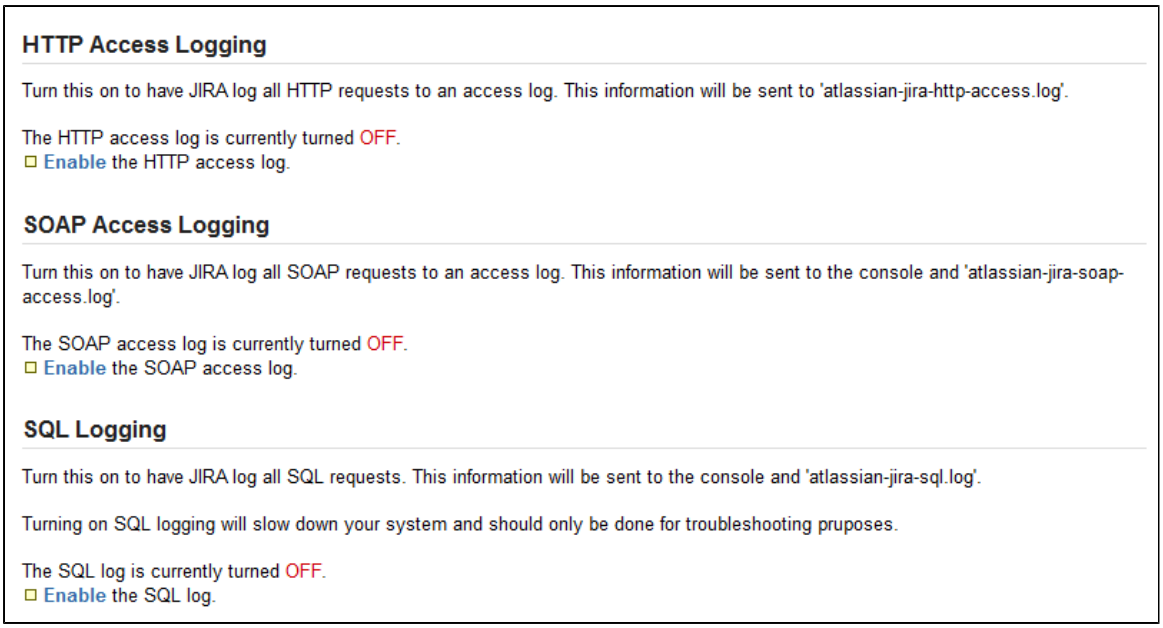
to:

```
log4j.category.com.atlassian.jira.web.filters.AccessLogFilter = INFO, console, filelog
log4j.additivity.com.atlassian.jira.web.filters = false
```

and then restart JIRA.

Usage in JIRA 4.x

URL logging is disabled in JIRA by default. To turn it on, go to **'Administration' > 'System' > 'Troubleshooting and Support' > 'Logging & Profiling'** and click the **'Enable'** link on 'HTTP Access Logging' and 'SOAP Access Logging'.



Note, the user access logs are not outputted to the atlassian-jira.log file, the http logs are written to `<JIRA Data folder>/log/atlassian-jira-http-access.log` and SOAP logs are written to `<JIRA Data folder>/log/atlassian-jira-soap-access.log`

Application Server Access Logs

JIRA's application server (Apache Tomcat) can also produce access logs. These are enabled by default in JIRA and result in `logs/access_log.<date>.log` files being generated in your JIRA Installation Directory. If you are using JIRA WAR, this feature may be enabled in the `conf/server.xml` file (of the Tomcat application server installation running JIRA) by adding the following line before `</Context>`:

```
<Valve className="org.apache.catalina.valves.AccessLogValve" pattern="%h %l %u %t
&amp;quot;%r&amp;quot; %s %b %T %S" resolveHosts="false" />
```

You will need to restart JIRA for the changes to take effect.

The Apache Tomcat Access Log Valve documentation describes each of the above parameters.

This will generate logs that include the IP address, like:

```
127.0.0.1 - - [19/Oct/2006:12:38:09 +0800] "GET / HTTP/1.1" 302 - 0.240 -
127.0.0.1 - - [19/Oct/2006:12:38:09 +0800] "GET / HTTP/1.1" 302 - 0.243 -
127.0.0.1 - - [19/Oct/2006:12:38:11 +0800] "GET /secure/Setup!default.jspa HTTP/1.1" 302 -
1.732 39F9C7F048F7F060A4B9883A7348082D
127.0.0.1 - - [19/Oct/2006:12:38:16 +0800] "GET
/secure/Setup!input.jspa?title=Your+Company+JIRA&mode=public HTTP/1.1" 200 24678 4.885
39F9C7F048F7F060A4B9883A7348082D
127.0.0.1 - - [19/Oct/2006:12:38:24 +0800] "GET /styles/global.css HTTP/1.1" 200 548 0.213
39F9C7F048F7F060A4B9883A7348082D
127.0.0.1 - - [19/Oct/2006:12:38:26 +0800] "GET
/secure/Setup!input.jspa?title=Your+Company+JIRA&mode=public HTTP/1.1" 200 24678 0.443
39F9C7F048F7F060A4B9883A7348082D
127.0.0.1 - - [19/Oct/2006:12:38:26 +0800] "GET /styles/global.css HTTP/1.1" 200 548 0.001
39F9C7F048F7F060A4B9883A7348082D
127.0.0.1 - - [19/Oct/2006:12:38:28 +0800] "GET /includes/js/combined-javascript.js HTTP/1.1"
200 65508 1.712 39F9C7F048F7F060A4B9883A7348082D
127.0.0.1 - - [19/Oct/2006:12:38:28 +0800] "GET /includes/js/combined-javascript.js HTTP/1.1"
200 65508 4.386 39F9C7F048F7F060A4B9883A7348082D
```

Related pages

No content found for label(s) logging.

Clearing 'Remember my login' Tokens

Introduction

When a user logs in to a JIRA site, they have the option of making JIRA remember their login on a specific computer and browser, by selecting the '**Remember my login...**' check box before they click the '**Log In**' button. Upon doing so, a 'Remember my login' token is stored by the JIRA server and a cookie containing this token is set in the user's browser.

A user who revisits a JIRA site from the same computer and browser, will automatically be logged in *if* JIRA detects that one of the user's 'Remember my login' tokens has a matching token contained in one of that browser's cookies. If the user logs out of JIRA, the 'Remember my login' token (which matches the relevant browser cookie) is cleared from the JIRA server.

To maximise and maintain the security of your JIRA site, JIRA provides features for:

- clearing 'Remember my login' tokens associated with individual user accounts and
- clearing *all* 'Remember my login' tokens stored by your JIRA site.

These features are especially useful in situations where users have been accessing your JIRA site in a public environment, selected the '**Remember by login...**' check box before logging in, but have forgotten to log out.

On this page:

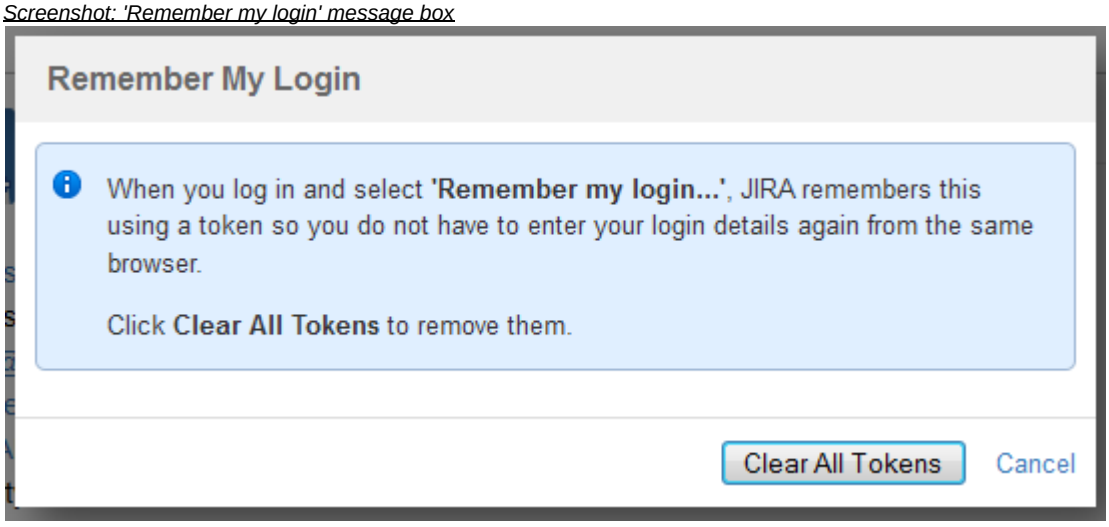
- Introduction
- Clearing 'Remember my login' Tokens from a User Profile
- Clearing a User's 'Remember my login' Tokens from the Administration Console
- Clearing *all* 'Remember my login' Tokens from the JIRA site

If you are a JIRA administrator who wishes to disable this feature from your JIRA site, please refer to Disabling Remember My Login on this Computer.

Clearing 'Remember my login' Tokens from a User Profile

A JIRA user can clear all of their own 'Remember my login' tokens from JIRA through their user profile. To do this:

1. Visit your User Profile.
2. In the **Details** section, click the '**Clear All Tokens**' link. The **Remember my login** message box appears.
   Screenshot: 'Remember my login' message box

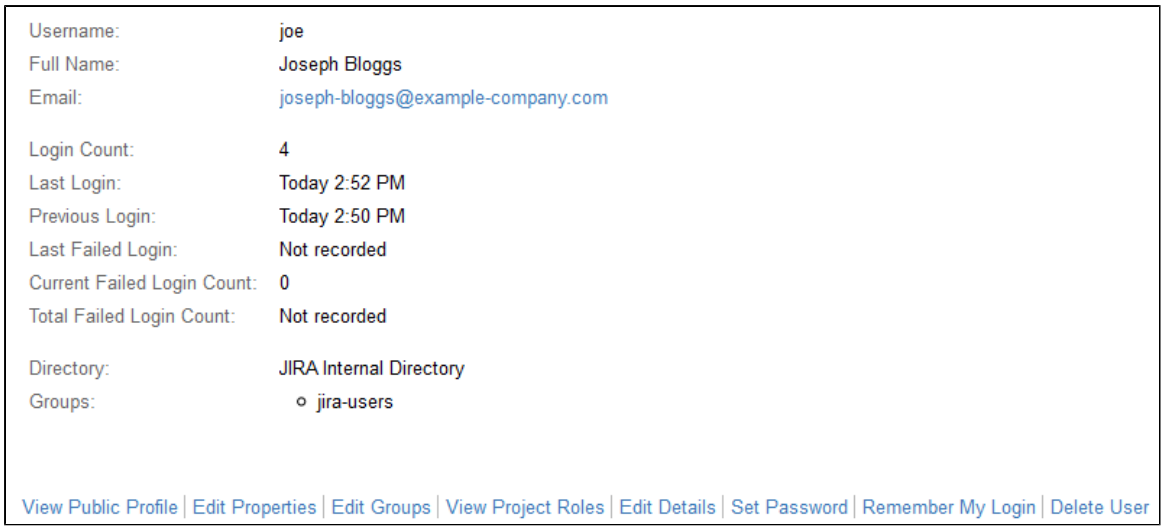


3. Click the '**Clear All Tokens**' button. All tokens associated with your user account will be removed from the JIRA server.

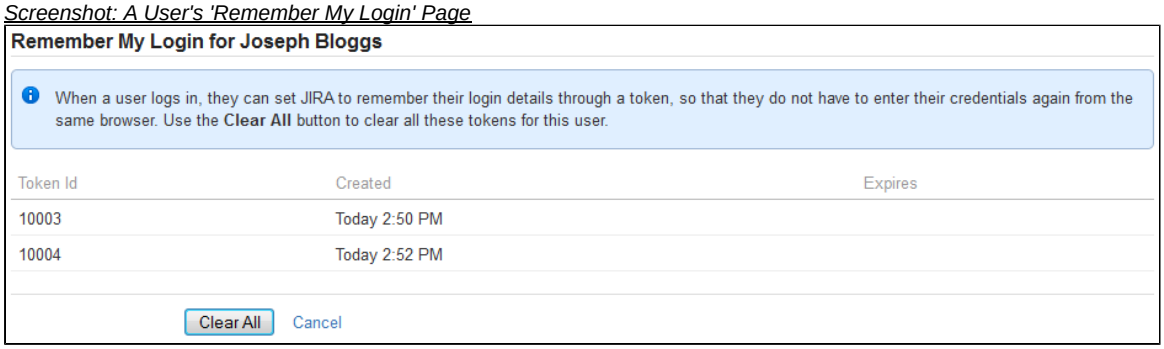
Clearing a User's 'Remember my login' Tokens from the Administration Console

JIRA administrators can clear all 'Remember my login' tokens associated with a user's account through the JIRA administration console. To do this:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Users' > 'Users'** to open the 'User Browser' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'users'**
3. Click the **Username** or **Email Address** of the user whose 'Remember my login' tokens you wish to remove. Details about that user and their login information is displayed.
   To restrict the list of users shown in the **User Browser**, use the Filter form at the top of the User Browser. Specifying (part of) the user's email and/or group, then clicking the '**Filter**' button, will reduce the list to only those users who match those criteria.
4. Click the '**Remember My Login**' link to display that user's **Remember My Login** page.
   Screenshot: 'Remember My Login' link on the User Administration Page



Screenshot: A User's 'Remember My Login' Page



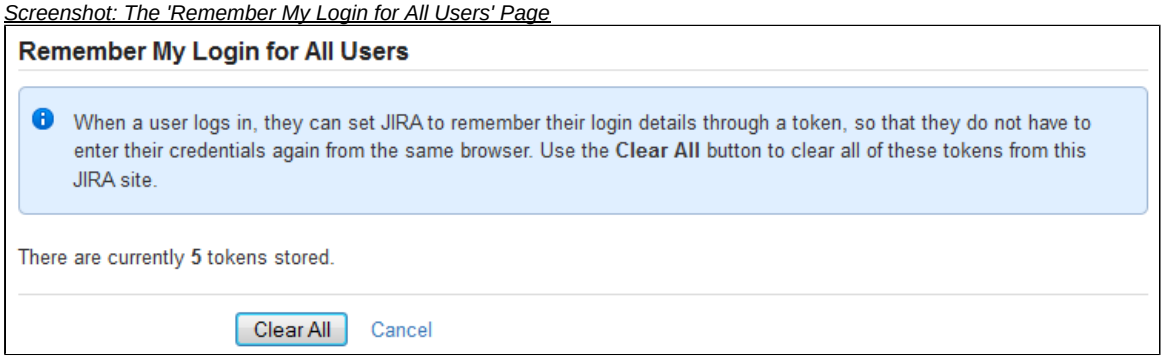
5. Click the '**Clear All**' button to remove all 'Remember my login' tokens associated with this user account from the JIRA server.

Clearing *all* 'Remember my login' Tokens from the JIRA site

JIRA administrators can also clear all 'Remember my login' tokens from their JIRA site through JIRA's administration console. To do this:

1. Log in as a user with the **JIRA Administrators** global permission.
2. # Select **'Administration' > 'System' > 'Security' > 'Remember My Login'** to open the 'Remember My Login for All Users' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'remember my login'**

Screenshot: The 'Remember My Login for All Users' Page



3. Click the '**Clear All**' button to remove all 'Remember my login' tokens from the JIRA server.

Disabling Remember My Login on this Computer

To remove the **Remember my login on this computer** option on the login page, follow the instructions below:

Option 1 (recommended)
The check box for this option can be disabled by setting the `jira.option.allowcookies` property to `false` in your jira-config.properties file.
You will need to restart JIRA in order for this change to take effect.

Option 2
Edit the `./atlassian-jira/includes/loginform.jsp` file.

Enabling Public Signup and CAPTCHA

About Public Signup and CAPTCHA

For some organisations it is appropriate to enable *signup,* which allows visitors to immediately create their own JIRA user accounts. If signup is not enabled, then only a JIRA administrator can create new user accounts.

For example, enabling signup can be useful if you are using JIRA as a support system and have a very large number of potential users, of which only some will need to log support tickets.

> For security reasons, even if you enable signup, it is still necessary for users to have the appropriate project permissions before they can see or create issues. Note that you can use automatic group membership to add all new users to appropriate groups.

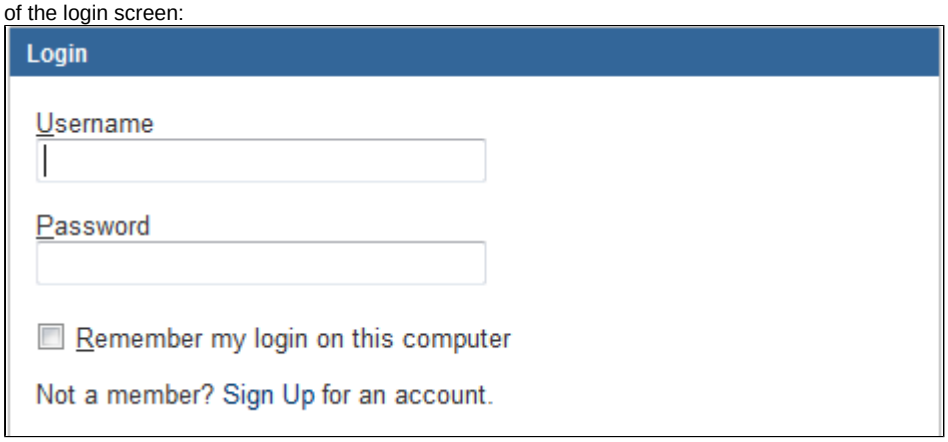
If your JIRA server is accessible from outside your organisation's firewall, and you have enabled signup, then you may want to also enable *CAPTCHA.* CAPTCHA helps ensure that only real humans (and not automated spam systems) can sign themselves up to JIRA. When CAPTCHA is enabled, visitors will need to recognise a distorted picture of a word (see example below), and must type the word into a text field. This is easy for humans to do, but very difficult for computers.

On this page:

- About Public Signup and CAPTCHA
- Enabling Public Signup
- Enabling CAPTCHA

Enabling Public Signup

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'General Configuration'** to open the 'General Configuration' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'general configuration'**
3. Click '**Edit Configuration**' at the end of the page.
4. In the 'Mode' drop-down, select **'Public'**.
5. Click the **'Update'** button at the bottom of the screen.
6. Log out of JIRA, then click the '**Log In**' link at the top right of the screen and verify that the **'Sign Up'** link is displayed at the bottom of the login screen:



Enabling CAPTCHA

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'General Configuration'** to open the 'General Configuration' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'general configuration'**
3. Click '**Edit Configuration**' at the end of the page.
4. Locate 'CAPTCHA on signup' and select **'On'**.
5. Click the **'Update'** button at the bottom of the screen.
6. Log out of JIRA, click the '**Log In**' link at the top right of the screen, then click the '**Sign Up**' link and verify that a random sequence of letters is displayed at the bottom of the **'Sign Up'** screen — e.g. "winzers" in the following screenshot:

Project Management

- Defining a Project
- Managing Project Role Membership
- Defining a Component
- Managing Versions
  - Running a Bamboo Build when Releasing a Version
  - Creating Release Notes
- Configuring Project Keys

Defining a Project

Creating a Project

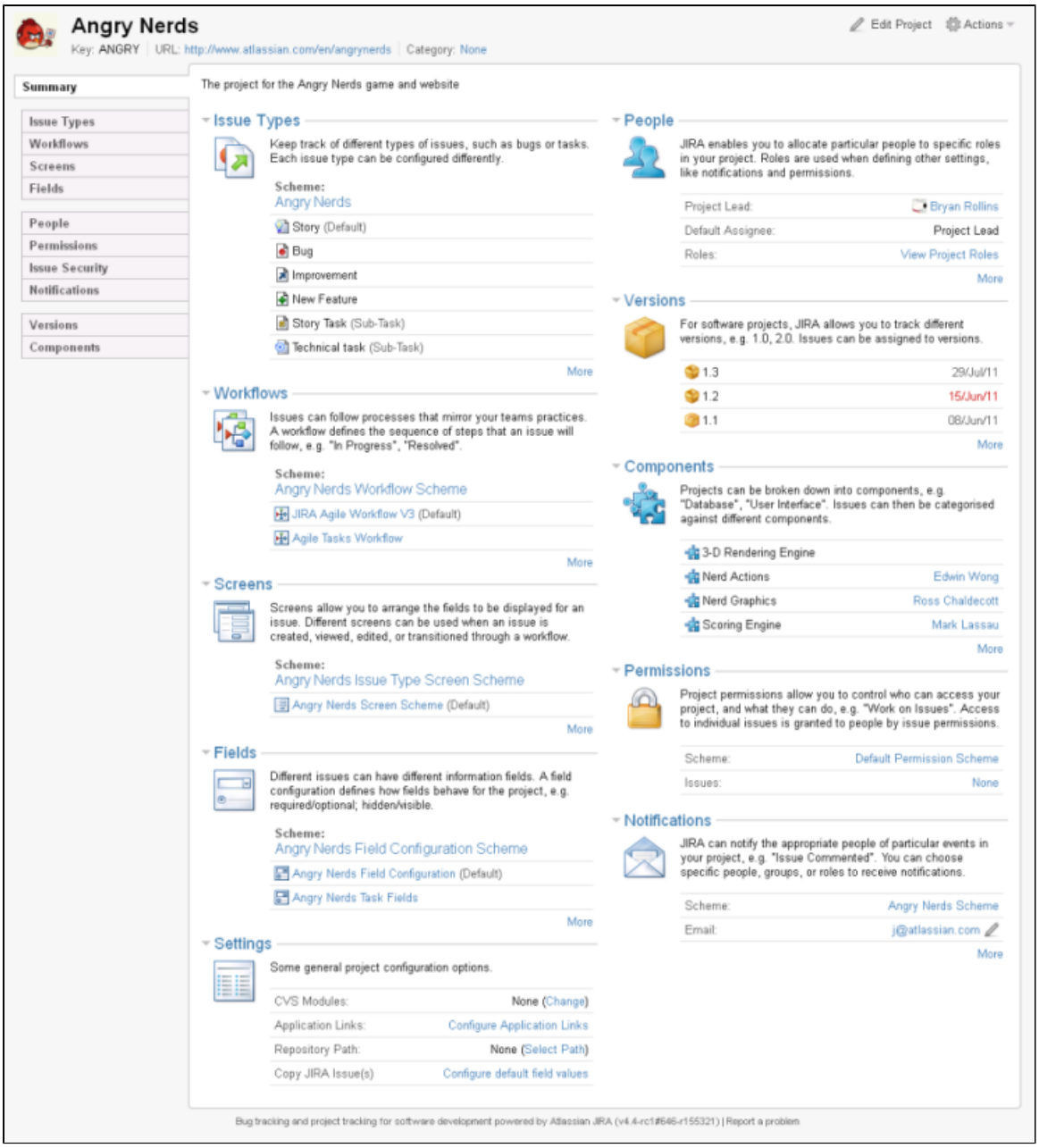
To add a new project in JIRA:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click the '**Administration**' link at the top of the screen.
3. Select **'Projects' > 'Projects'** then click **'Add Project'** to display the 'Add Project' screen (see Screenshot 2below).
   - **Name** — type a descriptive name. This can be changed later if you wish.
   - **Key** — type a 'key' unique to this project (eg. 'WEB'). This will be used as the prefix of this project's issue keys (e.g. 'WEB-100'). We recommend that you define a key that describes the project and is easy to type. Please note that the key cannot be changed once the project is created.
   - **Project Lead** — choose the person who will manage this project. You can change the Project Lead later if you wish. Note that issues can be automatically assigned to the Project Lead (see 'People' below).

     If there is only one user in your JIRA system, the Project Lead will default to that person and this field will not be available.
4. Once created, the 'Project Summary' screen for your new project will be displayed (see Screenshot 1 below). You can then configure other details of your project as described below.

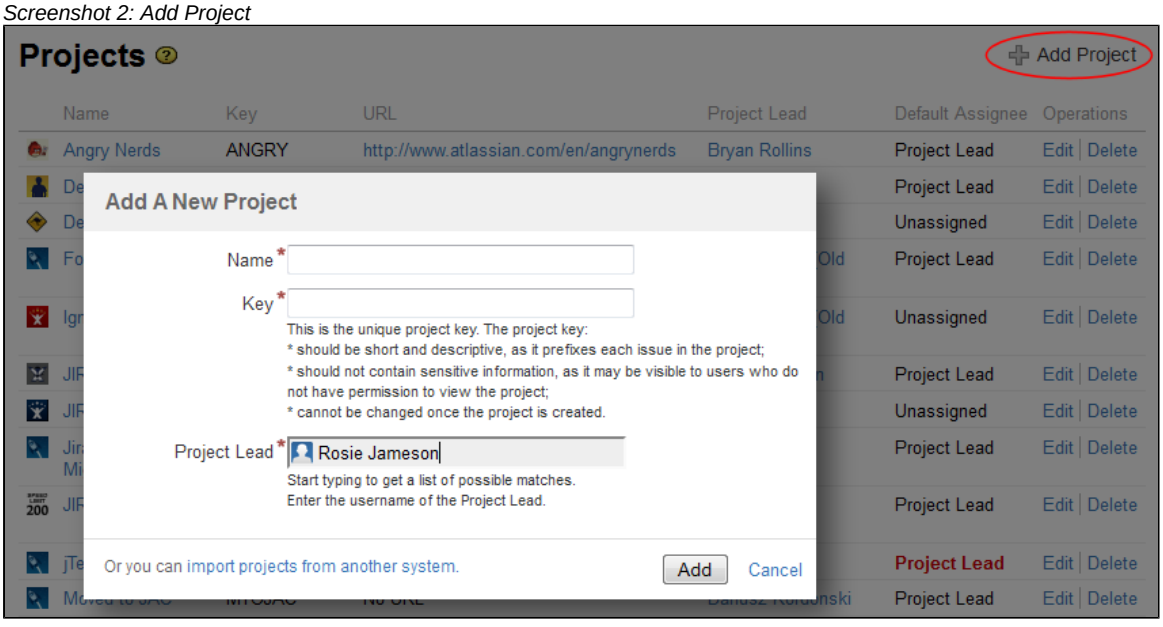
   Here is what a project looks like once created:

   Screenshot 1: Project Summary



(View full size)

Screenshot 2: Add Project



On this page:

- Creating a Project
- Configuring a Project
  - Project Details
  - Issue Types
  - Workflows
  - Screens
  - Fields
  - Settings
  - People
  - Versions
  - Components
  - Permissions
  - Notifications
- A note about Project Administrators

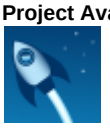
Configuring a Project

To configure a project in JIRA:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click the '**Administration**' link at the top of the screen.
3. Click **'Projects'** and select the project of interest. The 'Project Summary' screen will be displayed (see Screenshot 1 above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'project'**

You can then edit the project's configuration settings as follows:

Project Details

- **Name** — type a descriptive name. This can be changed later if you wish.
- **URL** — an optional URL associated with this project, eg. pointing to project documentation.
- **Project Avatar** — an image (48x48 pixels) that represents the project. You can either use the default image, i.e.:
  
  or choose a different image. If you prefer not to use an image, simply upload a transparent pixel.
- **Description** — an optional description of this particular project. You can include HTML, but make sure all your tags are closed.
  Warning: Please be aware that this is completely unfiltered HTML and as such, it is susceptible to cross site scripting attacks.

Click the link next to the 'Category' field under the project name to assign the project into a logical category/group. This is useful for managing multiple related projects. If no categories exist, click the '**Add**' link on the following page to add a new category. New categories can also be created via **'Administration' > 'Projects' > 'Project Categories'**.

Issue Types

JIRA enables you to keep track of different types of things — bugs, tasks, helpdesk tickets, etc — by using different *issue types*. You can also configure each issue type to act differently, e.g. to follow a different process flow or track different pieces of information.

- **Issue Type Scheme** — the project's issue type scheme determines which issue types apply to this project.

Workflows

Your JIRA issues can follow a process that mirrors your team's practices. A *workflow* defines the sequence of steps (or statuses) that an issue will follow, e.g. Open, In Progress, Resolved. You can configure how issues will transition between statuses, e.g. who can transition them, under what conditions, and which screen will be displayed for each transition.

- **Workflow Scheme** — the project's workflow scheme determines which workflows (issue state transitions) apply to issue types in this project.

Screens

JIRA allows you to display particular pieces of issue information at particular times, by defining *screens*. A screen is simply a collection of fields. You can choose which screen to display when an issue is being created, viewed, edited, or transitioned through a particular step in a workflow.

- **Screen Scheme** — the project's screen scheme determines which screens are displayed for different issue operations (view, edit, create);

OR
Issue Type Screen Scheme — the project's issue type screen scheme determines which screens are displayed for different issue operations (view, edit, create), for different issue types.

Fields

JIRA enables you to define field behaviour: each field can be required/optional, rich text/plain text, hidden/visible. You define this behaviour by using a *field configuration*.

- **Field Configuration Scheme** — the project's field configuration scheme determines which field configuration applies to issue types in this project. (A field configuration determines each field's overall visibility, requiredness, formatting (wiki/rich-text or plain) and help-text).

Settings

- **CVS Modules** — configures CVS integration for this project.
- **Application Links** — projects or other entities on other applications or sites to which this JIRA project has been linked via application links. New project/entity links can be created by clicking the '**Configure Application Links**' link. See Adding Project Links between Applications for details.

People

Different people may play different roles in different projects — the same person may be a leader of one project but an observer of another project. JIRA enables you to allocate particular people to specific roles in your project.

- **Project Lead** — user fulfilling the role of project leader. Used as the 'Default Assignee' (see below), and potentially elsewhere in JIRA (e.g. in permission schemes, notification schemes, issue security schemes and workflows).
- **Default Assignee** — the user to whom issues in this project are initially assigned when created. Can be either the 'Project Lead' (above), or, if **Allow unassigned issues** is set to 'On' in JIRA's general configuration, 'Unassigned'. There are also default component assignees.
- **Project Roles** — members are users/groups who fulfil particular functions for this project. Project roles are used in permission schemes, notification schemes, issue security schemes and workflows.

Versions

If you are using JIRA to manage the development of a product, you may want to define different *versions* to help you track which issues relate to different releases of your product (e.g. 1.0, 1.1, 1.2, 2.0 beta, 2.0). JIRA can help you manage, release and archive your versions. Versions can also have a Release Date, and will automatically be highlighted as "overdue" if the version is unreleased when this date passes.

- **Versions** — versions defined in the project. See the version management page for details.

Components

You may want to define various *components* to categorise and manage different issues. For a software development project, for example, you might define components called "Database", "Usability", "Documentation" (note that issues can belong to more than one component). You can choose a Default Assignee for each component, which is useful if you have different people leading different sub-teams in your project.

- **Components** — logical groups that this project's issues can belong to. See the component management page for details.

Permissions

JIRA allows you to control who can access your project, and exactly what they can do (e.g. "Work on Issues", "Comment on Issues", "Assign Issues"), by using *project permissions*. You can also control access to individual issues by using *security levels*. You can choose to grant access to specific users, or groups, or roles (note that roles are often the easiest to manage).

- **Permission Scheme** — the project's permission scheme determines who has permission to view or change issues in this project.
- **Issue Security Scheme** — the project's issue security scheme determines what visibility levels issues in this project can have (see issue-level security).

Notifications

JIRA can notify the appropriate people when a particular event occurs in your project (e.g. "Issue Created", "Issue Resolved"). You can choose specific people, or groups, or roles to receive *email notifications* when different events occur. (Note that roles are often the easiest to manage.)

- **Notification Scheme** — the project's notification scheme determines who receives email notifications of changes to issues in this project.

- **Email** — specifies the 'From' address for emails sent from this project. Only available if an SMTP email server has been configured in JIRA.

A note about Project Administrators

A JIRA project administrator is someone who has the project-specific 'Administer Project' permission, but not necessarily the global 'JIRA Administrator' permission.

A project administrator can:

- Edit the project name
- Edit the project description
- Edit the project avatar image
- Edit the URL
- Edit the Project Lead
- Edit project role membership
- Configure entities for application links
- Define project components
- Define project versions
- View — but not edit — the project's schemes (notification scheme, permission scheme, etc)

Managing Project Role Membership

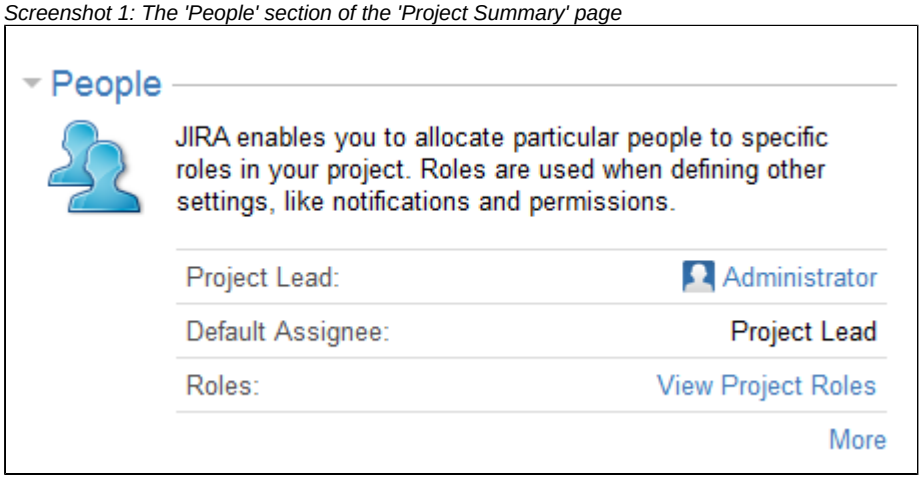
A JIRA project role is a flexible way to associate users and/or groups with a particular project.

Unlike groups, which have the same membership throughout JIRA, project roles have specific members for each project. Users may play different roles in different projects.

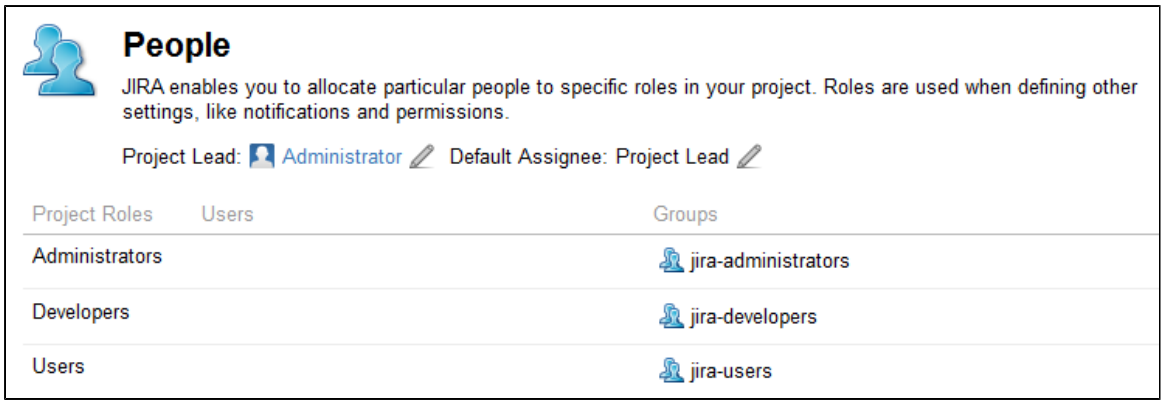
This page contains instructions for managing membership of *existing project roles*. For information on creating and using project roles, please see Managing project roles.

Viewing project role members

To see which users and groups belong to each project role for a particular project:

1. Log in to JIRA as a project administrator. (A project administrator is someone who has the **Administer Project** project-specific permission, but not necessarily the **JIRA Administrators** global permission.)
2. Click the **Administration** link at the top of the page.
3. Click **Projects** and select the project of interest from the dropdown menu. The 'Project Summary' page (see Defining a Project) for your selected project is shown.
   Keyboard shortcut: **g + g +** start typing **project**
4. Locate the **People** section at the right of the 'Project Summary' page (see Screenshot 1 below) and click the **View Project Roles** link to display the **People** page (see Screenshot 2 below).
   Screenshot 1: The 'People' section of the 'Project Summary' page
   

5. The **People** page is displayed, from where you can manage the project role membership as described below.
   Screenshot 2: The 'People' page



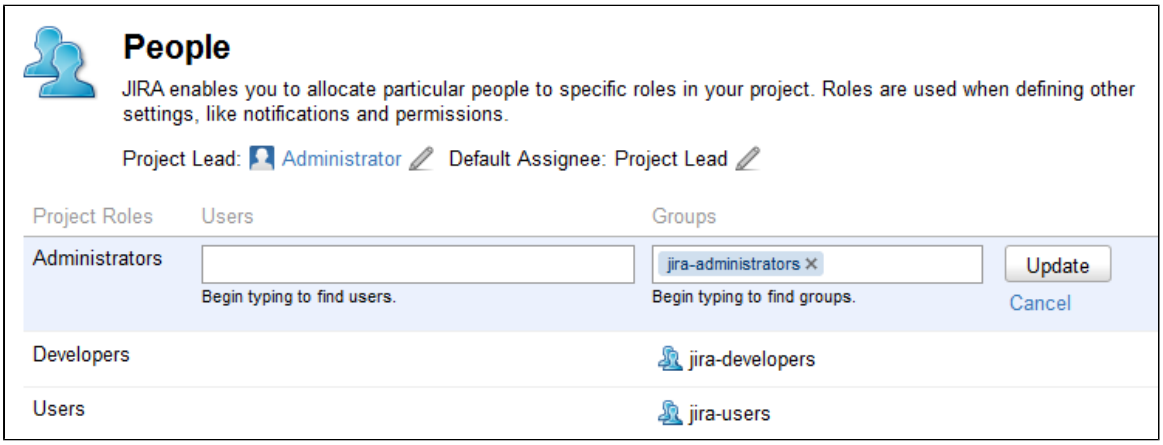
On this page:

- Viewing project role members
- Assigning a user to a project role
- Removing a user from a project role
- Assigning a group to a project role
- Removing a group from a project role

Assigning a user to a project role

1. Open the **People** page (see Screenshot 2) as described in 'Viewing project role members' above.
2. Hover over the **Users** column for the project role in which you are interested (e.g. **Administrators**) and click the yellow box which appears.
3. The users and groups will become editable (see Screenshot 3). Type the name of the user(s) you wish to add to this project role. Then click the **Update** button.

Screenshot 3: Editing project role membership



Removing a user from a project role

1. Open the **People** page (see Screenshot 2) as described in 'Viewing project role members' above.
2. Hover over the **Users** column for the project role in which you are interested (e.g. **Administrators**) and click the yellow box which appears.
3. The users and groups will become editable (see Screenshot 3). Click the **x** next to the name of the user(s) you wish to remove from this project role. Then click the **Update** button.

Assigning a group to a project role

1. Open the **People** screen (see Screenshot 2) as described in 'Viewing project role members' above.
2. Hover over the **Groups** column for the project role in which you are interested (e.g. **Administrators**) and click the yellow box which appears.
3. The users and groups will become editable (see Screenshot 3). Type the name of the group(s) you wish to add to this project role. Then click the **Update** button.

Group membership can only be viewed/edited by people with the **JIRA Administrator** global permission. Project administrators may therefore prefer to assign users, rather than groups, to their project roles.

Removing a group from a project role

1. Open the **People** page (see Screenshot 2) as described in 'Viewing project role members' above.
2. Hover over the **Groups** column for the project role in which you are interested (e.g. **Administrators**) and click the yellow box which appears.
3. The users and groups will become editable (see Screenshot 3). Click the **x** next to the name of the group(s) you wish to remove from this project role. Then click the **Update** button.

A project role need not have any users or groups assigned to it, although project administrators should be careful with this. Depending on how a project role is used (e.g. if the project's permission scheme is using project roles), it is possible that not having anyone in a particular project role could make some project activities unavailable.

Defining a Component

Components are sub-sections of a project. They are used to group issues within a project into smaller parts.

Note that you can set a Default Assignee for a component. This will override the project's default assignee, for issues in that component.

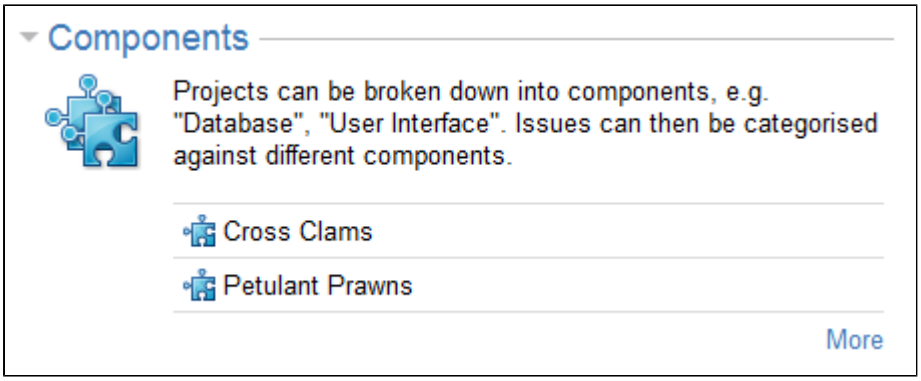
Managing a project's components

1. Login to JIRA as a project administrator. (A project administrator is someone who has the project-specific permission **'Administer Project',** but not necessarily the global permission **'JIRA Administrators'**.)
2. Click the '**Administration**' link at the top of the screen.
3. Click **'Projects'** and select the project of interest.
   Keyboard shortcut: **'g' + 'g' +** start typing **'project'**
4. You will now see the 'Project Summary' screen (see Defining a Project). On the lower right, a summary of the components is displayed (see Screenshot 1 below).
5. Click '**More**' to display the 'Components' screen (see Screenshot 2 below), which shows a list of components. From here you can manage the project's components as described below.

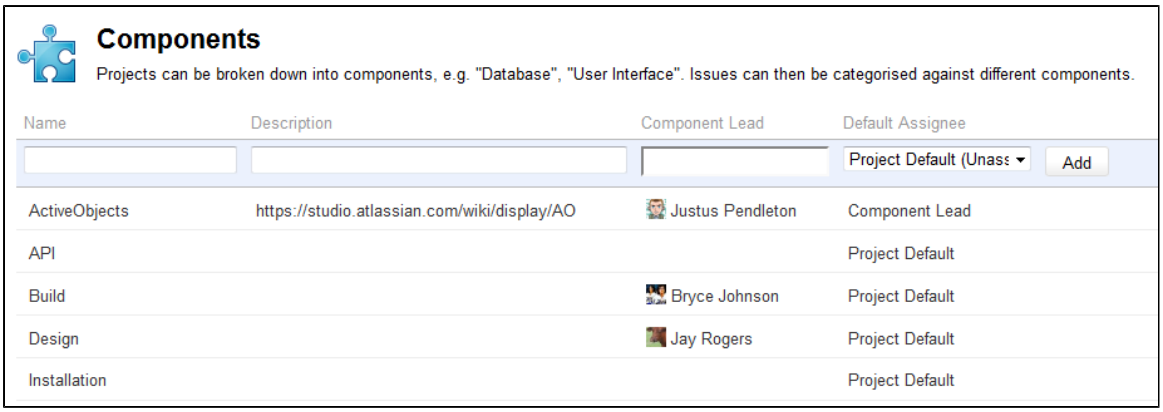
On this page:

- Managing a project's components
- Adding a new component
  - Selecting a Default Assignee
- Editing a component's details
- Deleting a component

Screenshot 1: The 'Components' section of the Project Summary screen



Screenshot 2: The 'Components' screen



Adding a new component

1. The 'Add Component' form is located at the top of the 'Components' screen (see Screenshot 2 above).
2. Enter the **Name** for the component. Optionally enter a **Description** and select a **Component Lead** and **Default Assignee** (see options below).

Selecting a Default Assignee

You can optionally set a Default Assignee for a component. This will override the project's default assignee, for issues in that component.
If an issue has multiple components, and the default assignees of components clash, the assignee will be set to the default assignee of the component that is first alphabetically.

Default Assignee Option	Description	Notes
Project Default	Issues matching this component will have the assignee set to the same default assignee as the parent project.	
Project Lead	The assignee will be set to the project leader.	If the project leader is not permitted to be assigned to issues in the permission scheme this option will be disabled and will say "Project Lead is not allowed to be assigned issues".
Component Lead	The assignee will be set to the component leader.	If the component leader is not permitted to be assigned to issues in the permission scheme this option will be disabled and will say "Component Lead is not allowed to be assigned issues". The Component Lead option will also not be available if the component does not have a lead assigned to the component. Instead under this option it will say "Component does not have a lead.".
Unassigned	The assignee of the issue will not be set on the creation of this issue.	This option will only be available if "Allow unassigned issues" is enabled in the General Configuration.

Editing a component's details

1. On the 'Components' screen (see Screenshot 2 above), hover over the relevant component to display the pencil icon.
2. This will allow you to edit the component's Name, Description, Lead and Default Assignee.
3. Click the **'Update'** button to save your changes.

Deleting a component

1. On the 'Components' screen (see Screenshot 2 above), hover over the relevant component to display the '**Delete**' button.
2. You will be prompted to associate these issues with another component if you wish.

Managing Versions

Versions are points-in-time for a project. They help you schedule and organise your releases. Once a version is created, and issues are assigned to it, the following reports are useful:

- Road Map report — gives you a view of upcoming versions

- Change Log report — gives you a review of released versions

The Change Log and Road Map reports are driven by the 'Fix For Version' field on each issue.

Versions can be:

- Added — create a new version against which issues can be aligned.
- Released — mark a version as released. This changes the Road Map report, Change Log report and some issue fields' drop-downs If you have integrated JIRA with Bamboo, you can also trigger builds when releasing a version.
- Rescheduled — re-arrange the order of versions.
- Archived — hide an old version from the Road Map and Change Log reports, and in the JIRA User Interface.
- Merged — combine multiple versions into one.

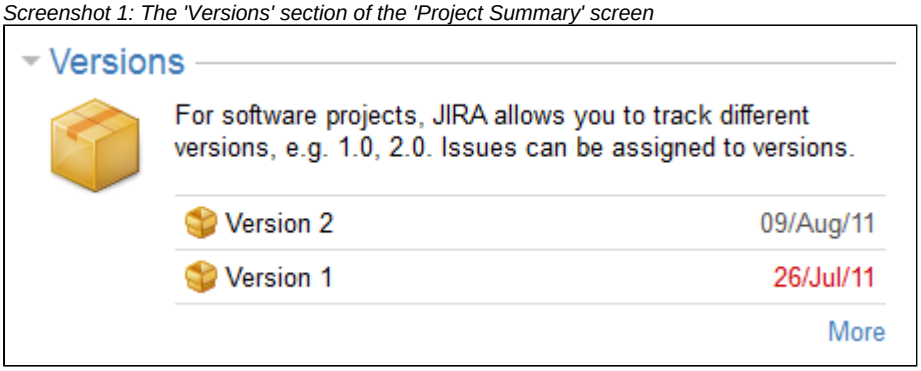
On this page:

- Managing a project's versions
  - Version status
- Adding a new version
- Releasing a version
- Archiving a version
- Merging multiple versions
- Editing a version's details
- Deleting a version
- Rescheduling a version
- See also

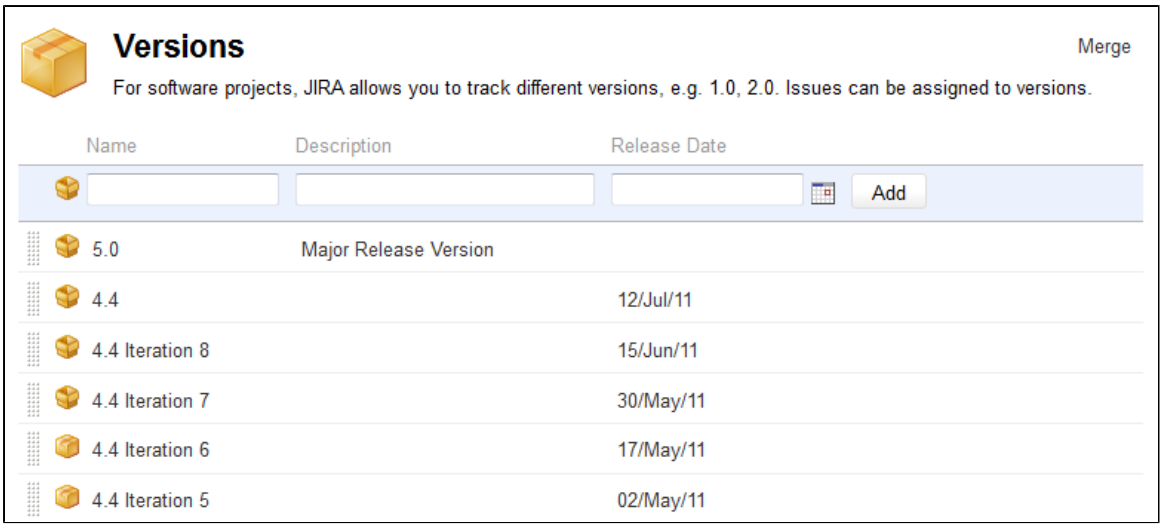
Managing a project's versions

1. Log in to JIRA as a project administrator.
   A project administrator is someone who has the project-specific permission **'Administer Project',** but not necessarily the global permission **'JIRA Administrators'**.
2. Click the '**Administration**' link at the top of the screen.
3. Click **'Projects'** and select the project of interest.
   Keyboard shortcut: **'g' + 'g' +** start typing **'project'**
4. You will now see the 'Project Summary' screen (see Defining a Project). On the lower right, a summary of the versions is displayed, showing each version's status and the scheduled release date for that version (see Screenshot 1 below).
5. Click '**More**' to display the 'Versions' screen (see Screenshot 2 below), which shows a list of versions and each version's status. From here you can manage the project's versions as described below.

Screenshot 1: The 'Versions' section of the 'Project Summary' screen



Screenshot 2: The 'Versions' screen



Version status

Each version can have any of the following four statuses:

- **Released** — a bundled package
- **Unreleased** — an open package
- **Archived** — a semi-transparent package
- **Overdue**— the release date is highlighted

> The status affects where the version appears in drop-down lists for version-related issue fields ('Fix For Version' and 'Affects Version').

Adding a new version

1. The 'Add Version' form is located at the top of the 'Versions' screen (see Screenshot 2 above).
2. Enter the name for the version. The name can be:
   - simple numeric, e.g. "2.1", or
   - complicated numeric, e.g. "2.1.3", or
   - a word, such as the project's internal code-name, e.g. "Memphis".
3. Optional details such as the version description and release date (i.e. the planned release date for a version) can be also be specified.
4. Click the '**Add**' button. The version management list is updated immediately, with the newly created version added at the top of the list — you can drag it to a different position if you wish, by hovering over the 'drag' icon at the left of the version name:

Releasing a version

Before you begin: If you have integrated JIRA with Atlassian's Bamboo, you can trigger a Bamboo build to run automatically when releasing a version of JIRA. The JIRA version will only be released if the build is successful. See these alternate instructions: Running a Bamboo Build when Releasing a Version.

1. On the 'Versions' screen (see Screenshot 2 above), hover over the relevant version to display the cog icon, then select '**Release**' from the drop-down menu.
2. If there are any issues set with this version as their 'Fix For' version, JIRA allows you to choose to change the 'Fix For' version if you wish. Otherwise, the operation will complete without modifying these issues.

To revert the release of a version, simply select '**Unrelease**' from the drop-down menu.

Archiving a version

1. On the 'Versions' screen (see Screenshot 2 above), hover over the relevant version to display the cog icon, then select '**Archive**' from the drop-down menu.
2. The version list indicates the version 'archived' status with a semi-transparent icon. The list of available operations is replaced with the 'Unarchive' operation. No further changes can be made to this version unless it is un-archived. Also it is not possible to remove any existing archived versions from an issue's affected and fix version fields or add any new archived versions.

To revert the archive of a version, simply select '**Unarchive**' from the drop-down menu.

Merging multiple versions

Merging multiple versions allows you to move the issues from one or more versions to another version.

1. On the 'Versions' screen (see Screenshot 2 above), click the '**Merge**' link at the top right of the screen.
2. The 'Merge Versions' popup will be displayed. On this page are two select lists — both listing all un-archived versions. In the 'Merging From Versions' select list, choose the version(s) whose issues you wish to move. Versions selected on this list will be removed from the system. All issues associated with these versions will be updated to reflect the new version selected in the 'Merge To Version' select list. It is only possible to select one version to merge to.
3. Click the '**Merge**' button. If you are shown a confirmation page, click '**Merge**' again to complete the operation.

Editing a version's details

1. On the 'Versions' screen (see Screenshot 2 above), hover over the relevant version to display the pencil icon.
2. This will allow you to edit the version's Name, Description and Release Date.
3. Click the **'Update'** button to save your changes.

Deleting a version

1. On the 'Versions' screen (see Screenshot 2 above), hover over the relevant version to display the cog icon, then select '**Delete**' from the drop-down menu.
2. This will bring you to the **"Delete Version: <Version>"** confirmation page. From here, you can specify the actions to be taken for issues associated with the version to be deleted. You can either associate these issues with another version, or simply remove references to the version to be deleted.

Rescheduling a version

Rescheduling a version changes its place in the order of versions.

- On the 'Versions' screen (see Screenshot 2 above), click the icon for the relevant version, and drag it to its new position in the version order.

See also

- The GreenHopper documentation on Setting Up a Version Hierarchy.

Running a Bamboo Build when Releasing a Version

Releasing a new version of software usually involves a number of tasks, such as releasing the version in JIRA, building and testing, merging code, creating tags, creating branches, labelling builds, etc. If you have integrated JIRA with Atlassian's Bamboo, you can trigger these tasks to run automatically at the release of a version in JIRA.

When you release a JIRA version, you will have the option of selecting a Bamboo Plan and specifying which Stages in the Plan to run. Releasing the version will run the Plan in Bamboo. If the Plan is successful, the version will be released on JIRA. Otherwise, the version will not be released.

On this page:

- Procedure
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Procedure

Before you begin:

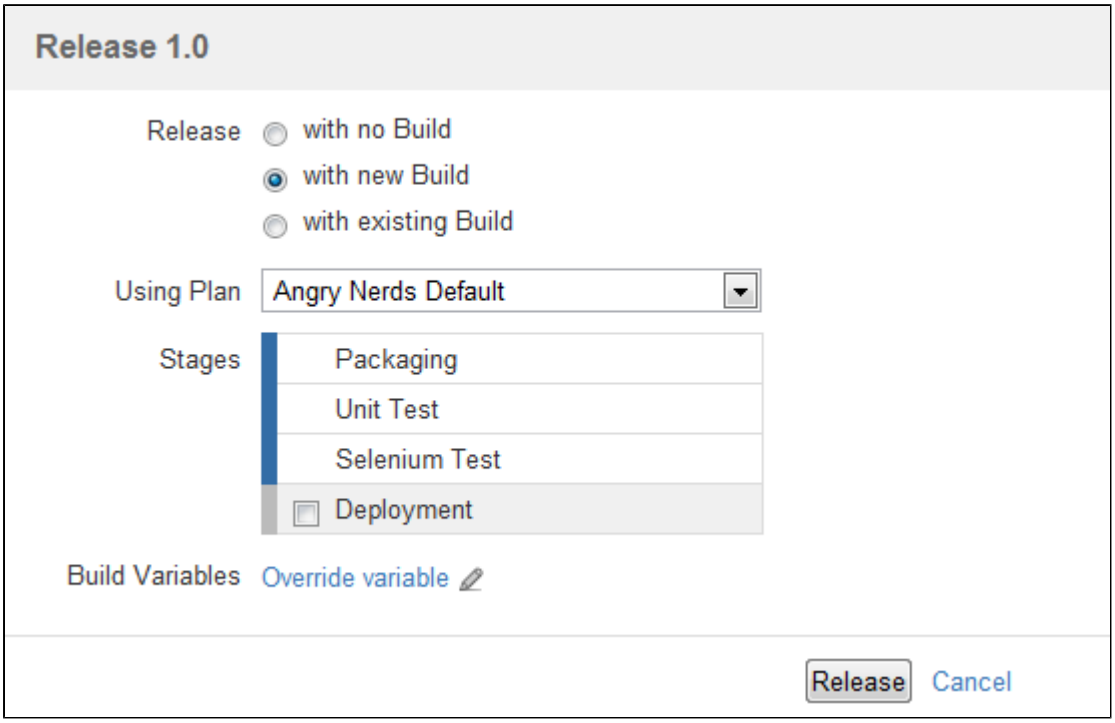
- Your JIRA administrator must have **integrated JIRA with Bamboo** (i.e. set up a two-way application link between JIRA and Bamboo). For instructions on how to integrate JIRA with Bamboo, see Integrating JIRA with Bamboo.
- Your JIRA administrator must have installed the latest JIRA Bamboo plugin to use the release management feature. For instructions on how to install a plugin, see Managing JIRA's Plugins.

To run a Bamboo build when releasing a version:

1. Log in to JIRA as a project administrator. (A project administrator is someone who has the project-specific permission **'Administer Project',** but not necessarily the global permission **'JIRA Administrators'**.)
2. Navigate to **'Projects' menu > the desired project > the desired version > 'Release' tab**.
   If you are a JIRA administrator, ensure that you are not in 'Administration' mode, otherwise you will not see the controls described above.
3. Release the version. The release build dialogue will be displayed (see screenshot below).
4. Enter the build details for the release:

'No Build'	Choose this option if you do not want to run a Bamboo build, i.e. you only want to release the version in JIRA.
'Release new Build'	Choose this option, if you want to run a Bamboo build that has not been started: * '**Using Plan**' — You will be able to select any Plan in the linked Bamboo instance that you have permission to view (unless your administrator has configured basic HTTP authentication). * '**Stages**' — You will be able to select the Stages that you want to run for this release. Note, you cannot skip Stages. * '**Build Variables**' — You can choose to override build variables (similar to Triggering a Plan Build Manually in Bamboo).
'Release existing Build'	Choose this option, if you want to run a Bamboo build that is in progress and has been paused at an optional Stage: * '**Using Plan**' — You will be able to select any Plan in the linked Bamboo instance that you have permission to view (unless your administrator has configured basic HTTP authentication). * '**Stages**' — You will be able to select the Stages that you want to run for this release. Note, you cannot re-run Stages that have been completed nor skip Stages. * '**Build Variables**' — You can choose to override build variables (similar to Triggering a Plan Build Manually in Bamboo).

5. Click the '**Release**' button. The Bamboo build will run. If it is successful, the JIRA version will be released. If not, you can choose to run it again or select a different Plan.



Screenshot above: Selecting the build for a release

Notes

Related Topics

Integrating JIRA with Bamboo
Managing JIRA's Plugins

Creating Release Notes

JIRA provides the functionality to create release notes for a specific version of a project. The release notes contain all issues within the specified project that are marked with a specific "Fix For" version. The release notes can also be generated in a number of formats (e.g. HTML, plain text, etc.) so as they can be included in various documents.

At present, two example format templates are provided - HTML and Text - using Velocity templates. Further format templates can be created and added to the system.

Generating Release Notes

1. Click **'Projects'** and select the project of interest.
2. Click '**Road Map**' tab on the left of the screen.
   Tip: *If you wish to see past release notes click on the '**Change Log**' tab instead.*

3. Click '**Release Notes**' link for the project version whose release notes you wish to generate. The 'Release Notes' page will be displayed.
4. Click the '**Configure Release Notes**' link to configure the release notes. The 'Configure Release Notes' page will be displayed:
    - Select the required project version for which the release notes will be generated in the '**Please select version**' dropdown.
    - Select the required format of the release notes — HTML and plain text format templates are provided in the '**Please select style**' dropdown.
5. Selecting the '**Create**' button will generate the release notes using the specified template in the specified format. The release notes will be displayed on screen and can be copied and pasted to another application.

Adding a New Format Template

1. Create a Velocity template similar in content to that of the examples provided — `releasenotes-text.vm` and `releasenotes-html.vm`. Consult the JIRA API documentation and the Apache Velocity User Guide.
2. The title within the template should be modified along with the code within the text area. The other sections of the template do not need to be modified.
3. Add the new format template to the list of existing ones within the `jira-config.properties`file. For each new template format, corresponding entries must be added to the existing values of the following properties:
    - `jira.releasenotes.templatenames`
    - `jira.releasenotes.templates`

    ⚠ Corresponding entries in both of these properties must be in the same order.

    ✅ If these properties do not exist in your `jira-config.properties` file, then for each of these properties, add the property's name, followed by an '=', followed by the content of the property's corresponding `<default-value/>` element copied from your JIRA installation's `jpm.xml` file. Next, begin adding the corresponding entries for the new format template.

    ℹ See Making changes to the `jira-config.properties` file for more information.
4. The new format template is available for selection as a release note format template.

Also see the tutorial on Creating a Custom Release Notes Template Containing Release Comments.

Configuring Project Keys

JIRA provides the ability to specify the format of project keys within the system. This is achieved by defining a regular expression 'rule' that governs the valid project key format

ℹ **Please Note:**

- By default, JIRA issue keys (aka Issue IDs) are of the format **<project key>-<issue number>**, eg. ABC-123.
- ⚠ If you use GreenHopper or have integrated JIRA with Bamboo, you should avoid changing JIRA's default project key format as both GreenHopper and Bamboo only support this key format.

Project Key Pattern

The `jira.projectkey.pattern` property allows JIRA administrators to specify a Perl5 regular expression value that defines the rule for a valid project key. During project creation, JIRA project administrators must specify a project key that conforms to this rule.

This property and its regular expression value can be defined through the 'Advanced Settings' page.

Default Project Key Pattern

By default, the JIRA project key configuration requires two or more uppercase alphabetical characters — based on the regular expression (`[A-Z][A-Z]+`).

Configuring Different Project Key Patterns

To configure a different rule for your project key syntax, change the regular expression (via the 'Advanced Settings' page). Below is a list of common examples and patterns:

Pattern Requested	Expression needed	Resulting Issue IDs	Comments
XXYY, where X indicates two fixed letters, Y represents two fixed digits	(`[A-Z]{2}[0-9]{2}`)	TQ09-01, TQ09-02, etc.	`[A-Z]` Any character from A to Z `{2}` Matches the preceding character 2 times exactly `[0-9]` Any character (i.e.digit) from 0 to 9
XZ+, where X indicates one fixed letter, Z+ presents one or more digit or alphabet	`([A-Z][A-Z0-9]+)`	ACAT51-1, AAA5-1330, A20091-15	`[A-Z]` Any characters from A to Z `[A-Z0-9]` Any character from A to Z or 0 to 9 + specifies `[A-Z0-9]` as one or more characters from A to Z or 0 to 9

Please Note:

- JIRA prepends the regular expression specified with '^' and closes it with '$' for an exact matching rule within the system.
- The project key must only be allowed to contain ASCII characters, as it is used in HTTP GET requests.
- The project key only supports uppercase characters. For simplicity, use uppercase characters in your expressions as JIRA will convert any lowercase characters to uppercase ones.

Do not use "." and "-"

- Avoid using the dash (`-`) character, as this character is inserted automatically after the regular expression and before the issue ID number.
- Avoid using the dash (`.`) character (i.e. dot/period/full-stop), as the CreateOrCommentHandler currently fails to add comments to existing issues if the project key contains a dot(.) — see JRA-23180.

Warning!

- If a number of issues have already been created in your JIRA installation, then *changing the project key is not recommended*.
- If you must change the project key pattern after issues have already been created, use a regular expression that allows a more 'permissive' project key pattern than the current one (i.e. use a regular expression which will still be valid for existing project keys defined in your JIRA installation).

Customising the Error Message for Invalid Keys

If JIRA detects that the project key entered does not match the `jira.projectkey.pattern`, it will throw the error message defined in `jira.projectkey.warning`.

You can change `jira.projectkey.warning` in the `jira-config.properties` file so that when a user keys in the wrong format, they will be informed of the correct pattern to use.

Testing Your Regular Expression

A variety of tools allow searching using a Regular Expression. Most text editors will allow a Regular Expression search. There are also a variety of websites available to for testing a Regular Expression available from an Internet search.

Project Key Details

You can customise the following properties in the `jira-config.properties` file:

- `jira.projectkey.description` — a configurable description (to match the project key pattern) displayed on project creation
- `jira.projectkey.warning` — a configurable validation warning (to match the project key pattern)

Please Note:

- It is not possible to configure the issue key pattern as JIRA expects this key to conform to specific rules.
- Further information on Perl5 is available here.

Configuring Security

When configuring security for your JIRA instance, there are two areas to address:

- security within JIRA itself
- security in the external environment

Configuring security within JIRA

JIRA has a flexible security system which allows you to configure who can access JIRA, and what they can do/see within JIRA.

There are five types of security within JIRA:

1. Global permissions — these apply to JIRA as a whole (e.g. who can log in).
2. Project permissions — organised into permission schemes, these apply to projects as a whole (e.g. who can see the project's issues ('Browse' permission), create, edit and assign them).
3. Issue security levels — organised into security schemes, these allow the visibility of individual issues to be adjusted, within the bounds of the project's permissions.
4. Comment visibility — allows the visibility of individual comments (within an issue) to be restricted.
5. Work-log visibility — allows the visibility of individual work-log entries (within an issue) to be restricted. Does not restrict visibility of progress bar on issue time tracking.

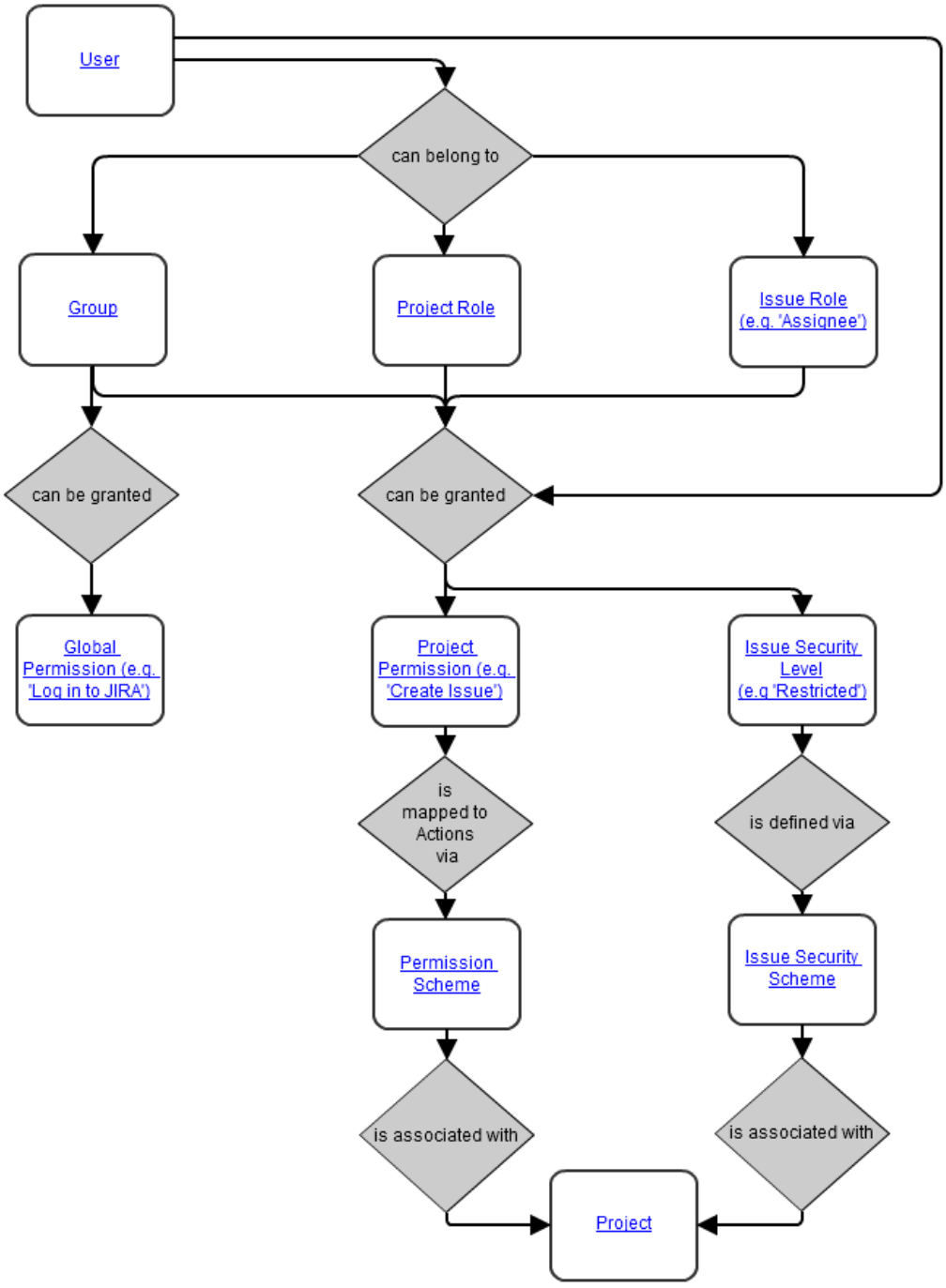
On this page:

- Configuring security within JIRA
  - Diagram: People and permissions
- Configuring security in the external environment
- Other security resources

In this section:

- Configuring Issue Level Security
- Managing Project Permissions
- Managing Global Permissions
- Configuring Secure Administrator Sessions
- Preventing Security Attacks
- JIRA Cookies

Diagram: People and permissions



Configuring security in the external environment

If your JIRA instance contains sensitive information, you may want to configure security in the environment in which your JIRA instance is running. Some of the main areas to consider are:

- Database:
  - If you are using an external database as recommended for production systems (i.e. you are not using JIRA's internal/bundled HSQL database), you should restrict access to the database that your JIRA instance uses.
  - If you are using JIRA's internal/bundled HSQL database, you should restrict access to the directory in which you installed JIRA. (Note that the user which your JIRA instance is running as will require full access to this directory.)
- File system — you should restrict access to the following directories (but note that the user which your JIRA instance is running as will require full access to these directories):
  - Index directory
  - Attachments directory
- SSL — if you are running your JIRA instance over the Internet, you may want to consider using SSL.

Other security resources

Security Addendum 2010-04-16 - Preventing security attacks

No content found for label(s) security-resources.

Configuring Issue Level Security

What is an Issue Security Level?

Issue security levels allow you to control who can see individual issues within a project (subject to the project's permissions).

An issue security level is a *named collection of users.* Issue security levels are created within issue security schemes, which are then associated with projects. Once an issue security scheme has been associated with a project, its security levels can be applied to issues in that project (note, sub-tasks will inherit the security level of their parent issue). Those issues will then only be accessible to members of that security level.

A security level's members may consist of:

- Individual users
- Groups
- Project roles
- Issue roles such as 'Reporter', 'Project Lead', and 'Current Assignee'
- 'Anyone' (eg. to allow anonymous access)
- A (multi-)user picker custom field.
- A (multi-)group picker custom field. This can either be an actual group picker custom field, or a (multi-)select-list whose values are group names.

Only users with the project-specific 'Set Issue Security' permission can apply a security level to an issue, regardless of whether they are members of the security level.

On this page:

- What is an Issue Security Level?
- Why use Issue Security?
- Creating an Issue Security Scheme
  - Adding a Security Level to an Issue Security Scheme
  - Setting the Default Security Level for an Issue Security Scheme
  - Adding Users/Groups/Project Roles to a Security Level
- Assigning an Issue Security Scheme to a Project
- Deleting an Issue Security Scheme
- Copying an Issue Security Scheme
- Related Topics

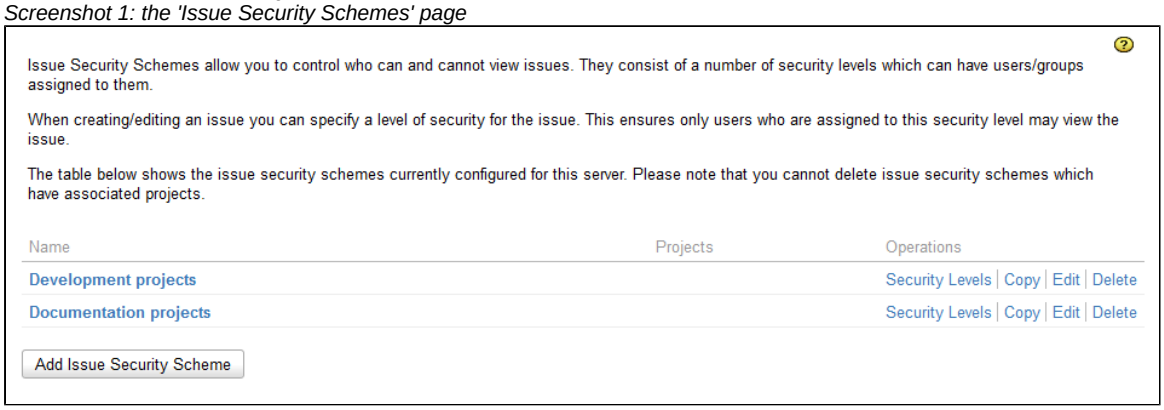
Why use Issue Security?

As an example, a company may have a public instance of JIRA running. Within this instance they may have several projects that external people (customers) can browse. However, it may not be appropriate to show all issues to the customers. To achieve this you could:

- Create an issue security scheme.
- Create an issue security level named 'Private' for this scheme.
- Add appropriate people to the 'Private' security level.
- Associate the issue security scheme with the relevant projects.
- Set the security level of specific issues to 'Private'.

Creating an Issue Security Scheme

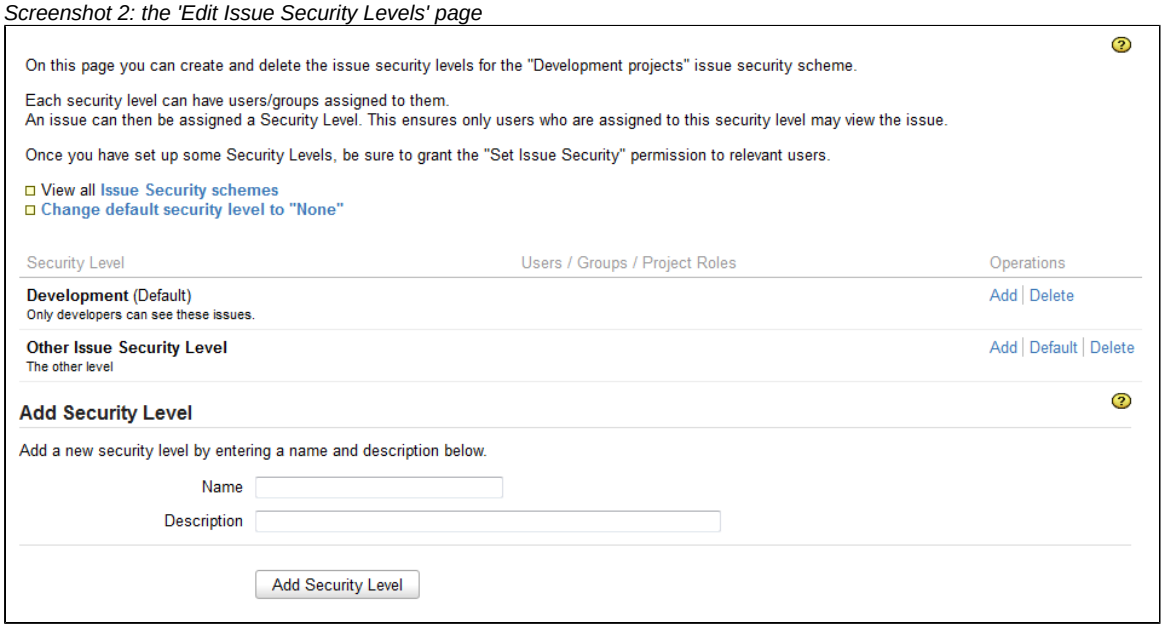
1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Security Schemes'** to open the 'Issue Security Schemes' page, which lists all the Issue Security Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue security schemes'**
3. Click the '**Add Issue Security Scheme**' button.
   Screenshot 1: the 'Issue Security Schemes' page



4. In the 'Add Issue Security Scheme' form, enter a name for the issue security scheme, and a short description of the scheme. Then click the '**Add**' button.
5. You will return to the 'Issue Security Schemes' page, which now contains the newly added scheme.

Adding a Security Level to an Issue Security Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Security Schemes'** to open the 'Issue Security Schemes' page, which lists all the Issue Security Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue security schemes'**
3. Click the name of any scheme, or the link '**Security Levels**' (in the '**Operations**' column), to bring up the 'Edit Issue Security Levels' page.
4. In the '**Add Security Level**' box, enter a name and description for your new security level. Then click '**Add Security Level**'.
   Screenshot 2: the 'Edit Issue Security Levels' page



Setting the Default Security Level for an Issue Security Scheme

You can choose to specify a Default Security Level for your issue security scheme.

The Default Security Level is used when issues are created. If the reporter of an issue does not have the permission 'Set Issue Security', then the issue's security level will be set to the Default Security Level. If the project's issue security scheme does not have a Default Security Level, then the issue's security level will be set to 'None'. (A security level of 'None' means that anybody can see the issue.)

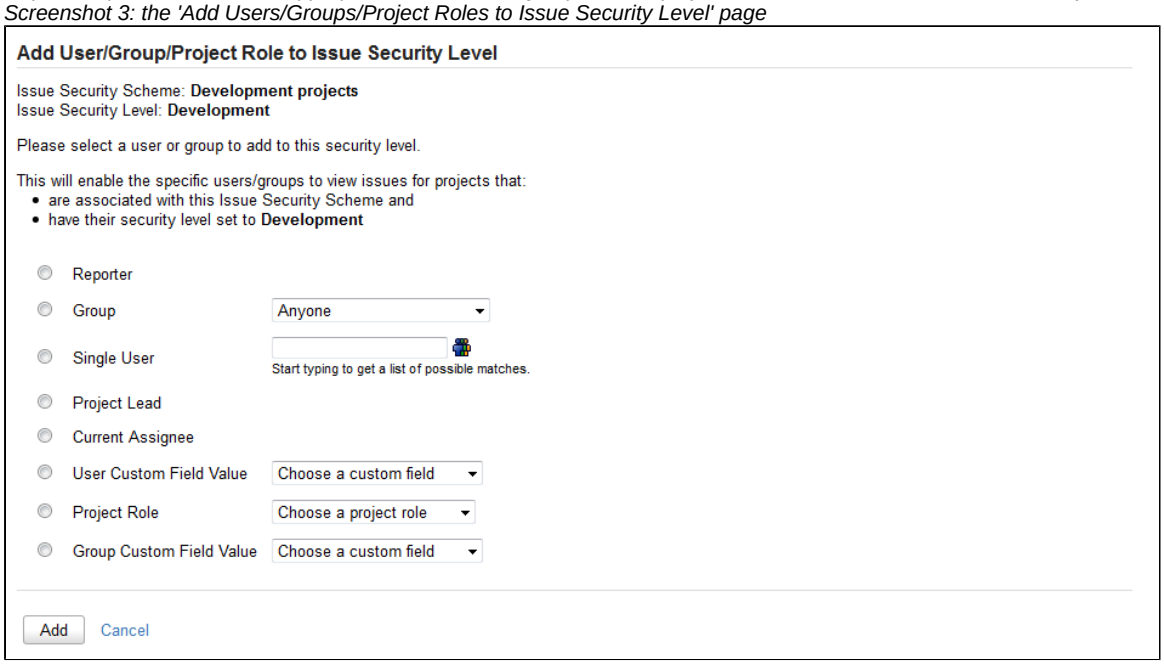
1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Security Schemes'** to open the 'Issue Security Schemes' page (above), which lists all the Issue Security Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue security schemes'**
3. Click the name of any scheme or the link '**Security Levels**' to bring up the 'Edit Issue Security Levels' page (above).

- To set the Default Security Level for an issue security scheme, locate the appropriate Security Level and click its '**Default**' link (in the 'Operations' column).
- To remove the Default Security Level for an issue security scheme, click the '**Change default security level to "None"**' link (near the top of the page).

Adding Users/Groups/Project Roles to a Security Level

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Security Schemes'** to open the 'Issue Security Schemes' page (above), which lists all the Issue Security Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue security schemes'**
3. Click the name of any scheme or the link '**Security Levels**' to bring up the 'Edit Issue Security Levels' page (above).
4. Locate the appropriate Security Level and click its '**Add**' link (in the 'Operations' column).
5. This will display the 'Add Users/Groups/Project Roles to Issue Security Level' page. Select the appropriate user, group or project role, then click the '**Add**' button.
6. Repeat steps 5 and 6 until all appropriate users and/or groups and/or project roles have been added to the security level.

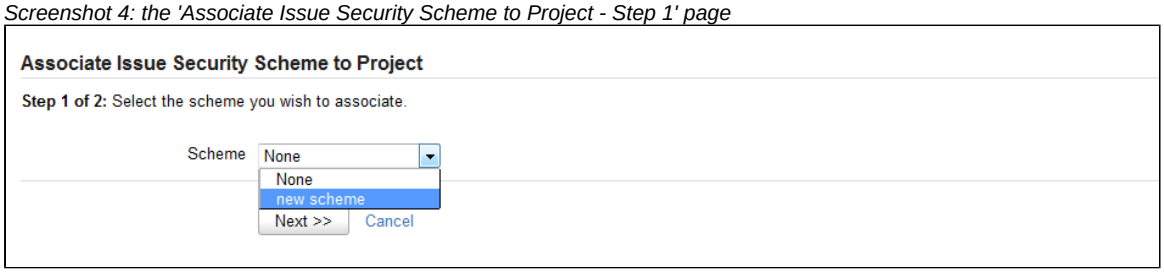
Screenshot 3: the 'Add Users/Groups/Project Roles to Issue Security Level' page



Assigning an Issue Security Scheme to a Project

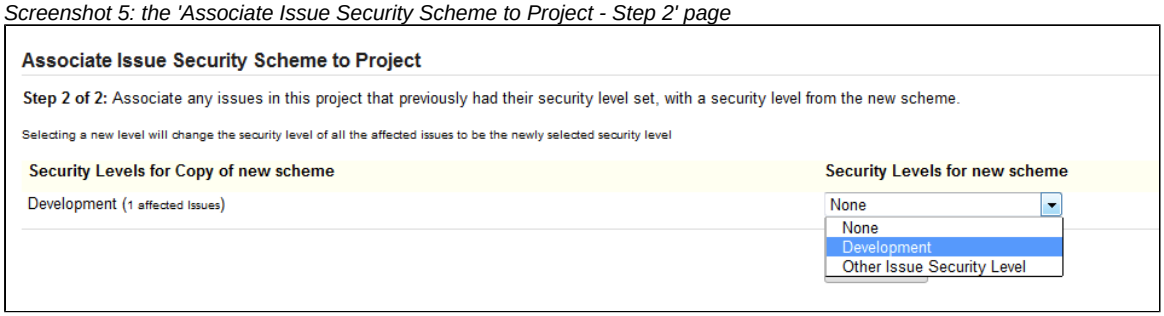
1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click the '**Administration**' link at the top of the screen.
3. Click **'Projects'** and select the project of interest. This will display the Project Summary screen (see Defining a Project).
4. In the **'Permissions'** section, click the link that corresponds to the '**Issues**' link (this will either be the name of the project's current Issue Security Scheme, or the word '**None**').
5. This will bring up a list of Issue Security Schemes. Select the Issue Security Scheme that you want to associate with this project.

Screenshot 4: the 'Associate Issue Security Scheme to Project - Step 1' page



6. If there are no previously secured issues (or if the project didn't previously have an issue security scheme), skip the next step.
7. If there are any previously secured issues, select a new security level to replace each old level. All issues with the security level from the old scheme will now have the security level from the new scheme. You can choose 'None' if you want the security to be removed

from all previously secured issues.
Screenshot 5: the 'Associate Issue Security Scheme to Project - Step 2' page



8. Click the '**Associate**' button to associate the project with the issue security scheme.

Deleting an Issue Security Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Security Schemes'** to open the 'Issue Security Schemes' page (above), which lists all the Issue Security Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue security schemes'**
3. Click the '**Delete**' link (in the 'Operations' column) for the scheme that you want to delete.
   You cannot delete a Issue Security Scheme if it is associated with a project; you must first unassign the scheme. To unassign a scheme, please refer to Assigning an Issue Security Scheme.
4. On the confirmation page, click '**Delete**' to confirm the deletion. Otherwise, click '**Cancel**'.

Copying an Issue Security Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Security Schemes'** to open the 'Issue Security Schemes' page (above), which lists all the Issue Security Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue security schemes'**
3. Click the '**Copy**' link (in the 'Operations' column) for the scheme that you want to copy.
4. A new scheme will be created with the same security levels and the same users/groups/project roles assigned to them. Your new scheme will be called '**Copy of ...**'. You can edit your new scheme to give it a different name if you wish.

Related Topics

- Setting Security on an Issue

Managing Project Permissions

Project permissions are created within Permission Schemes, which are then assigned to specific projects.

Project permissions can be granted to:

- Individual users
- Groups
- Project roles
- Issue roles such as 'Reporter', 'Project Lead' and 'Current Assignee'
- 'Anyone' (e.g. to allow anonymous access)
- A (multi-)user picker custom field.
- A (multi-)group picker custom field. This can either be an actual group picker custom field, or a (multi-)select-list whose values are group names.

The following table lists the different types of project permissions and the functions they secure. Note that project permissions can also be used in workflow conditions.

On this page:

- Permission Schemes
  - What is a Permission Scheme?
  - Why Permission Schemes?
- Creating a Permission Scheme
  - Adding Users, Groups or Roles to a Permission Scheme
  - Deleting Users, Groups or Roles from a Permission Scheme
- Associating a Permission Scheme with a Project
- Deleting a Permission Scheme
- Copying a Permission Scheme

Project Permissions	Explanation
Administer Projects	Permission to administer a project in JIRA. This includes the ability to edit project role membership, project components, project versions and some project details ('Project Name', 'URL', 'Project Lead', 'Project Description').
Browse Projects	Permission to browse projects, use the Issue Navigator and view individual issues (except issues that have been restricted via Issue Security). **Many other permissions are dependent on this permission**, e.g. the 'Work On Issues' permission is only effective for users who also have the 'Browse Projects' permission.
View Version Control	Permission to view the version control information (e.g. CVS, Subversion, FishEye, etc) for an issue. Note that for CVS, to view the Version Control information the project needs to be associated with at least one Repository.
View (Read-Only) Workflow	Permission to view the project's 'read-only' workflow when viewing an issue. This permission provides the 'View Workflow' link against the 'Status' field of the 'View Issue' page.
Issue Permissions	**Explanation**
Assign Issues	Permission to assign issues to users. (See also Assignable User permission below)
Assignable User	Permission to be assigned issues. (Note that this does not include the ability to assign issues; see Assign Issue permission above).
Close Issues	Permission to close issues. (This permission is useful where, for example, developers resolve issues and testers close them). Also see the Resolve Issues permission.
Create Issues	Permission to create issues in the project. (Note that the Create Attachments permission is required in order to create attachments.) Includes the ability to create sub-tasks (if sub-tasks are enabled).
Delete Issues	Permission to delete issues. Think carefully about which groups or project roles you assign this permission to; usually it will only be given to administrators. Note that deleting an issue will delete all of its comments and attachments, even if the user does not have the Delete Comments or Delete Attachments permissions. However, the Delete Issues permission does not include the ability to delete individual comments or attachments.
Edit Issues	Permission to edit issues (excluding the 'Due Date' field — see the Schedule Issues permission). Includes the ability to convert issues to sub-tasks and vice versa (if sub-tasks are enabled). Note that the Delete Issue permission is required in order to delete issues. The Edit Issue permission is usually given to any groups or project roles who have the Create Issue permission (perhaps the only exception to this is if you give **everyone** the ability to create issues — it may not be appropriate to give everyone the ability to edit too). Note that all edits are recorded in the Issue Change History for audit purposes.
Link Issues	Permission to link issues together. (Only relevant if Issue Linking is enabled).
Modify Reporter	Permission to modify the 'Reporter' of an issue. This allows a user to create issues 'on behalf of' someone else. This permission should generally only be granted to administrators.
Move Issues	Permission to move issues from one project to another, or from one workflow to another workflow within the same project. Note that a user can only move issues to a project for which they have Create Issue permission.
Resolve Issues	Permission to resolve and reopen issues. This also includes the ability to set the 'Fix For version' field for issues. Also see the Close Issues permission.
Schedule Issues	Permission to schedule an issue — that is, set and edit the 'Due Date' of an issue.
Set Issue Security	Permission to set the security level on an issue to control who can access the issue. Only relevant if issue security has been enabled.

Voters & Watchers Permissions	**Explanation**
Manage Watcher List	Permission to manage (i.e. view/add/remove users to/from) the watcher list of an issue.
View Voters and Watchers	Permission to view the voter list and watcher list of an issue. Also see the Manage Watcher List permission.
Comments Permissions	**Explanation**
Add Comments	Permission to add comments to issues. Note that this does not include the ability to edit or delete comments.
Delete All Comments	Permission to delete any comments, regardless of who added them.
Delete Own Comments	Permission to delete comments that were added by the user.
Edit All Comments	Permission to edit any comments, regardless of who added them.
Edit Own Comments	Permission to edit comments that were added by the user.
Attachments Permissions	**Explanation**
Create Attachments	Permission to attach files to an issue. (Only relevant if attachments are enabled). Note that this does not include the ability to delete attachments.
Delete All Attachments	Permission to delete any attachments, regardless of who added them.
Delete Own Attachments	Permission to delete attachments that were added by the user.
Time Tracking Permissions	**Explanation**
Work On Issues	Permission to log work against an issue, i.e. create a worklog entry. (Only relevant if Time Tracking is enabled).
Delete All Worklogs	Permission to delete any worklog entries, regardless of who added them. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.
Delete Own Worklogs	Permission to delete worklog entries that were added by the user. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.
Edit All Worklogs	Permission to edit any worklog entries, regardless of who added them. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.
Edit Own Worklogs	Permission to edit worklog entries that were added by the user. (Only relevant if Time Tracking is enabled). Also see the Work On Issues permission.

Permission Schemes

What is a Permission Scheme?

A permission scheme is a set of user/group/role assignments for the project permissions listed above. Every project has a permission scheme. One permission scheme can be associated with multiple projects.

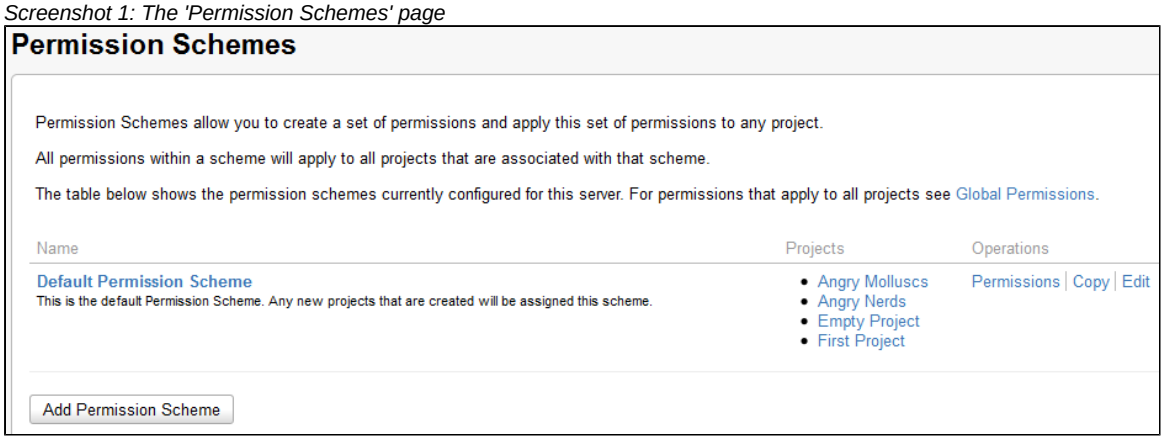
Why Permission Schemes?

In many organisations, multiple projects have the same needs regarding access rights. (For example, only the specified project team may be authorised to assign and work on issues).

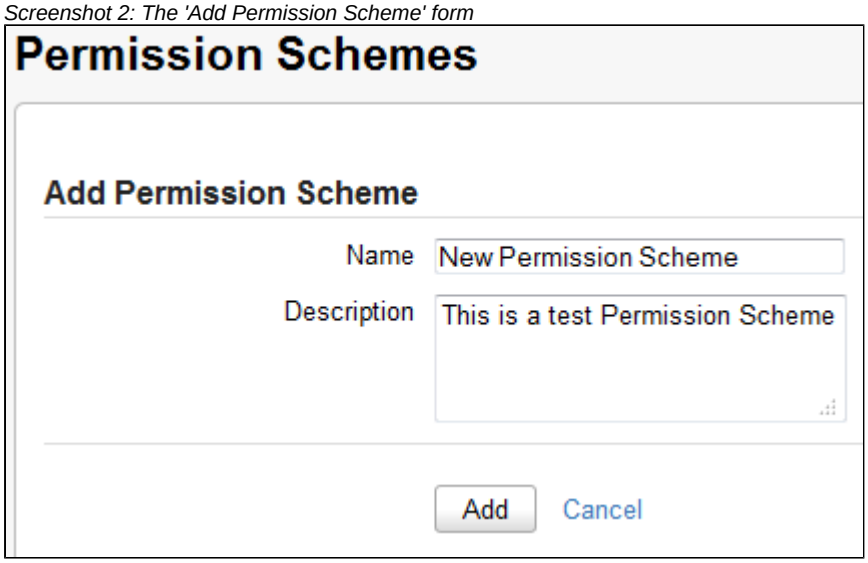
Permission schemes prevent having to set up permissions individually for every project. Once a permission scheme is set up it can be applied to all projects that have the same type of access requirements.

Creating a Permission Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Permission Schemes'** to open the 'Permission Schemes' page, which lists all the Permission Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'permission schemes'**
3. Click the '**Add Permission Scheme**' link.
   Screenshot 1: The 'Permission Schemes' page



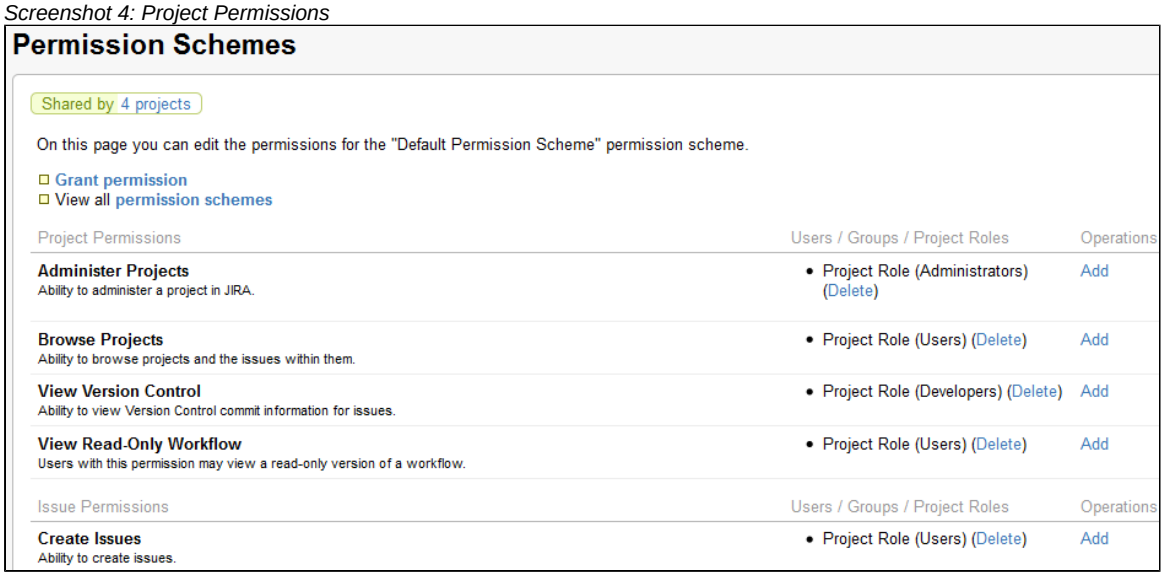
4. In the 'Add Permission Scheme' form, enter a name for the scheme, and a short description of the scheme. Click the '**Add**' button.
   Screenshot 2: The 'Add Permission Scheme' form



5. You will return to the 'Permission Schemes' page which now contains the newly added scheme.
   Screenshot 3: the 'View Permission Schemes' page, showing the newly added scheme

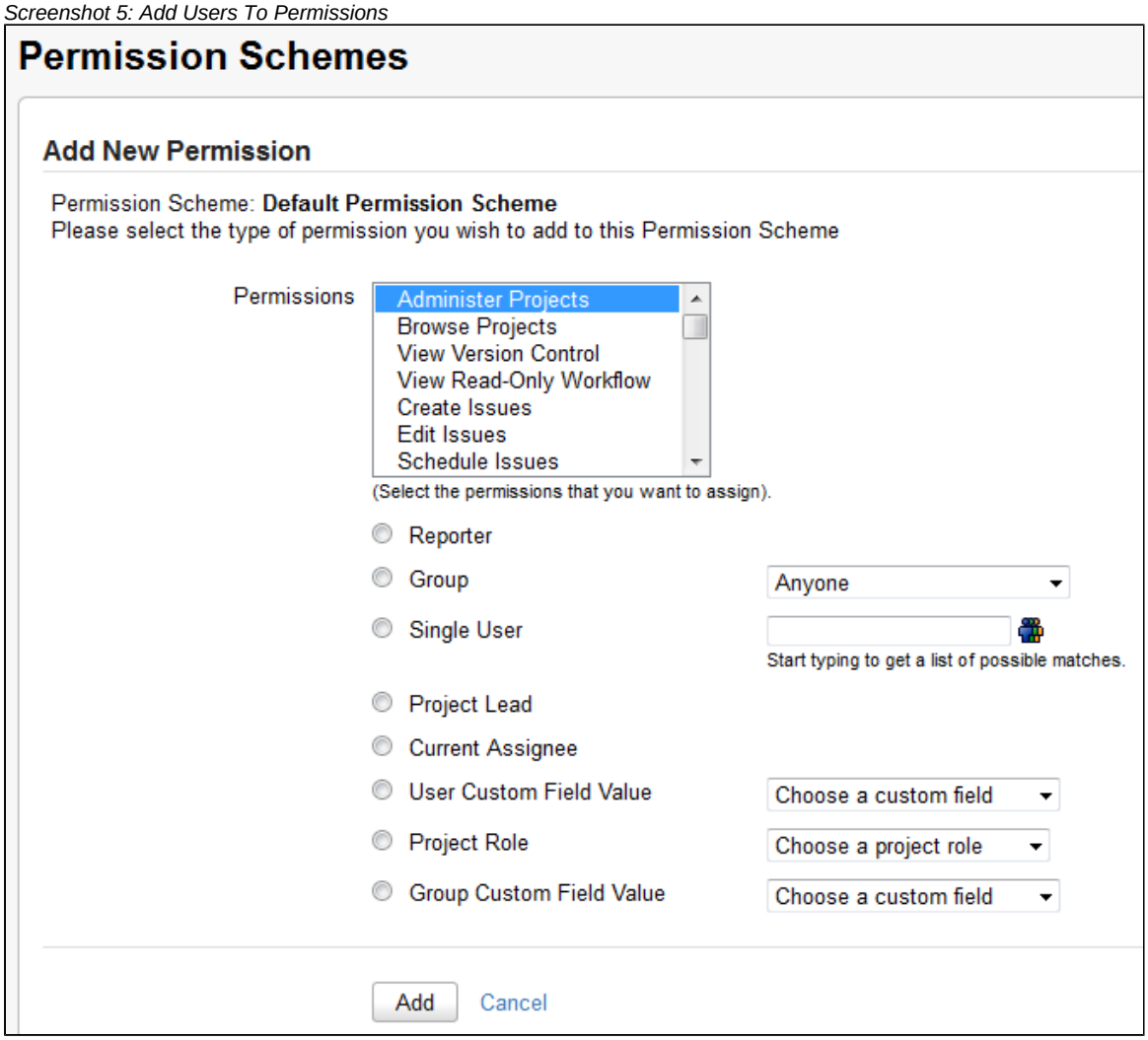
Adding Users, Groups or Roles to a Permission Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Permission Schemes'** to open the 'Permission Schemes' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'permission schemes'**
3. Locate the permission scheme of interest and click its name (or click the '**Permissions**' link in the 'Operations' column) to show a list of permissions.
4. Click the '**Add**' link in the 'Operations' column, which displays the 'Add Permission' page.
   Screenshot 4: Project Permissions



5. After selecting one or more permissions to add and who to add the selected permissions to, click the '**Add**' button. The users/groups/roles will now be added to the selected permissions. Note that project roles are useful for defining specific team members for each project. Referencing project roles (rather than users or groups) in your permissions can help you minimise the

number of permission schemes in your system.
Screenshot 5: Add Users To Permissions



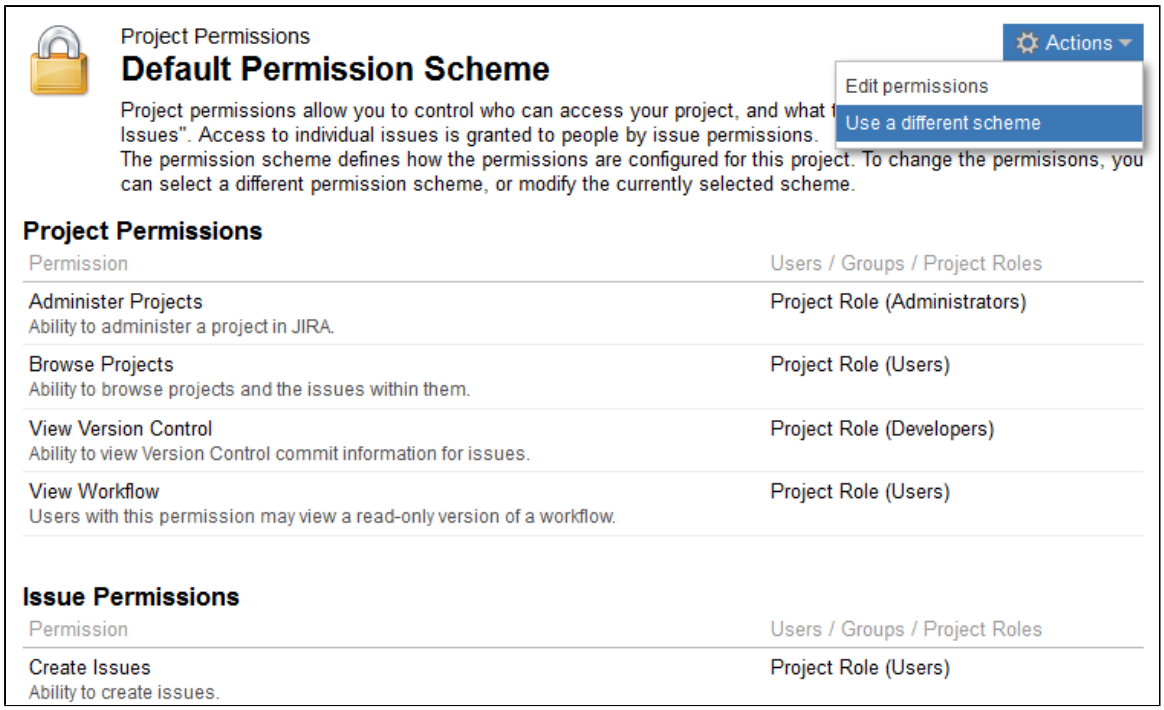
6. Repeat the last 2 steps until all required users/groups/roles have been added to the permissions.

Deleting Users, Groups or Roles from a Permission Scheme

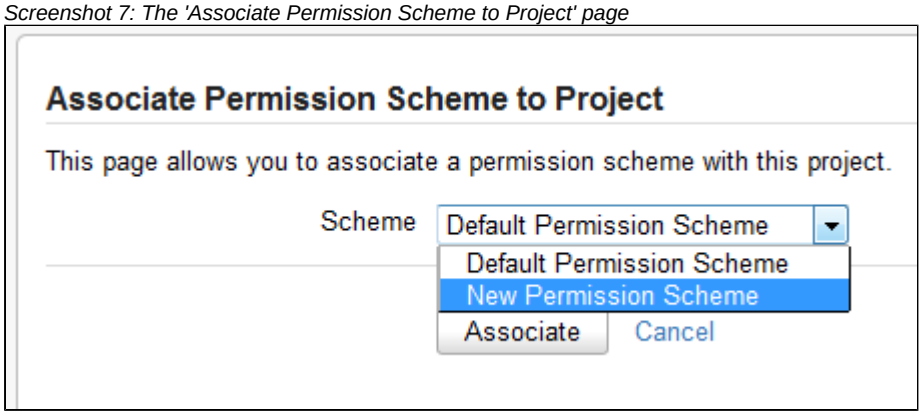
1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Permission Schemes'** to open the 'Permission Schemes' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'permission schemes'**
3. This will display the 'Managing Project Permissions' page (see above).
4. Location the permission scheme of interest and click its name (or click the '**Permissions**' link in the 'Operations' column) to show the list of 'Project Permissions' (above).
5. Click the '**(Delete)**' link in the "Users / Groups / Roles" column next to the name of the user, group or project role you wish to delete.

Associating a Permission Scheme with a Project

1. Log in to JIRA as a project administrator. (A project administrator is someone who has the project-specific permission **'Administer Project'**, but not necessarily the global permission '**JIRA Administrators**'.)
2. Click the '**Administration**' link at the top of the screen.
3. Click **'Projects'** and select the project of interest. You will now see the 'Project Summary' screen (see Defining a Project).
4. On the lower right, in the 'Permissions' section, click the name of the current scheme (e.g. 'Default Permission Scheme') to display the details of the project's current permission scheme.
5. Click the '**Actions**' dropdown menu and choose '**Use a different scheme**'.
   Screenshot 6: The Project Permissions page



6. On the 'Associate Permission Scheme to Project' page, which lists all available permission schemes, select the permission scheme you want to associate with the project.
Screenshot 7: The 'Associate Permission Scheme to Project' page



7. Click the '**Associate**' button to associate the project with the permission scheme.

Deleting a Permission Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Permission Schemes'** to open the 'Permission Schemes' page, which lists all the Permission Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'permission schemes'**
3. Click the '**Delete**' link (in the 'Operations' column) for the scheme that you want to delete.
4. A confirmation screen will appear. To delete click '**Delete**' otherwise click '**Cancel**'.
5. The scheme will be deleted and all associated projects will be automatically associated with the Default Permission Scheme. (Note that you cannot delete the Default Permission Scheme.)

See also Minimising the number of Permission Schemes and Notification Schemes.

Copying a Permission Scheme

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'Issues' > 'Permission Schemes'** to open the 'Permission Schemes' page, which lists all the Permission Schemes currently available in your JIRA installation.
   Keyboard shortcut: **'g' + 'g' +** start typing **'permission schemes'**
3. Click the '**Copy**' link (in the 'Operations' column) for the scheme that you want to copy.
4. A new scheme will be created with the same permissions and the same users/groups/roles assigned to them.

Managing Global Permissions

Global permissions are system wide and are granted to groups of users.

See also project permissions, which apply to individual projects.

*Some functionality described on this page is restricted in **JIRA OnDemand**.*

This table lists the different global permissions and the functions they secure:

Global Permission	Explanation
JIRA System Administrators	Permission to perform all JIRA administration functions. This does not include the **JIRA Users** permission. A user with **JIRA System Administrators** will be able to log in to JIRA without the **JIRA Users** permission, but may not be able to perform all regular user functions (e.g. edit their profile) unless they also belong to a group that has the **JIRA Users** permission.
JIRA Administrators	Permission to perform most JIRA administration functions (see list of exclusions below). This does not include the **JIRA Users** permission. A user with **JIRA Administrators** will be able to log in to JIRA without the **JIRA Users** permission, but may not be able to perform all regular user functions (e.g. edit their profile) unless they also belong to a group that has the **JIRA Users** permission.
JIRA Users	Permission to log in to JIRA. The number of users that count towards your JIRA license is the sum of all users (including users in groups) that have this permission. If you want to reduce this count, see How do I reduce my user count in JIRA. Granting the **JIRA Users** permission to a group also means that all newly created users will be automatically added to that group, unless the group has the **JIRA System Administrators** permission (as it would be unwise to automatically grant the **JIRA System Administrators** permission to all new users).
Browse Users	Permission to view a list of all JIRA user names and group names. Used for selecting users/groups in popup screens (such as the 'User Picker'). Enables auto-completion of user names in the 'User Picker' popup screen.
Create Shared Objects	Permission to share a filter or dashboard globally or with groups of users.
Manage Group Filter Subscriptions	Permission to manage (create and delete) group filter subscriptions.
Bulk Change	Permission to execute the bulk operations within JIRA: - Bulk Edit * - Bulk Move * - Bulk Workflow Transition - Bulk Delete * (* subject to project-specific permissions.) The decision to grant the **Bulk Change** permission should be considered carefully. This permission grants users the ability to modify a collection of issues at once. For example, in JIRA installations configured to run in Public mode (i.e. anybody can sign up and create issues), a user with the **Bulk Change** global permission and the **Add Comments** project permission could comment on *all* accessible issues. Undoing such modifications may not be possible through the JIRA application interface and may require changes made directly against the database (which is not recommended).

About 'JIRA System Administrators' and 'JIRA Administrators'

People who have the **JIRA System Administrators** permission can perform all of the administration functions in JIRA, while people who have only the **JIRA Administrators** permission cannot perform functions which could affect the application environment or network. This is useful for organisations which need to delegate some administrative privileges (e.g. creating users, creating projects) to particular people, without granting them complete rights to administer the JIRA system.

By default, the `jira-administrators` group has both the **JIRA Administrators** permission *and* the **JIRA System Administrators** permission. If you need some people to have only the **JIRA Administrators** permission (and not the **JIRA System Administrators** permission), you will need to use two separate groups, e.g.:

1. Create a new group (e.g. called `jira-system-administrators`).
2. Add to the `jira-system-administrators` group everyone who needs to have the **JIRA System Administrators** permission.
3. Grant the **JIRA System Administrators** permission to the `jira-system-administrators` group.
4. Remove the **JIRA System Administrators** permission from the `jira-administrators` group.
5. *(Optional, but recommended for ease of maintenance)* Remove from the `jira-administrators` group everyone who is a member of the `jira-system-administrators` group.

People who have the **JIRA Administrators** permission (and not the **JIRA System Administrators** permission) ***cannot*** do the following:

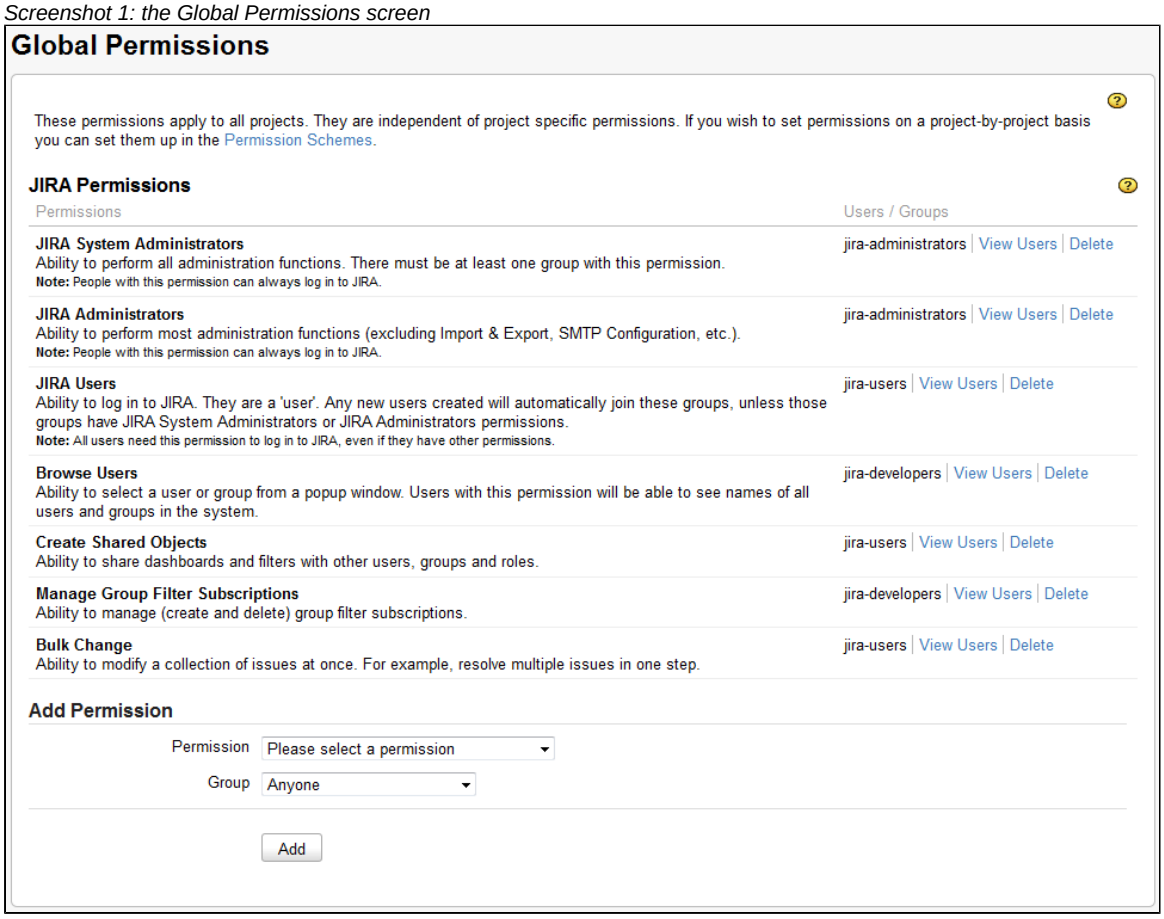
- Configure JIRA's SMTP mail server for notifications (but *they can* configure POP/IMAP mail servers for the receipt of email messages that create issue comments and new issues, and fully administer email notification schemes).
- Configure a CVS source code repository (but *they can* associate a project with a configured repository).
- Configure listeners.
- Configure services (except for POP/IMAP services).
- Change the index path (but *they can* reindex and optimise the index).
- Run the integrity checker.
- Access logging and profiling information.
- Access the scheduler.
- Export/backup JIRA data to XML.
- Import/restore JIRA data from XML.
- Import XML workflows into JIRA.
- Configure attachments (but *they can* set the size limits of attachments and enable thumbnails).
- Run Jelly scripts.
- Add gadgets to the Gadget Directory.
- Configure user directories (e.g. LDAP).
- Configure Application Links (but *they can* configure Entity Links).
- View user sessions.
- Access license details.
- Grant/revoke the **JIRA System Administrators** global permission.
- Edit (or Bulk Edit) groups that have the **JIRA System Administrators** global permission.
- Edit, change the password of or delete a user who has the **JIRA System Administrators** global permission.

It is recommended that people who have the **JIRA Administrators** permission (and not the **JIRA System Administrators** permission) are not given direct access to the JIRA filesystem or database.

Granting global permissions

1. Log in as a user with the **JIRA Administrators** global permission (or the **JIRA System Administrators** global permission, if you need to grant the **JIRA System Administrators** global permission).
2. Select **'Administration' > 'Users' > 'Global Permissions'** to open the **Global Permissions** page, which shows a list of JIRA's global permissions and which groups currently have these permissions.
   Keyboard shortcut: **'g' + 'g' +** start typing **'global permissions'**

Screenshot 1: the Global Permissions screen



The **Add Permission** box is shown below the list.

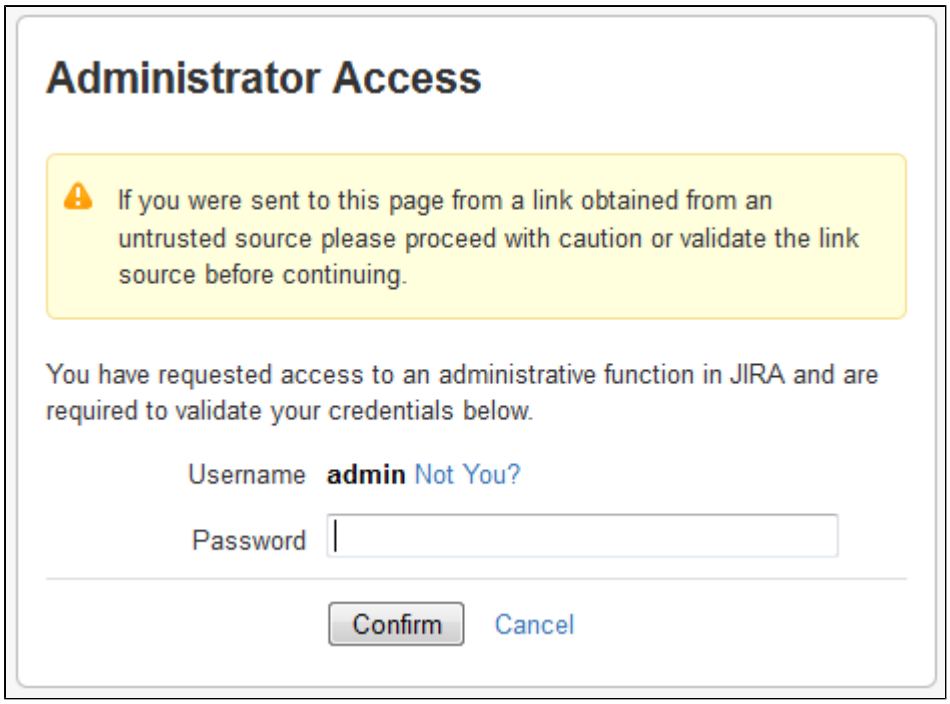
3. In the **Permission** drop-down list, select the global permission you wish to grant.
4. In the **Group**drop-down list, either:
    - select the group to which you wish to grant the permission; or
    - if you wish to grant the permission to non logged-in users, select **Anyone** (not recommended for production systems). Note that the **JIRA Users** permission (i.e. permission to log in) cannot be granted to **Anyone**(i.e. to non logged-in users) since this would be contradictory.

> If you have a user limited license (e.g. personal license) and have reached your user limit, you will not be able to grant login permissions (i.e. jira-users permission) to any further groups without first reducing the number of users with login permissions.

Configuring Secure Administrator Sessions

JIRA protects access to its administrative functions by requiring a secure administration session in order to use the JIRA administration screens. (This is also known as websudo.) When a JIRA administrator (who is logged into JIRA) attempts to access an administration function, they are prompted to log in again. This logs the administrator into a temporary secure session that grants access to the JIRA administration screens.

Screenshot: Logging in to a temporary secure session



On this page:

- Manually ending a Secure Administrator Session
- Disabling Secure Administrator Sessions
- Changing the Timeout
- Developer Notes

*The information on this page does not apply to **JIRA OnDemand**.*

The temporary secure session has a rolling timeout (defaulted to 10 minutes). If there is no activity by the administrator in the JIRA administration screens for a period of time that exceeds the timeout, then the administrator will be logged out of the secure administrator session (note that they will remain logged into JIRA). If the administrator does click an administration function, the timeout will reset.

Note that Project Administration functions (as defined by the 'Project Administrator' permission) do not require a secure administration session.

Manually ending a Secure Administrator Session

An administrator can choose to manually end their secure session by clicking the '**drop access**' link in the banner displayed at the top of their screen.

Disabling Secure Administrator Sessions

Secure administrator sessions (i.e. password confirmation before accessing administration functions) are enabled by default. If this causes issues for your JIRA site (e.g. if you are using a custom authentication mechanism), you can disable this feature by specifying the following line in your jira-config.properties file:

```
jira.websudo.is.disabled = true
```

You will need to restart your JIRA server for this setting to take effect.

Changing the Timeout

To change the number of minutes of inactivity after which a secure administator session will time out, specify the `jira.websudo.timeout` property (in your jira-config.properties file) whose value is the number of minutes of inactivity required before a secure administration session times out.

For example, the following line in your jira-config.properties file will end a secure administration session in 10 minutes:

```
jira.websudo.timeout = 10
```

You will need to restart your JIRA server for this setting to take effect.

Developer Notes

If you have written a plugin that has webwork actions in the JIRA Administration section, those actions should have the `@WebSudoRequired` annotation added to the class (not the method or the package, unlike Confluence).

Please also see Developing against JIRA with Secure Administrator Sessions and Adding WebSudo Support to your Plugin.

Preventing Security Attacks

This page provides guidelines which, to the best of our knowledge, will help prevent security attacks on your JIRA installation.

1. Use Strong Passwords

1.1 Administrators should use Strong Passwords

All your JIRA administrators, JIRA system administrators and administrators of all Atlassian products should have strong passwords. Ask your administrators to update their passwords to strong passwords.

Do not use passwords that are dictionary words. Use mixed-case letters, numbers and symbols for your administrator passwords and make sure they are sufficiently long (e.g. 14 characters). We encourage you to refer to the Strong Password Generator for guidelines on selecting passwords.

Using strong passwords greatly increases the time required by an attacker to retrieve your passwords by brute force, making such an attack impractical.

1.2 Administrators should have Different Passwords for Different Systems

As well as choosing a strong password, administrators should have *different* strong passwords for different systems.

This will reduce the impact the attacker can have if they do manage to obtain administrator credentials on one of your systems.

2. Apply JIRA Security Patches

Apply the patches found in JIRA Security Advisory 2010-04-16 for your version of JIRA.

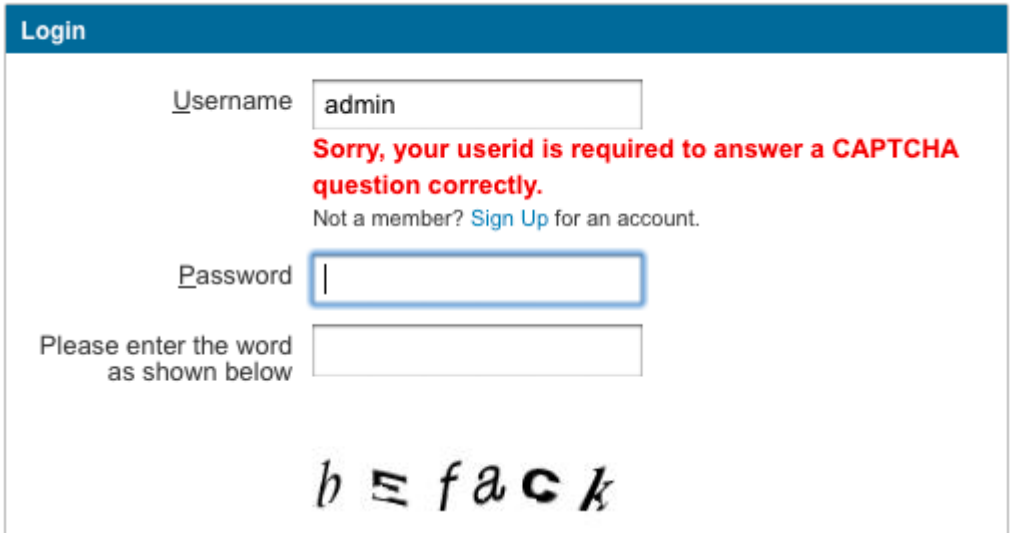
These patches protect JIRA from recently detected privilege escalation and XSS vulnerabilities.

3. Protect Against Brute Force Attack

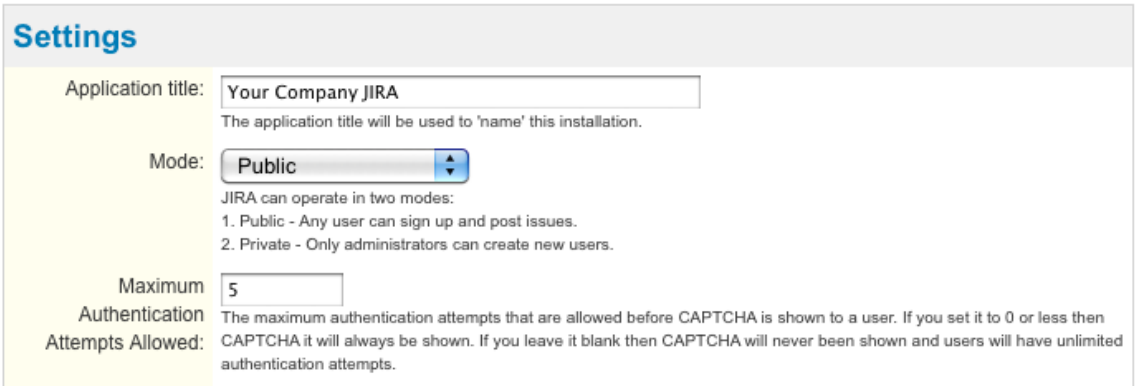
You can also actively protect your systems against repeated unsuccessful login attempts, known as "brute force" login attacks.

3.1 Upgrade to JIRA 4.1

JIRA 4.1 contains built-in protection for brute force attacks by displaying a CAPTCHA after a number of failed authentication attempts.



In JIRA 4.1.1 this option is enabled by default. (Please refer to the JIRA 4.1.1 Upgrade Guide for details.) To enable this protection in JIRA 4.1, log in as an administrator and navigate to Administration -> General Configuration and set the "Maximum Authentication Attempts Allowed" to a small number (e.g. 5).



For more details, see Configuring JIRA Options.

3.2 Enable Brute Force Login Protection on your Web Server

It is possible to also enable brute force login protection on your web server by detecting repeated authentication failures in application logs. Once repeated login failures have been detected, you can set up an automated system to ban access to your web server from that particular IP address.

For more information on how to configure an automated approach to this kind of login prevention, refer to Using Fail2Ban to limit login attempts.

4. Restrict Network Access to Administrative Sections of Applications

An Atlassian application's administration interface is a critical part of the application; anyone with access to it can potentially compromise not only the application instance but the entire machine. As well as limiting access to only users who really need it, and using strong passwords, you should consider limiting access to it to certain machines on the network.

For more information on how to implement Apache blocking rules to restrict access to administrative or sensitive actions in:

- JIRA, refer to Using Apache to Limit Access to the JIRA Administration Interface
- Confluence, refer to Using Apache to limit access to the Confluence administration interface

You can use a similar approach to protecting all Atlassian applications.

5. Restrict File System Access by the Application Server

The application server (e.g. Tomcat) runs as a process on the system. This process is run by a particular user and inherits the file system rights of that particular user. By restricting the directories that can be written to by the application server user, you can limit unnecessary exposure of your file system to the application.

For example, ensure that only the following directories can be written to by JIRA's application server:

- The following subdirectories of your JIRA Installation Directory for 'recommended' JIRA distributions (or for JIRA WAR distributions, the installation directory of the Apache Tomcat application running JIRA):
  - `logs`
  - `temp`
  - `work`
- Your JIRA Home Directory.

For detailed instructions, please see Tomcat security best practices.

6. Disable Jelly

Jelly is disabled in JIRA by default. If you need to use Jelly, you should enable it immediately prior to use and disable it immediately afterwards. See the JIRA Jelly Tags documentation for details.

On this page:

- 1. Use Strong Passwords
  - 1.1 Administrators should use Strong Passwords
  - 1.2 Administrators should have Different Passwords for Different Systems
- 2. Apply JIRA Security Patches
- 3. Protect Against Brute Force Attack
  - 3.1 Upgrade to JIRA 4.1
  - 3.2 Enable Brute Force Login Protection on your Web Server
- 4. Restrict Network Access to Administrative Sections of Applications
- 5. Restrict File System Access by the Application Server
- 6. Disable Jelly
- 7. Configuring Tomcat to use HttpOnly Session ID Cookies
- See Also

*The information on this page does not apply to **JIRA OnDemand**.*

7. Configuring Tomcat to use HttpOnly Session ID Cookies

'Recommended' (formerly Standalone) distributions of JIRA from version 4.1.2 enforce the HttpOnly flag on session ID cookies by default, as a means to minimise the risk of common XSS attacks. For more information about this feature, please refer to the JIRA Security Advisory 2010-06-18.

If you are running the JIRA WAR distribution on Tomcat (version 5.5.27+ or another application server that is unsupported), it is likely that JIRA's session ID cookies will not be transmitted with the HttpOnly flag. Hence, to mitigate the risk of common XSS attacks, we recommend that you configure your application server to transmit HttpOnly session ID cookies.

To configure your JIRA WAR distribution running on Tomcat to use HttpOnly Session ID Cookies:

1. Shutdown the JIRA service running on Tomcat and the Tomcat application server.
2. Open the `context.xml` file of the Tomcat installation running JIRA in a text editor.
   This file is typically located in the `conf` subdirectory of the main Tomcat installation directory.
3. Add the following `Manager` element within the `Context`element of this file:

```
...
<Context>
  ...
  <Manager useHttpOnly="true"/>
  ...
</Context>
...
```

   To disable HttpOnly Session ID cookies, either remove this `Manager` element or change the value of its `useHttpOnly` parameter to `false`.

4. Save your changes to the `context.xml` file and restart JIRA.

See Also

If you suspect that your publicly accessible JIRA installation has been compromised by a security attack, please refer to our detection guide for more information on how to identify signs of attack.

JIRA Cookies

This page lists cookies stored in JIRA users' browsers which are generated by JIRA itself. This page does not list cookies that may originate from 3rd-party JIRA plugins.

Authentication cookies

JIRA uses Seraph, an open source framework, for HTTP cookie authentication. JIRA uses two types of cookies for user authentication:

- The JSESSIONID cookie is created by the application server and used for session tracking purposes. This cookie contains a randon string and the cookie expires at the end of every session or when the browser is closed.
- The 'remember my login' cookie (aka the 'remember me' cookie), `seraph.rememberme.cookie`, is generated by JIRA when the user selects the **Remember my login on this computer** check box on the login page.

You can read about cookies on the Wikipedia page about HTTP cookies.

On this page:

- Authentication cookies
  - The 'remember my login' cookie
- Other JIRA cookies

The 'remember my login' cookie

The 'remember my login' cookie, `seraph.rememberme.cookie`, is a long-lived HTTP cookie. This cookie can be used to authenticate an unauthenticated session. JIRA generates this cookie when the user selects the **Remember my login on this computer** check box on the login page.

Cookie key and contents

By default, the cookie key is `seraph.rememberme.cookie`, which is defined by the `login.cookie.key` parameter in the `<jira-application-dir>/WEB-INF/classes/seraph-config.xml` file of your JIRA Installation Directory.

The cookie contains a unique identifier plus a securely-generated random string.

Use of cookie for authentication

When a user requests a web page, if the request is not already authenticated via session-based authentication or otherwise, JIRA will match the 'remember my login' cookie (if present) against the token stored for the user in the JIRA database (if present).

If the random string matches the value stored in the database and the cookie has not expired, the user is authenticated.

Life of 'remember my login' cookies

You can configure the maximum age of the cookie. To do that you will need to modify the `<jira-application-dir>/WEB-INF/classes/seraph-config.xml` file of your JIRA Installation Directory and insert the following lines below the other `init-param` elements:

```
<init-param>
 <param-name>autologin.cookie.age</param-name>
 <param-value>2592000</param-value> <!-- The value of 30 days in seconds -->
</init-param>
```

Other JIRA cookies

There are several cookies that JIRA uses for a variety of other purposes, such as to enhance JIRA's security and to store basic presentation and browser capability states, including the type of search view that was last used and various other presentation states. JIRA users' authentication details are not stored by these cookies.

Cookie Key	Purpose	Cookie Contents	Expiry
atlassian.xsrf.token	Prevents XSRF attacks. Ensures that during a user's session, browser requests sent to a JIRA server originated from that JIRA server. For more information about XSRF checking by JIRA, see Form Token Checking on the Atlassian Developers site.	Your JIRA server's Server ID, a securely-generated random string (i.e. token) and a flag that indicates whether or not the user was logged in at the time the token was generated.	At the end of every session or when the browser is closed.
jira.issue.navigator.type	Tracks which type of search view was last used (i.e. simple or advanced searching).	A string that indicates the state of your last search view.	Approximately 10 years from the date it is set or was last updated.
AJS.conglomerate.cookie	Tracks which general tabs were last used (e.g. in JIRA's plugin manager) or expansion elements were last opened or closed.	One or more key-value strings that indicate the states of your last general tab views or expansion elements.	One year from the date it is set or was last updated.

UNSUPPORTED_BROWSER_WARNING	Acknowledges that the user has read a message displayed by JIRA indicating that the user's browser is not supported by JIRA.	A string that indicates the user has clicked a button acknowledging they have read the message stating they are using an unsupported browser.	At the end of every session or when the browser is closed.
AJS.thisPage	Indicates that the user's browser does not support local storage. This relates to a mechanism used by JIRA to store field information in search views when the user clicks their browser's back button.	A string that indicates the user's browser does not support local storage.	At the end of every session or when the browser is closed.

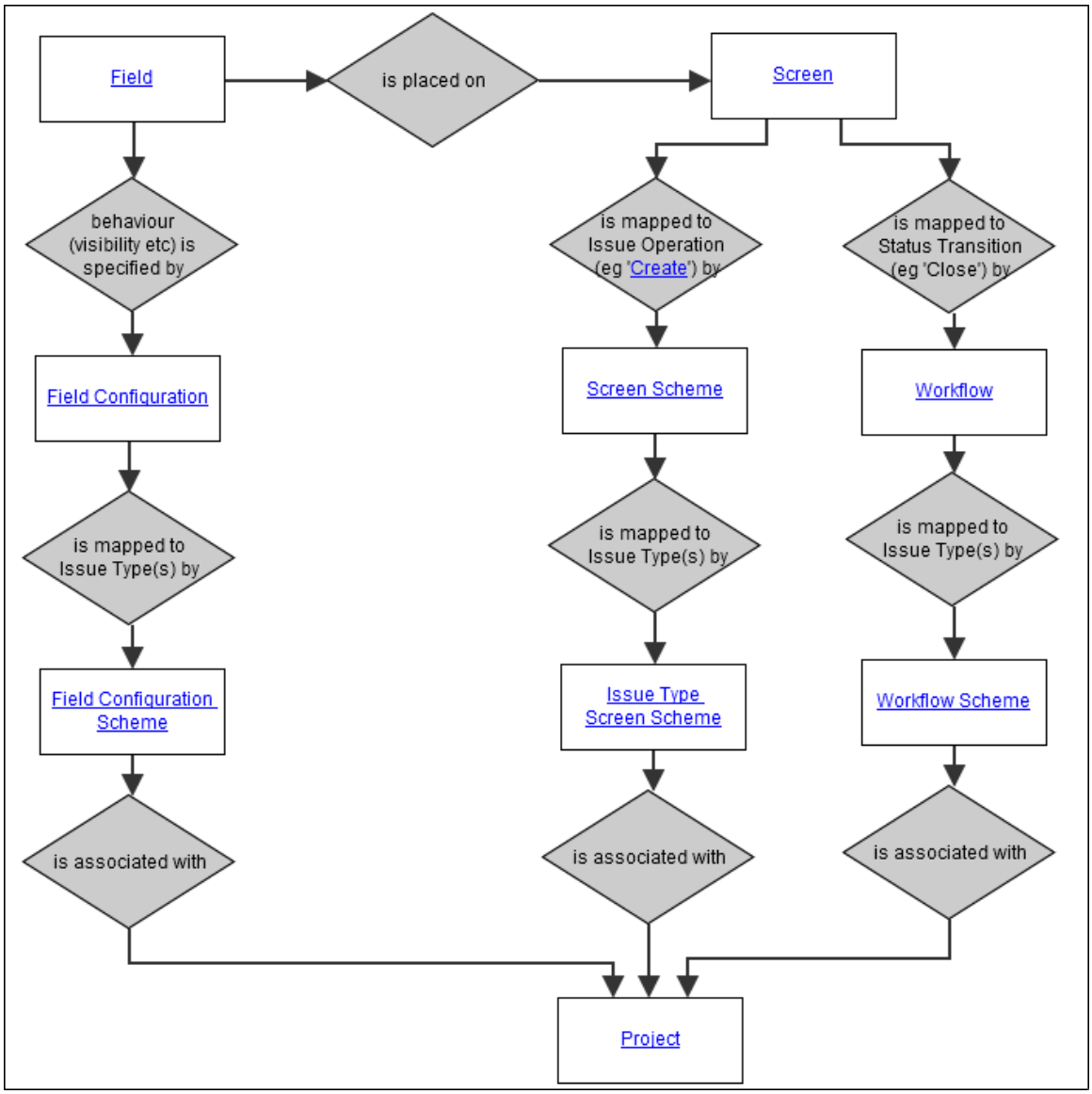
Configuring Fields and Screens

Overview

To help you tailor JIRA to your organisation's needs, JIRA enables you to manipulate the display and behaviour of issue fields ('Summary', 'Description', 'Issue Type', etc). You can:

- Change a field's description
- Make a field hidden or visible
- Make a field required or optional
- Add your own values for 'Issue Type', 'Priority', 'Resolution' and 'Status'
- Create new 'custom' fields
- Enable a rich text renderer for (some) fields
- Position fields on a screen
- Choose which screen should be displayed for each issue operation (e.g. 'Create Issue', 'Edit Issue') or workflow transition (e.g. 'Resolve Issue', 'Close Issue')

Diagram: How Fields, Screens and Workflow interrelate



On this page:

- Overview
  - Diagram: How Fields, Screens and Workflow interrelate
- Concepts

In this section:

- Configuring Built-in Fields
  - Defining 'Issue Type' Field Values
    - Associating Issue Types with Projects
  - Defining 'Priority' Field Values
  - Defining 'Resolution' Field Values
  - Defining 'Status' Field Values
  - Translating Resolutions, Priorities, Statuses and Issue Types
- Adding a Custom Field
  - Configuring a Custom Field
  - Creating Help for a Custom Field
- Specifying Field Behaviour
  - Associating Field Behaviour with Issue Types
  - Configuring Renderers
- Defining a Screen
  - Associating a Screen with an Issue Operation
  - Associating a Screen with an Issue Type

Concepts

Some key JIRA concepts include:

- Field Configuration — a set of definitions for all fields, comprising: each field's description; whether each field is hidden or visible; whether each field is required or optional; and what type of renderer to use for each text field.
- Screen — defines which fields are present on a screen, and their order. (Note that a hidden field can be present on a screen, but will still be invisible.)
- Screen Scheme — associates different screens with different issue operations (e.g. 'Create Issue', 'Edit Issue', 'View Issue').
- Workflow — defines the steps (i.e. statuses) and transitions to other steps that an issue moves through during its lifecycle. Screens can also be mapped to different transitions of a workflow.
- Field Configuration Scheme — associates Field Configurations with issue types, which in turn is applied to projects. This allows you to specify different behaviours for a field, for each type of issue in a given project.
- Issue Type Screen Scheme — associates Screen Schemes with issue types, which in turn is applied to projects. This allows you to specify different screens for a particular operation (e.g. 'Create Issue'), for each type of issue in a given project. For example, you could use one screen when *creating an issue* of type 'Bug', and a different screen when *creating an issue* of type 'Task'.
- Workflow Scheme — associates Workflows with issue types, which in turn is applied to projects. This allows you to specify different workflows for each type of issue in a given project.
- Issue Type Scheme — is applied to projects and defines (or *restricts*) which issue types are available to those projects.

  If the Field Configuration Scheme, Issue Type Screen Scheme and Workflow Scheme associated with a given project contain associations with other issue types that are not specified in the project's Issue Type Scheme, then those other issue types will be ignored by the project since the project's Issue Type Scheme restricts what issue types the project can use.

Configuring Built-in Fields

Each issue has a number of built-in fields, as shown in the sample issue in the *JIRA User's Guide*.

Some of the built-in fields can be customised as follows:

- Defining 'Issue Type' Field Values
  - Associating Issue Types with Projects
- Defining 'Priority' Field Values
- Defining 'Resolution' Field Values
- Defining 'Status' Field Values
- Translating Resolutions, Priorities, Statuses and Issue Types

Defining 'Issue Type' Field Values

JIRA ships with a set of default 'Issue Types' to help you get started. Everyone's needs are different and so JIRA also allows you to add, edit and delete your own custom Issue Types.

Note that you can also:

- control the set of available Issue Types for each project — see Associating Issue Types with Projects.
- control the display order of available Issue Types and the default Issue Type for each project — see Associating Issue Types with Projects.

  Reordering issue types changes the order in which they are displayed to the user who is creating an issue; and the *default* issue type is the one that is displayed in the selection-box (see Creating an Issue).
- associate particular Issue Types with particular Fields, Screens and Workflow — for details see 'Associating Field Behaviour with Issue Types', 'Associating Screens with Issue Types' and 'Activating Workflow', respectively. (Also see the diagram of how these interrelate.)

On this page:

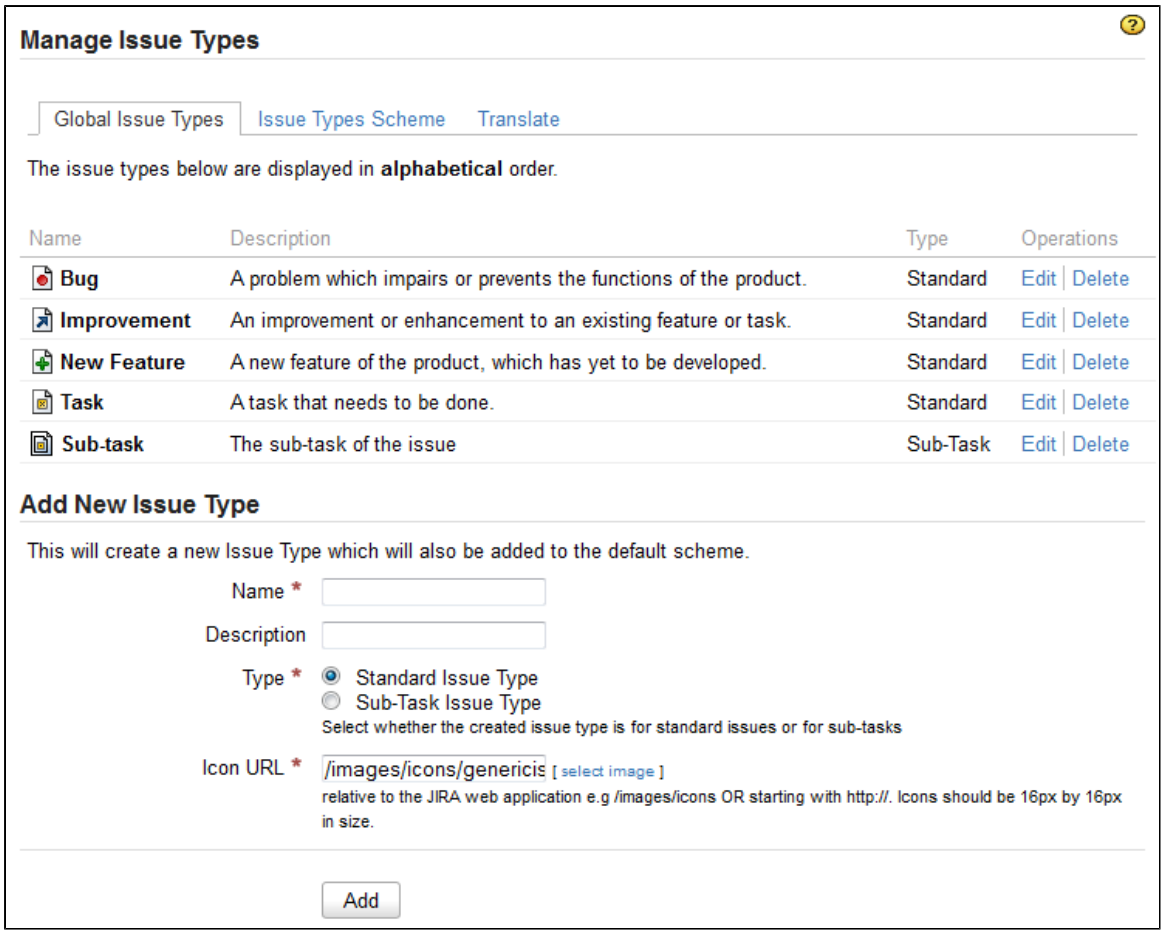
- Creating an Issue Type
- Deleting an Issue Type
- Editing an Issue Type

Creating an Issue Type

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Issue Types'** to open the 'Manage Issue Types' page, which lists all issue types, along with a form underneath to add new issue types.

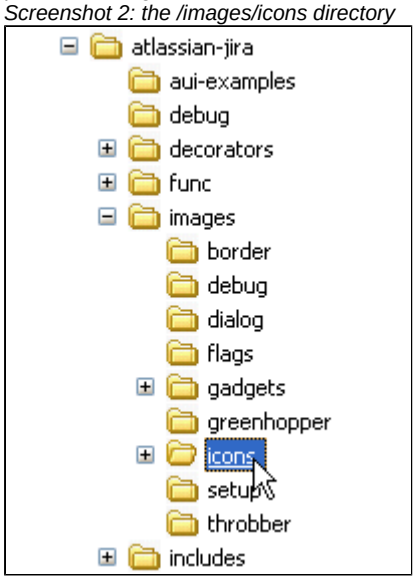
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue types'**

   Screenshot 1: the Manage Issue Types screen



Please Note: You must Enable Sub-tasks to create a sub-task issue type.

3. To add a new Issue Type, fill in the Add New Issue Type form. For the '**Name**', enter a short phrase that best describes your new Issue Type. For the '**Description**', enter a sentence or two to describe when this Issue Type should be used. For the '**Icon URL**' you need to supply the path of a 16-by-16-pixel image that has been placed somewhere inside JIRA's opened .war. We suggest you place it in /images/icons:
Screenshot 2: the /images/icons directory



4. Once you have created your new Issue Type, it will be automatically added to the **Default Issue Type Scheme**. You may want to also add it to other Issue Type Schemes — for more information, see Managing Issue Type Schemes.

Deleting an Issue Type

Before you begin:

- If any issues of the Issue Type you are about to delete exist in your JIRA installation, please ensure this Issue Type has the following requirements (to ensure JIRA prompts you to choose a new Issue Type for those issues):
  - the same Workflow in all Workflow Schemes that are associated with one or more projects.
  - the same Field Configuration in all Field Configuration Schemes that are associated with one or more projects.
  - the same Screen Scheme in all Issue Type Screen Schemes that are associated with one or more projects.
- **Alternatively**, you can simply search for all issues that currently use the Issue Type which you are about to delete and perform a

Bulk Move to change those issues to a different Issue Type.

To delete an Issue Type:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Issue Types'** to open the 'Manage Issue Types' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue types'**
3. Click the '**Delete**' link (in the 'Operations' column) for the issue type that you wish to delete.

Editing an Issue Type

To change the Name, Description or Icon for an Issue Type:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Issue Types'** to open the 'Manage Issue Types' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue types'**
3. Click the '**Edit**' link (in the 'Operations' column) for the issue type that you wish to edit.

Please Note: To reorder an Issue Type, or set it as a default, see Associating Issue Types with Projects. (*Reordering* issue types changes the order in which they are displayed to the user who is creating an issue; and the *default* issue type is the one that is displayed in the selection-box — see Creating an Issue.)

Associating Issue Types with Projects

What is an 'Issue Type Scheme'?

An **Issue Type Scheme** is a sub-set of issue types. An Issue Type Scheme allows you to:

- restrict the set of available issue types for each project.
- control the display order of available Issue Types and the default Issue Type for each project (ie. the Issue Type that is displayed in the selection-box when a user creates an issue).

An Issue Type Scheme can be shared across multiple projects, so that a group of similar projects can share the same issue type settings.

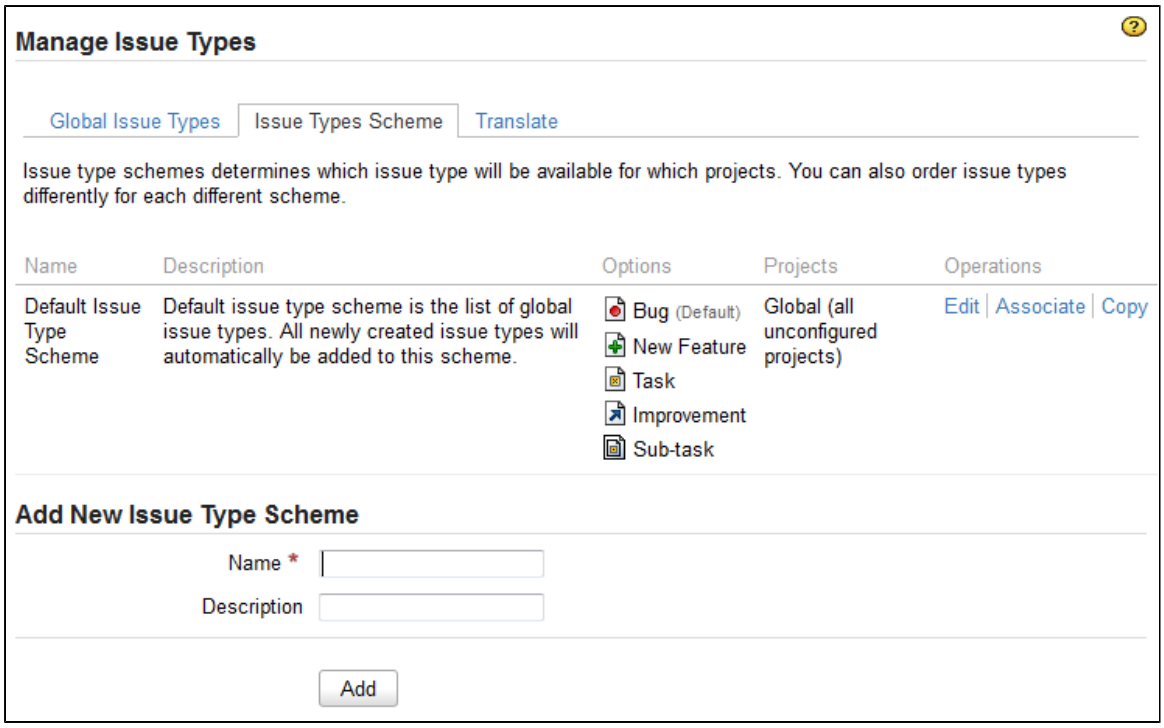
For example, in your company all projects may be one of two types, a development project or a support project. You could then create one scheme called *Task* with issue types *Bug* and *Development* and another called *Support Projects,* with issue types *Development Query* and *Support Request.* You can then associate each scheme with the appropriate project(s), giving your users a different set of issue types depending on which project they decide to create issues in. Your future maintenance workload is minimised, because any change you make to a scheme is made across all projects that are associated with the scheme. In this example, adding a new issue type to all support projects only requires the simple step of adding the issue type to the *Support Projects* scheme.

On this page:

- What is an 'Issue Type Scheme'?
- Managing Issue Type Schemes
  - Creating a new Issue Type Scheme
  - Editing an Issue Type Scheme
  - Associating an Issue Type scheme with Projects
- Choosing a Project's Issue Type Scheme
- Using the Issue Type Migration Wizard

Managing Issue Type Schemes

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Issue Types'** to open the 'Manage Issue Types' page. (For more information about this page, see Defining 'Issue Type' Field Values.)
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue types'**
3. Click the **'Issue Type Schemes'** tab. This will display all existing Issue Type Schemes, their related Issue Types and the associated projects:
   Screenshot 1: the 'Issue Type Schemes' tab on the 'Manage Issue Types' page

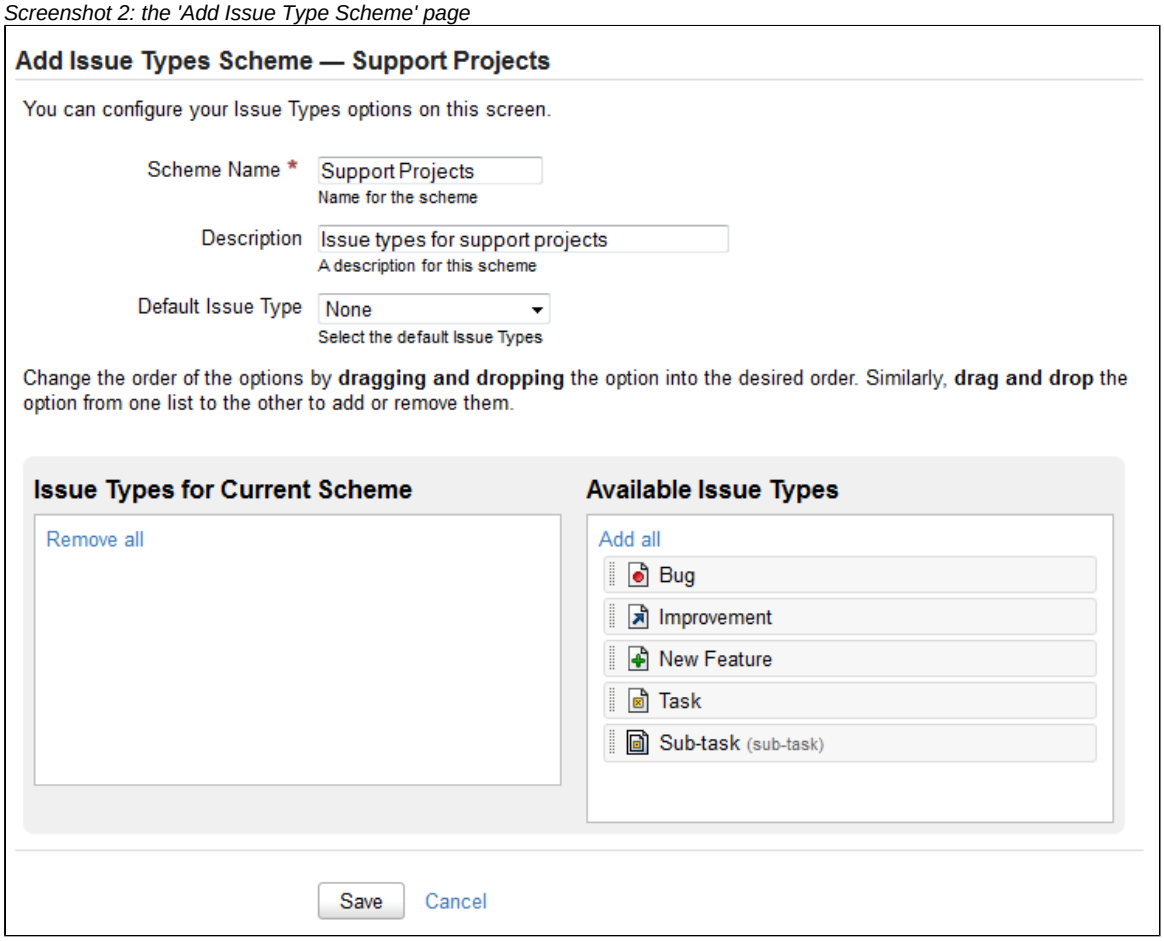


The **'Default Issue Type Scheme'** contains all the issue types that exist in your JIRA system. This scheme is associated with all newly created projects by default. If some of your issue types are not relevant to all of your projects, create one or more new Issue Type Schemes (e.g. 'Development Projects' in this screenshot) as described below, and associate them with the appropriate projects instead of using the **Default Issue Type Scheme**.

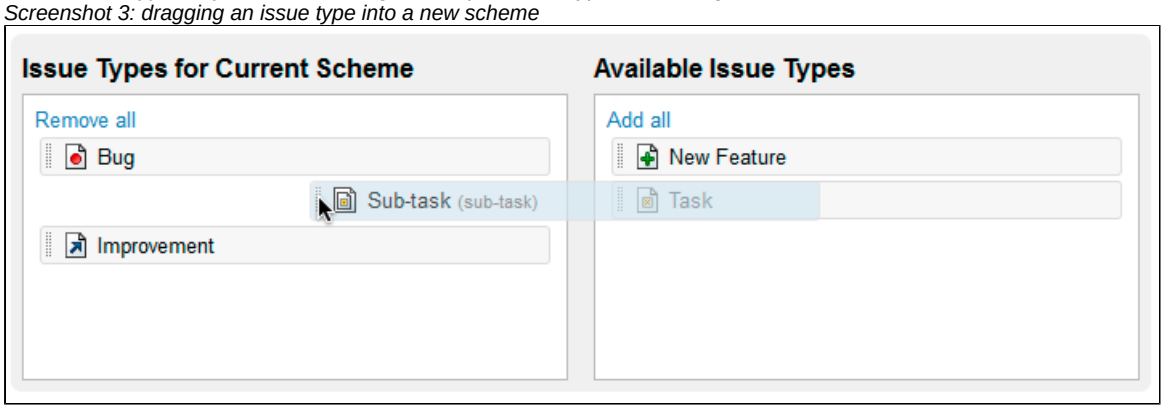
Creating a new Issue Type Scheme

To create a new Issue Type scheme:

1. Go to the 'Issue Type Schemes' tab (see above) and enter the 'Name' and 'Description' for the new Issue Type scheme. Ensure that the name is meaningful as this will be visible to other administrators and will allow them to better reuse the scheme.
2. Click the '**Add**' button to display the 'Add Issue Type Scheme' page.
Screenshot 2: the 'Add Issue Type Scheme' page



3. Set the default for the new scheme from the select list.
   The *default* issue type is the one that is displayed in the selection-box when a user creates an issue (see Creating an Issue).
4. To add issue types to your scheme, drag and drop an issue type from the right-hand list to the left:
   Screenshot 3: dragging an issue type into a new scheme



5. If you need an issue type that does not currently exist, you can easily add this by using the 'Add Issue Type' form at the bottom of the page. This will add the issue type to your JIRA system and also add it to the scheme you are editing.
6. To reorder the issue types, drag and drop them into the preferred positions.
   Reordering issue types changes the order in which they are displayed to the user who is creating an issue.
7. Click the '**Save**' button to create your new scheme.

Editing an Issue Type Scheme

To edit an Issue Type scheme:

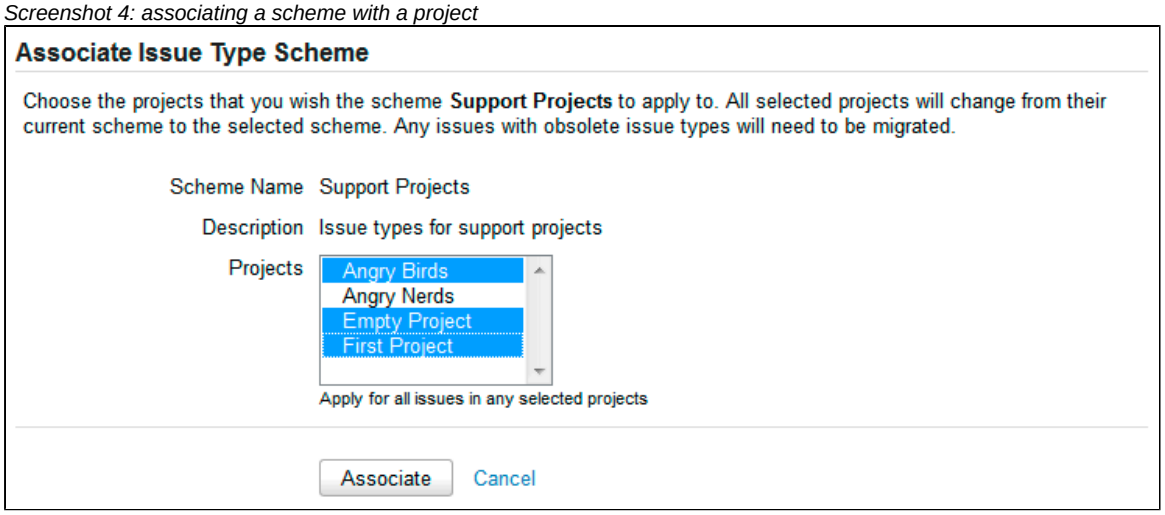
- Go to the 'Issue Type Schemes' tab (see above) and click the '**Edit**' (in the 'Operations' column) for the relevant Issue Type scheme.

The process of editing a scheme is identical to the creation process. You can set defaults, reorder, add and remove issue types from your scheme. However, if you are removing issue types from the scheme and there are issues associated with that issue type, you will be prompted to use the Issue Type Migration Wizard which will move your issues from the obsolete issue type to a valid one. Note that if you cancel out of this process at any time, your changes will not be saved. See below for more information about the Issue Type Migration Wizard.

Associating an Issue Type scheme with Projects

To associate an issue type scheme with one or more projects:

1. Go to the 'Issue Type Schemes' tab (see above) and click '**Associate**' (in the 'Operations' column) for the relevant Issue Type scheme.
2. Choose the projects that you wish your scheme to apply to. All selected projects will change from their current scheme to the selected scheme.
   Screenshot 4: associating a scheme with a project



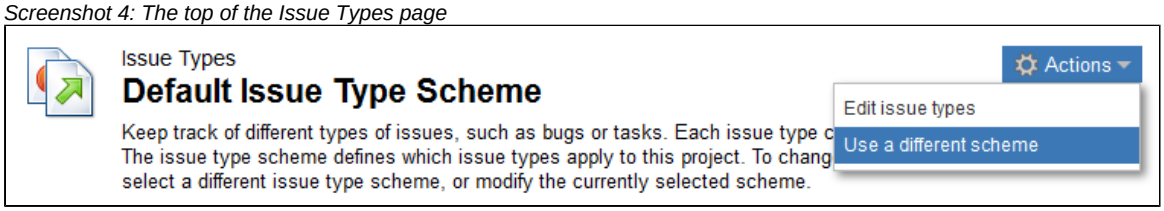
If the new scheme does not have an issue type that was present in the old scheme, you will be asked to use the Issue Type Migration Wizard (see below) to migrate the issues to a new issue type.

Choosing a Project's Issue Type Scheme

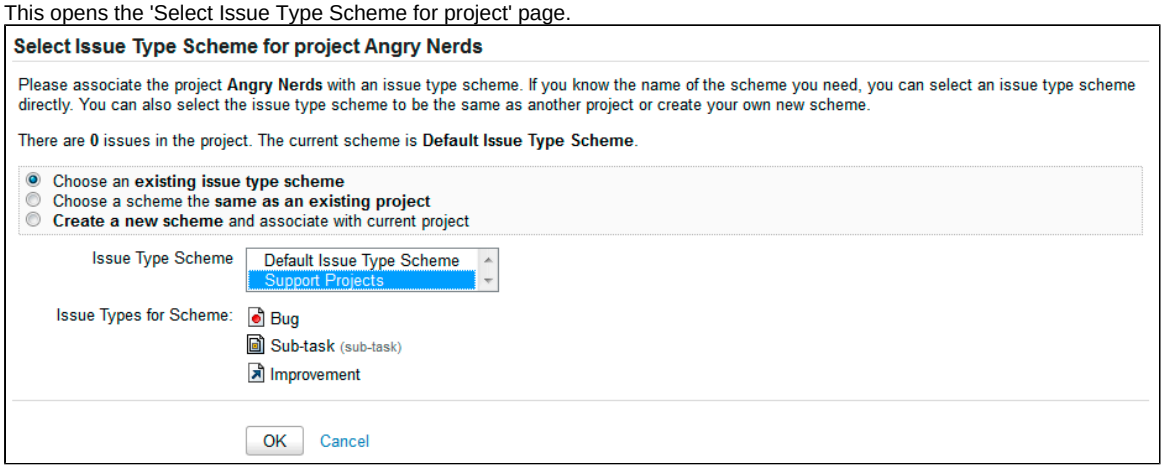
You may want to change a project to use a different set of issue types. To do this, choose a different Issue Type Scheme for the project as follows:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Click the '**Administration**' link at the top of the screen.

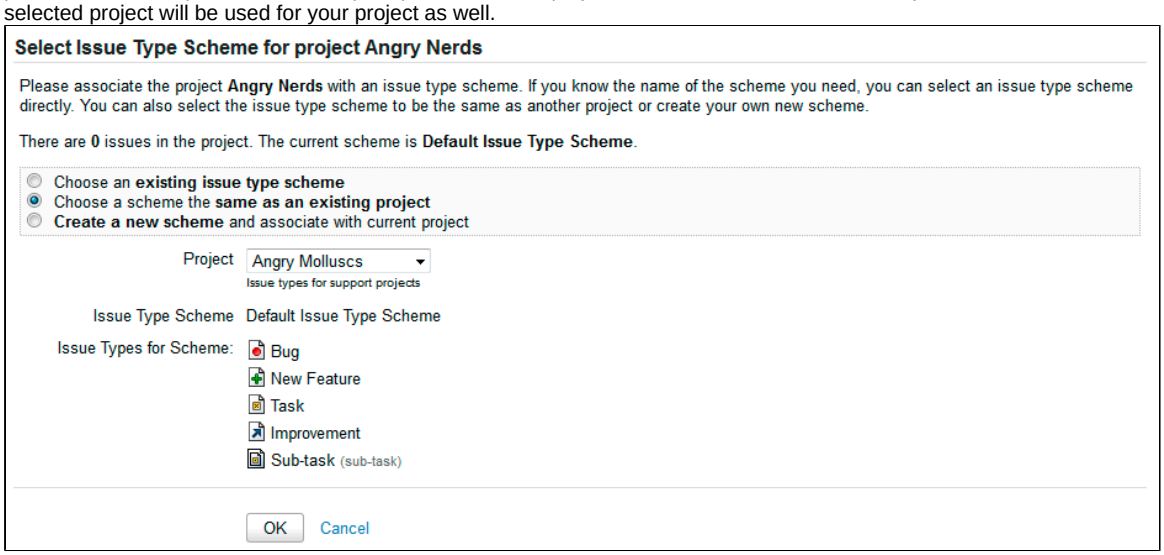
3. Click **'Projects'** and select the project of interest. You will now see the 'Project Summary' screen (see Defining a Project).
4. In the 'Issue Types' section, click the name of the current Scheme (e.g. 'Default Issue Type Scheme') to display the details of the project's issue type scheme.
5. Click the '**Actions**' dropdown menu and choose '**Use a different scheme**'.
   Screenshot 4: The top of the Issue Types page



This opens the 'Select Issue Type Scheme for project' page.

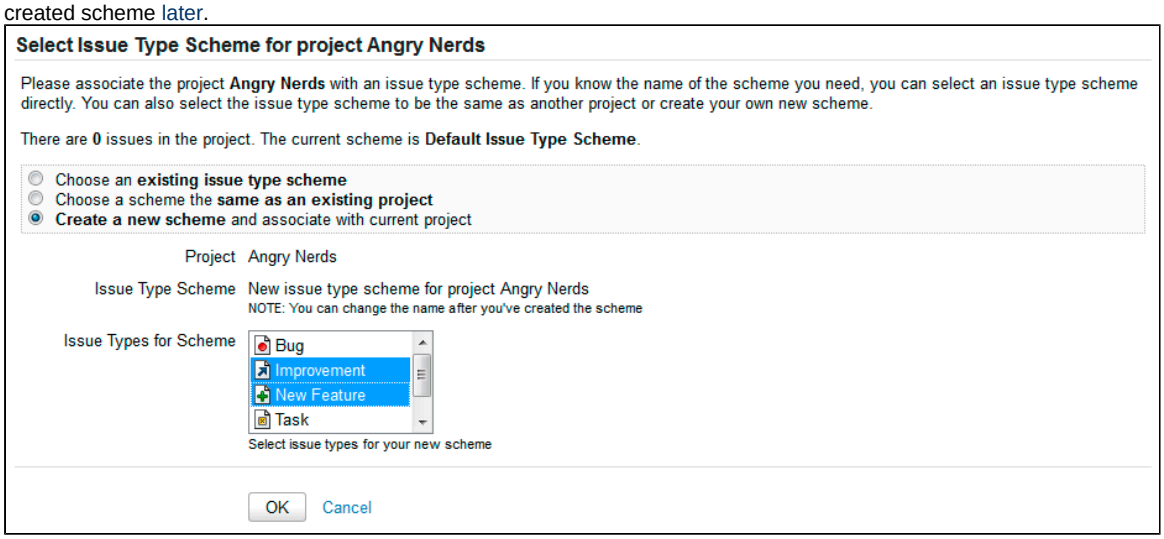


6. There are three ways you can select your issue type scheme. Select the radio button that is most relevant:
   a. *Choose an existing issue type scheme* — If you know the name of your scheme (e.g. Support Projects), you can immediately choose it from the list. You will see a preview of issue types that would be available for your project as well as the description of the scheme.
   b. *Choose a scheme that is the same as an existing project* — If you do not know the name of the scheme you would like to use, but you do know the name of the project whose set of issue types you wish to use for the project you are editing, please select this option. You will be prompted to select a project and the scheme that is currently associated with the selected project will be used for your project as well.



   c. *Create a new scheme and associate with current project* — Select this option if you can't find any existing scheme that fits your needs and would like to quickly create a new scheme. Simply select the relevant issue types for your project and a new

scheme will be created with the default name and order. You can edit the name, default value and order of the newly created scheme later.



7. If after you make your changes there are any issues in the selected project that will have obsolete issue types, they will have to be migrated with the Issue Type Migration Wizard.

Using the Issue Type Migration Wizard

The Issue Type Migration Wizard allows you to migrate issues from an obsolete issue type to a valid issue type. The wizard will be triggered whenever an action (e.g. editing a project's issue type scheme) results in an issue type becoming obsolete (not available in the scheme).

The wizard is similar to the Bulk Move function except for that you can't change the project of the issues. The major steps are:

1. Overview — provides a summary of the issues that will require migration
2. Choose Issue Type
3. Set new status
4. Set field values
5. Confirmation

Steps 2 to 4 will be repeated for each issue type that requires migration. After you have migrated all the issues you'll see a summary of changes that will occur. If you click the '**Confirm**' button, the wizard will migrate your issues to the new issue types and then complete your action.

Please refer to the Bulk Move documentation for more information on status changes and setting of fields values.

Defining 'Priority' Field Values

An issue's *priority* is its importance in relation to other issues.

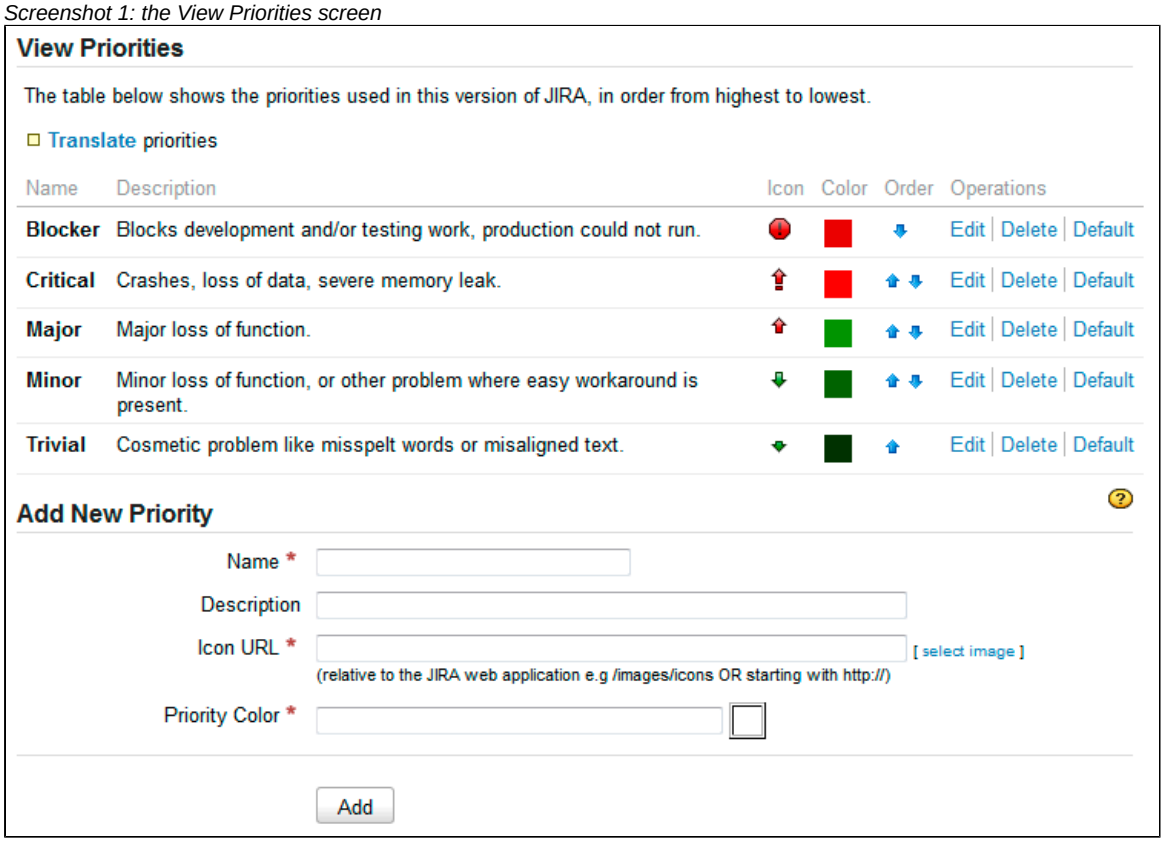
JIRA ships with a set of default priorities. You can modify these or add your own as follows.

Defining a new priority

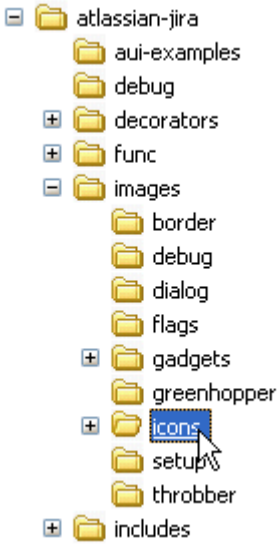
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Issue Priorities'** to open the '**View Priorities**' page, which lists the currently-defined Priorities and the 'Add New Priority' form below.

Keyboard shortcut: **'g' + 'g' +** start typing **'priorities'**

Screenshot 1: the View Priorities screen



3. In the 'Name' field, type a word or two to describe your new priority. (The Name will appear in the drop-down field when a user creates or edits an issue).
4. *(Optional)* In the 'Description' field, type a sentence or two to describe when this priority should be used.
5. In the 'Icon URL' field, specify an image file to represent this priority. The dimensions of the image must be 16-pixels by 16-pixels. You can either type a URL, or click the **'Select image'** link to browse to a file location somewhere inside your JIRA installation directory, usually in /images/icons:
   Screenshot 2: the /images/icons directory



6. In the **'Priority Color'** field, specify a colour to represent this priority. You can either type the HTML colour code, or click the box at the right of the field to select from a colour chart.
7. Click the **'Add'** button.

On this page:

- Defining a new priority
- Editing a priority
- Re-ordering priorities
- Translating priorities
- Deleting a priority

Editing a priority

1. Go to the **'View Priorities'** page as described in **'Adding a priority'** (above).
2. Click the **'Edit'** link (in the 'Operations) column corresponding to the priority you wish to edit.
3. Update the fields as described under **'Defining a new priority'** (above), then click the **'Update'** button.

Re-ordering priorities

Re-ordering priorities changes the order in which they appear in the drop-down list when a user creates or edits an issue.

1. Go to the **'View Priorities'** page as described in **'Adding a priority'** (above).
2. To re-order the priorities, click the arrows in the **'Order'** column:
    - Click the up-arrow to move a priority higher up in the list.
    - Click the down-arrow to move a priority lower down in the list.

Translating priorities

To translate your priorities into another language, please see Translating Resolutions, Priorities, Statuses and Issue Types.

Deleting a priority

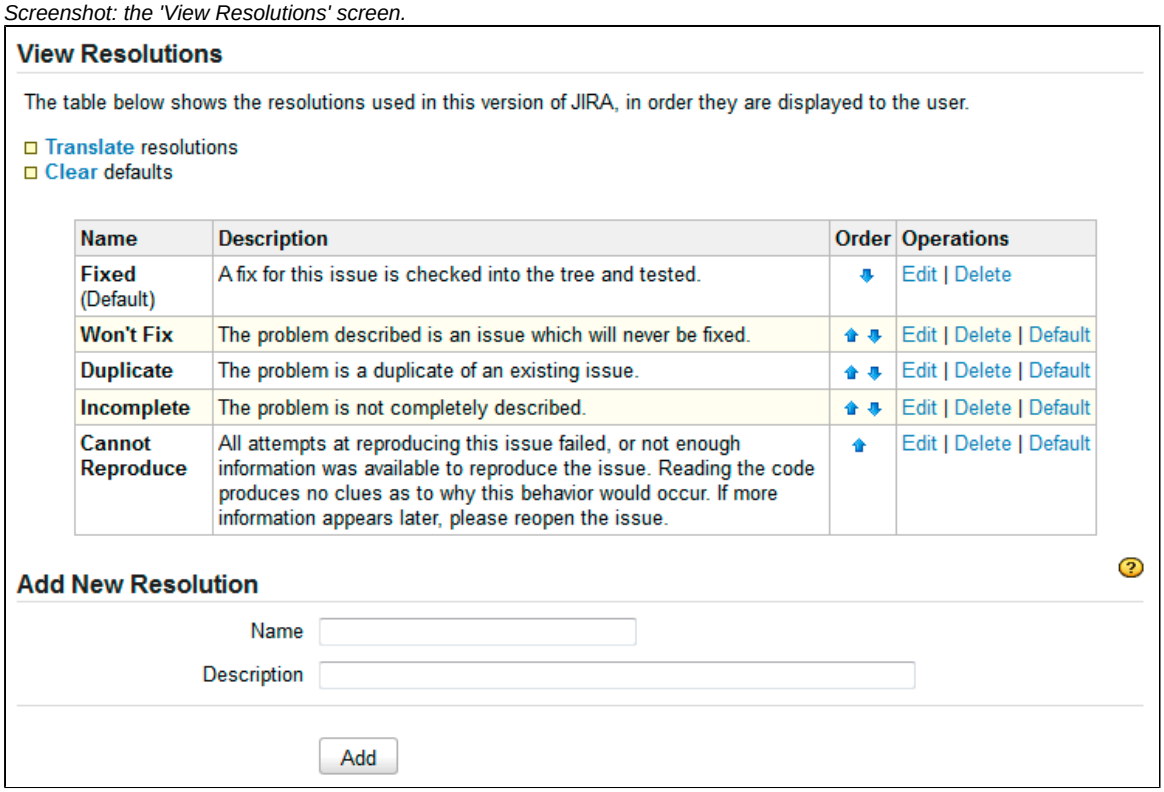
1. Go to the **'View Priorities'** page as described in steps 1-4 of **'Adding a priority'** (above).
2. Click the **'Delete'** link (in the 'Operations' column) corresponding to the priority you wish to delete.

Defining 'Resolution' Field Values

Resolutions are the ways in which an issue can be closed. JIRA ships with a set of default resolutions, but you can add your own as follows.

Defining a new resolution

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Resolutions'** to open the 'View Resolutions' page, which lists the standard resolutions, along with a form underneath to add new resolutions.
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue types'**
   Screenshot: the 'View Resolutions' screen.



3. To add a new resolution, fill in the 'Add New Resolution' form.
    - For the 'Name' field, enter a short phrase that best describes your new resolution.
    - For the 'Description' field, enter a sentence or two to describe when this resolution should be used.

The 'View Resolutions' screen can be used to edit, delete, set as default, and re-order the resolutions as they are displayed to the user who is resolving an issue.

Defining 'Status' Field Values

Statuses are used to represent the position of the issue in its workflow. A workflow represents a business process, represented as a set of

stages that an issue goes through to reach a final stage (or one of the final stages). Each stage in the workflow (called a *workflow step)* is linked to an *issue status,* and an issue status can be linked to only one workflow step in a given workflow.

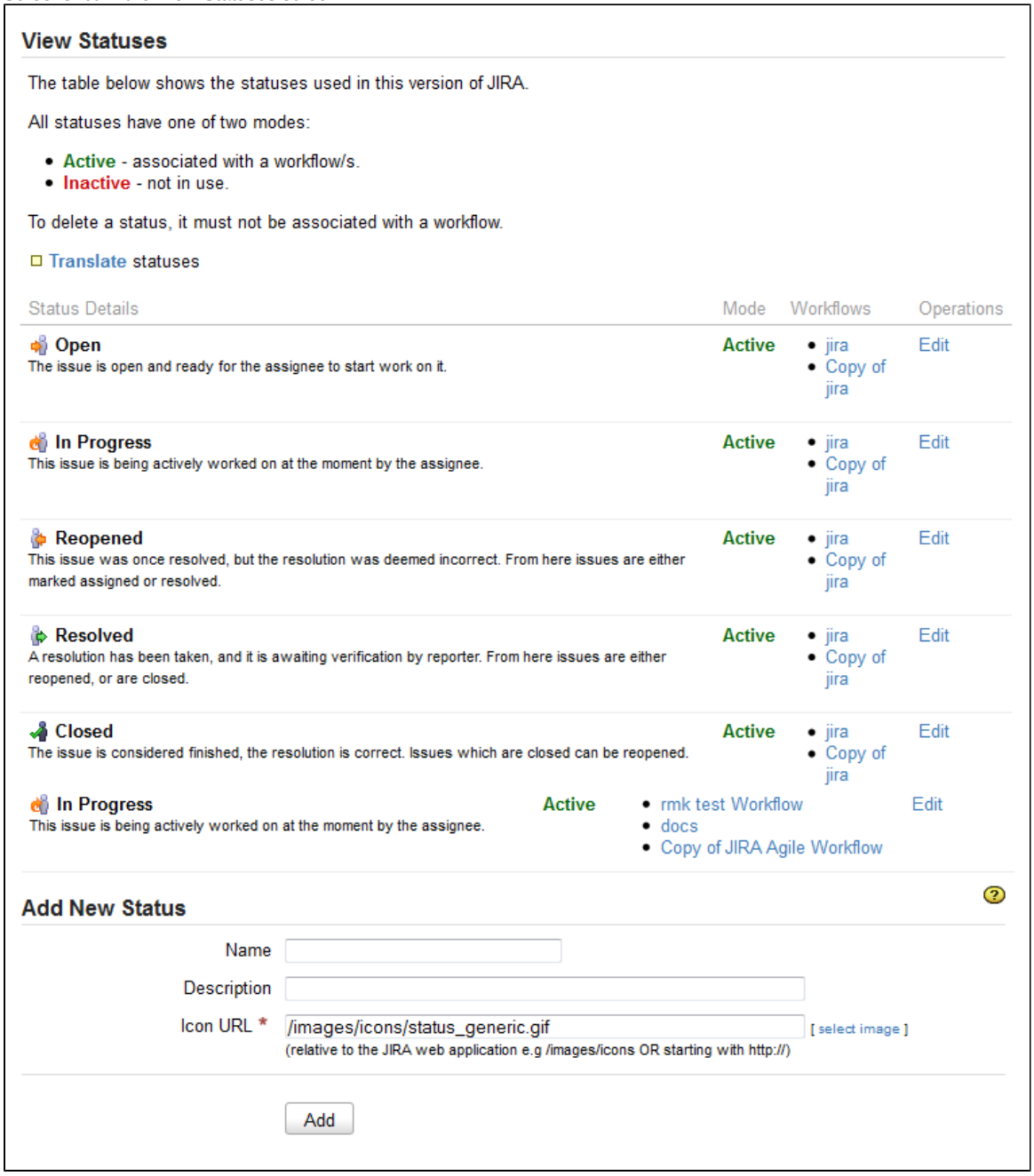
JIRA ships with a set of default statuses that are used by the default workflow. You can add your own statuses and customise the workflow, as well as change the names, descriptions and icons of existing Statuses.

On this page:

- Defining a New Status
- Deleting a Status

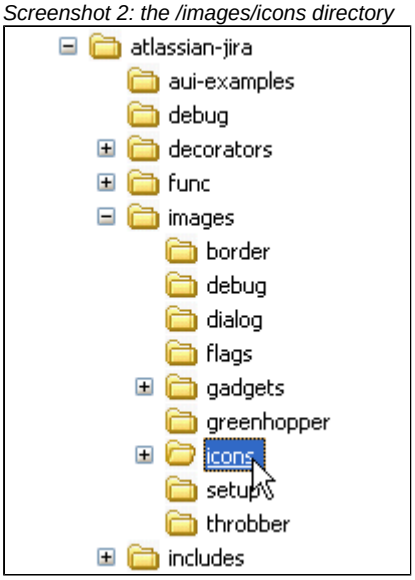
Defining a New Status

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Statuses'** to open the 'View Statuses' page, which lists all statuses, along with a form underneath to add a new status.
   Keyboard shortcut: **'g' + 'g' +** start typing **'statuses'**
   Screenshot 1: the View Statuses screen



3. To add a new status, fill in the 'Add New Status' form.
   - In the 'Name' field, specify a short phrase that best describes your new status.
   - In the 'Description', add a sentence or two to describe what workflow step this status represents.
   - In the 'Icon URL' field, specify the path of an icon-sized image (no greater than 30x35 pixels) which has been placed somewhere inside JIRA the JIRA Installation Directory (or is accessible from an HTTP address).

We recommend placing them in `/images/icons`:
Screenshot 2: the /images/icons directory



JIRA ships with a number of images that can be used as status icons. These images are located in the `/images/icons`directory inside the JIRA Installation Directory and includes:

- status_assigned.gif
- status_closed.gif
- status_document.gif
- status_down.gif
- status_email.gif
- status_generic.gif
- status_information.gif
- status_inprogress.gif
- status_invisible.gif
- status_needinfo.gif
- status_open.gif
- status_reopened.gif
- status_resolved.gif
- status_trash.gif
- status_unassigned.gif
- status_up.gif
- status_visible.gif

Next steps:

Now you will need to associate your new status with a workflow 'step'. See Configuring Workflow.

Deleting a Status

The 'View Statuses' page can be used to edit and delete Statuses. Please note that only **Inactive** statuses (i.e. statuses that are not used in any workflow) can be deleted.

A 'Delete' link for deleting a Status will only appear next to the 'Edit' link of an **Inactive** status.

Translating Resolutions, Priorities, Statuses and Issue Types

Further extending JIRA as an internationalisable issue manager, it is possible to easily specify a translated name and description for all values of the following 'issue constants':

- the Issue Type field
- the Status field
- the Resolution field
- the Priority field

This allows you to specify a translation set for each available language — providing each user with a more complete translation in their own chosen language. The translated field names and descriptions appear throughout JIRA, e.g. in reports, gadgets and all issue views.

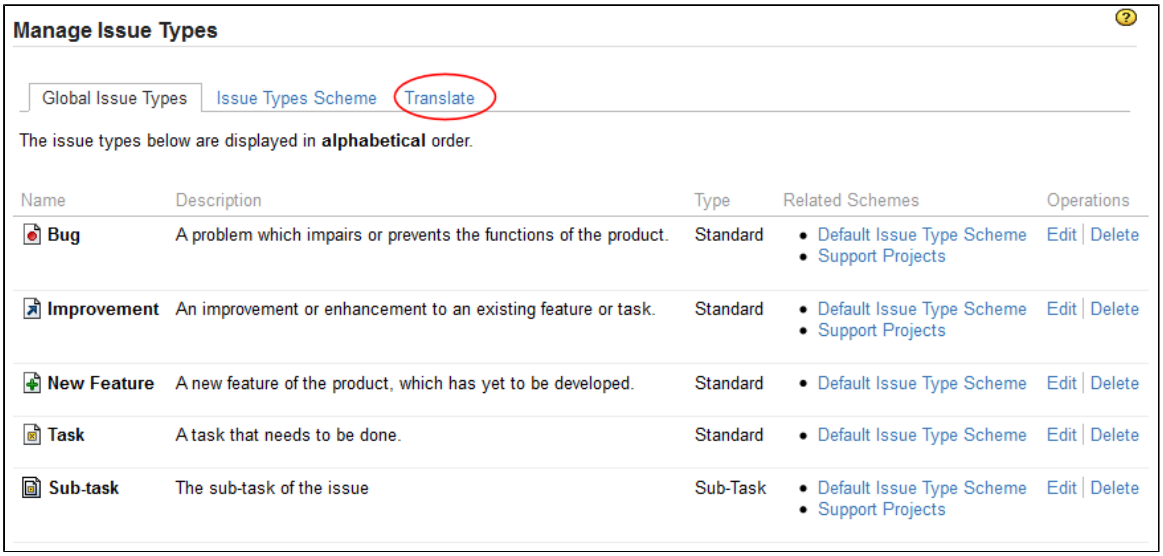
> *The information on this page does not apply to **JIRA OnDemand**.*

Translating an Issue Constant

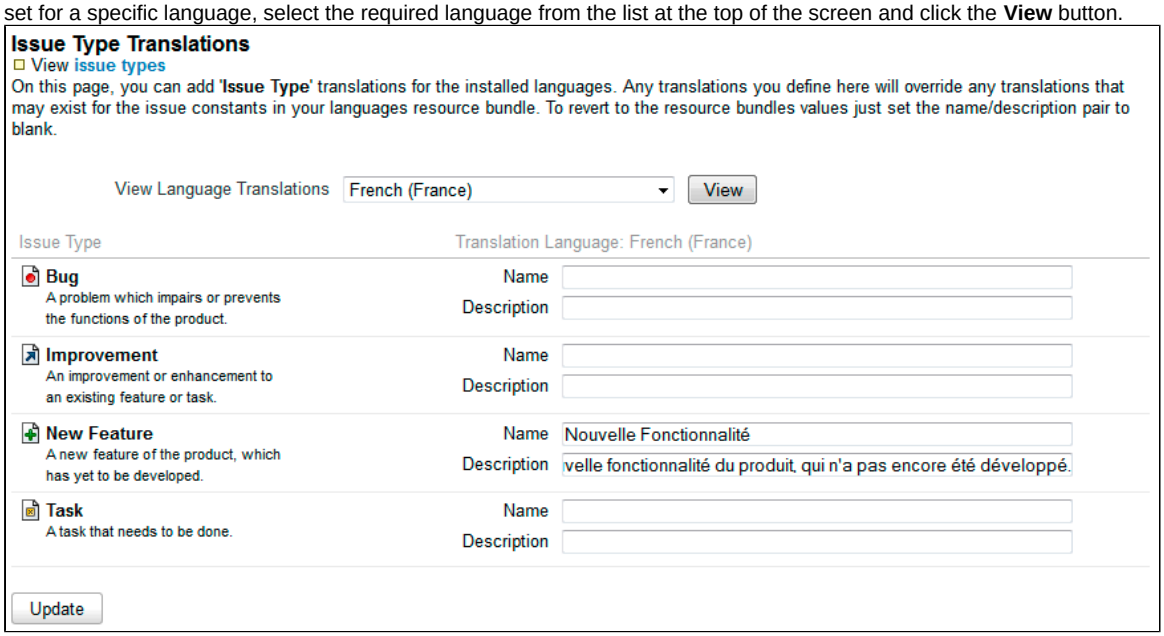
Each issue constant can be configured to have a translation set for each available language in your JIRA system. If no translation has been configured for a particular language, the default issue constant name and description are displayed.

To specify a translation:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Click the '**Translate**' link located on the issue constant management screen — i.e.
   - the 'Manage Issue Types' screen,
   - the 'View Statuses' screen,
   - the 'View Resolutions' screen or
   - the 'View Priorities' screen.



3. The issue constant translation screen displays the translation set for the currently selected language. To view/update a translation set for a specific language, select the required language from the list at the top of the screen and click the **View** button.



4. The currently selected language is displayed above the translation set table.
5. A translated name and description set can be specified for each type of issue constant. Once all translations have been entered, the translation set can be saved by clicking the **Update** button at the bottom of the translation table.
6. The process can be repeated for all of the issue constants — i.e. Issue Type, Status, Resolution and Priority fields.
7. The translated issue constant name and description will be displayed throughout JIRA, e.g. in reports, gadgets and all issue views.

The default issue constant name and description are displayed if a translation has not been specified.

Adding a Custom Field

Custom Fields Overview

To help you tailor JIRA to your organisation's needs, JIRA enables you to add custom fields in addition to the built-in fields. For example, if you needed to capture information about the database that each issue relates to, you could add a custom field called 'Database'.

You can choose the most suitable custom field type (see below) for your purposes. For example, you could choose to create this field as a Free Text Field, in which users can type whatever they wish, or as a Select List, which will force users to select from a list of pre-defined options.

Once you have created a new custom field (see below), you will need to add it to one or more screens so that it is available to users. For more information about how field and screens interrelate, please see Configuring Fields and Screens.

Custom fields are always optional fields. This means that you can create a new custom field without requiring existing issues to be changed. The existing issues will contain no value for the new custom field, even if a default value is defined.

On this page:

- Custom Fields Overview
  - Custom Field Types
  - Search Templates
  - Custom Field Context
- Adding a Custom Field
- Next Steps

Custom Field Types

JIRA ships with over 20 custom field types and you can find many more in the Plugin Exchange (e.g. the JIRA Toolkit). A sample of the types are listed as follows:

Custom Field Type	Description
Cascading Select	Multiple select lists where the options for the second select list dynamically updates based on the value of the first
Date Picker	Input field allowing input with a date picker and enforcing valid dates
Date Time	A custom field that stores dates with a time component.
Free Text Field (unlimited text)	Multiple line text-area enabling entry of longer text strings
Group Picker	Choose a user group using a popup picker window.
Labels	Input field allowing labels to be added an issue. E.g. If you are using GreenHopper, the 'Epics' feature is implemented via a 'Labels' custom field.
Multi Checkboxes	Checkboxes allowing multiple values to be selected
Multi Group Picker	Choose one or more user groups using a popup picker window.
Multi Select	Select list permitting multiple values to be selected
Multi User Picker	Choose one or more users from the user base via a popup picker window.
Number Field	Input field storing and validating numeric (floating point) values
Project Picker	Select list displaying the projects viewable by the user in the system
Radio Buttons	Radio buttons ensuring only one value can be selected

Select List	Single select list with a configurable list of options
Single Version Picker	Choose a single version from available versions in the project.
Text Field	Basic single line input field to allow simple text input of less than 255 characters
URL Field	Input field that validates a valid URL
User Picker	Choose a user from the user base via a popup picker window.
Version Picker	Choose one or more versions from available versions in the project.

To build your own custom field types, see the tutorial at the JIRA Developer Documentation.

Search Templates

Search Templates are responsible for indexing a custom field as well as making it searchable through the Issue Navigator (note that custom fields are not searchable via QuickSearch). Each of the default custom field types has a related preconfigured search template.

When you create a new custom field (see below) you will need to specify its Search Template.

Custom Field Context

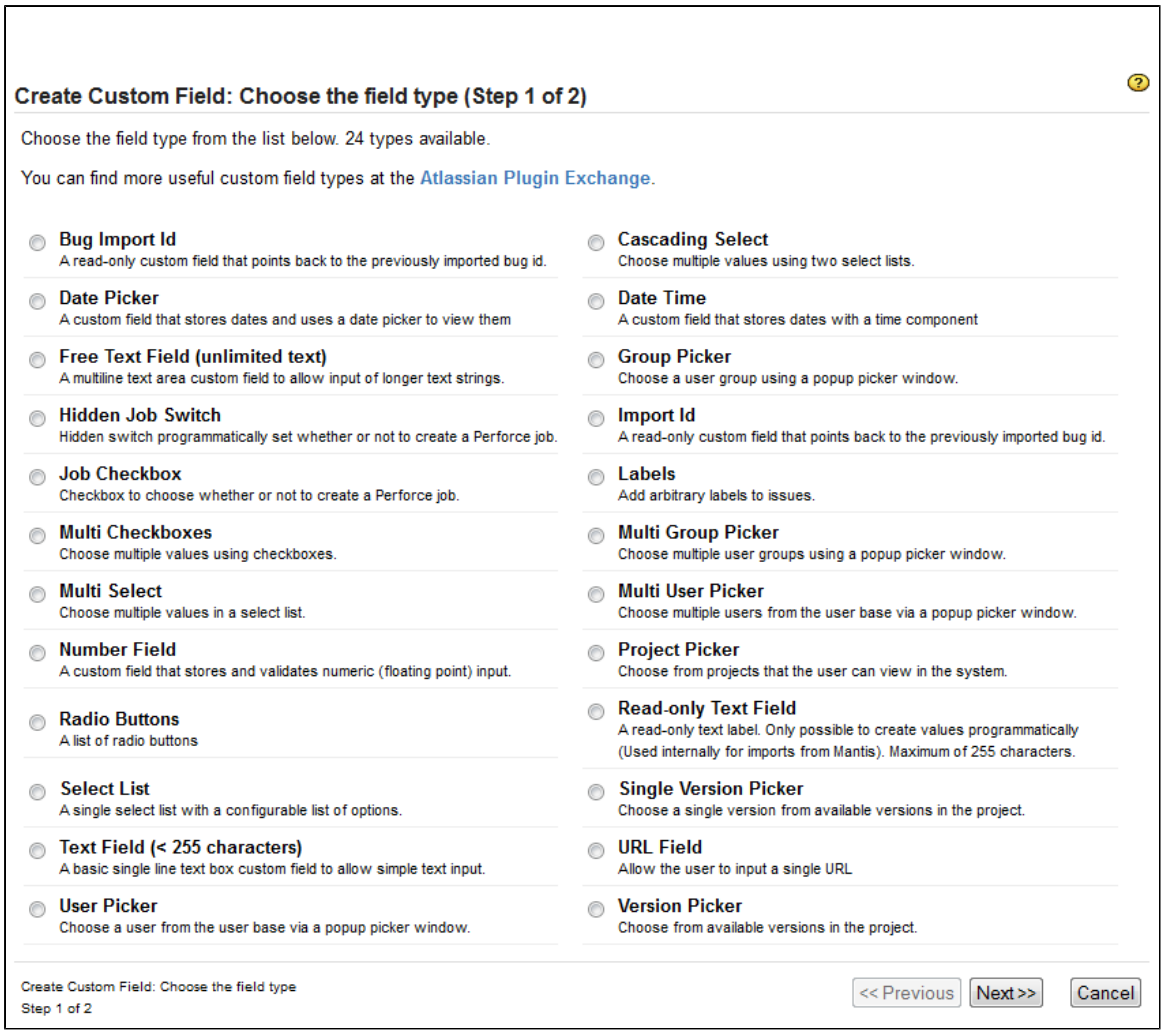
The custom field ***context*** allows your custom field to be configured differently for numerous different combinations of issue types and projects. For example, your custom field could have different default values for different projects (and/or issue types).

When you create a new custom field (see below) you will need to either select the applicable issue type(s) and project(s), or define the custom field to be global. You can change this later if required — see Configuring a Custom Field.

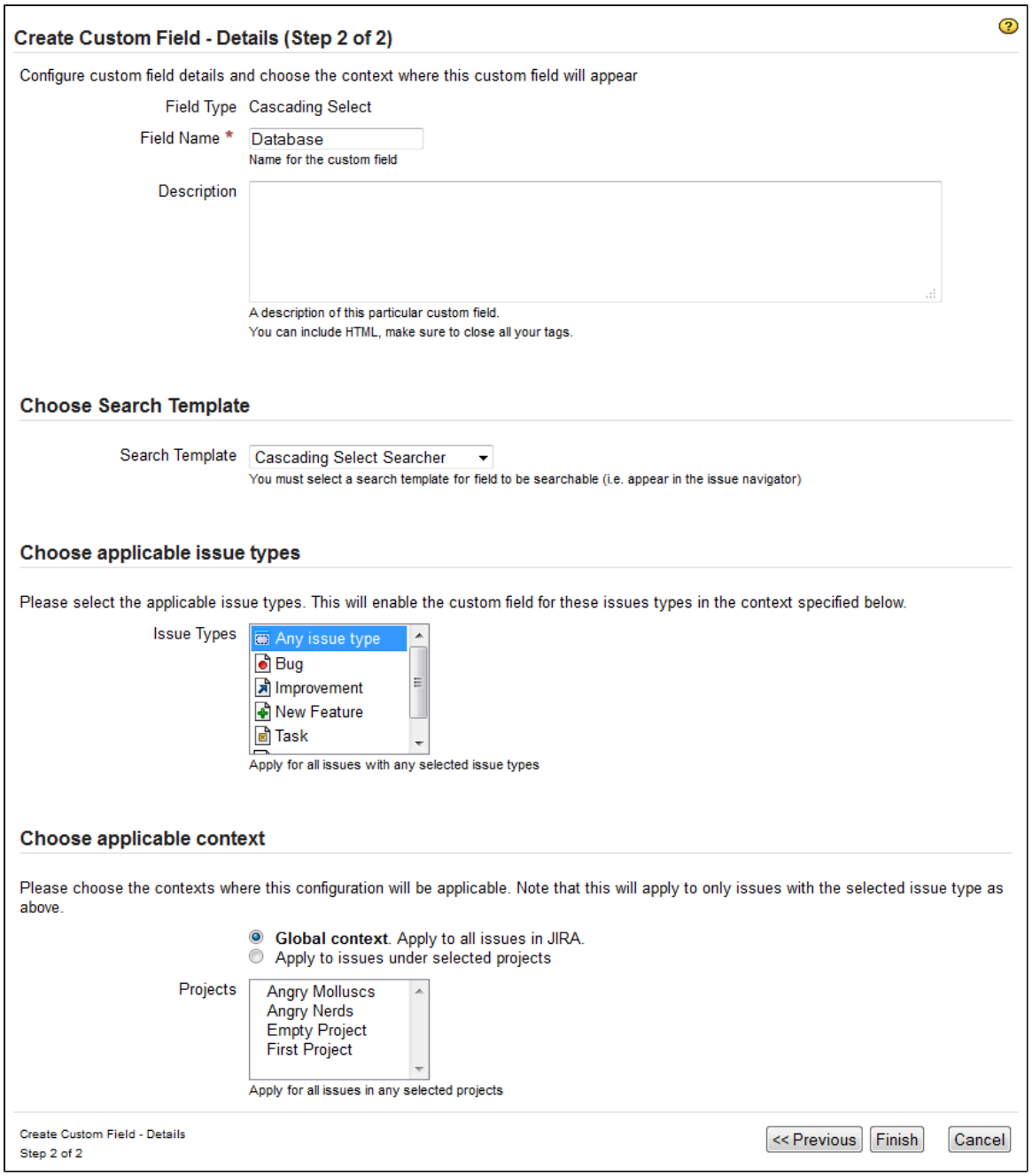
Adding a Custom Field

To create a new field, associate it with a context, and add it to a screen:

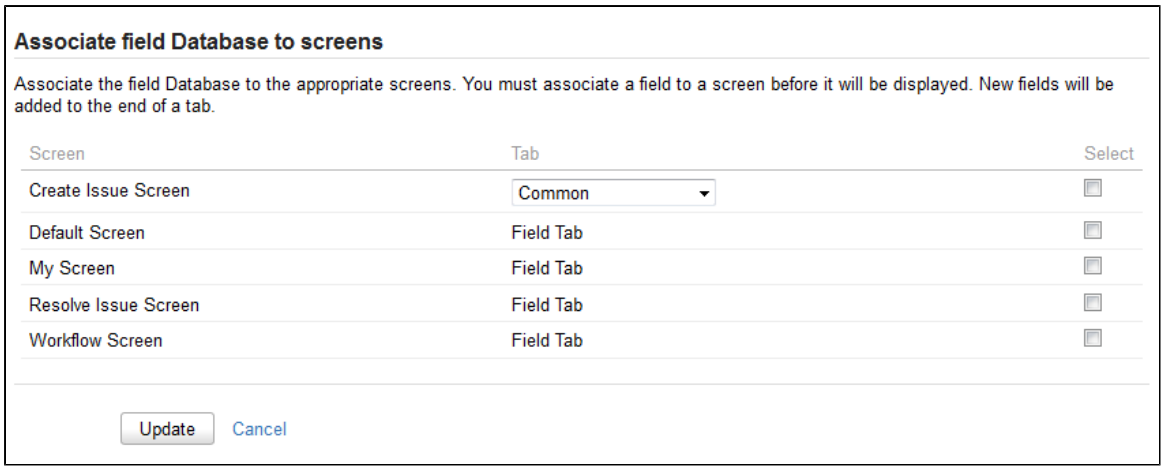
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Click the '**Add Custom Field**' link on the presented page and the 'Add Custom Field - Step 1' screen will be displayed.



4. Select the appropriate '**Field Type**' field the list.
5. Click the '**Next**' button and the '**Add Custom Field - Step 2**' screen will be displayed.



6. Fill in the '**Field Name**' and '**Field Description**'. The Field Name will appear as the custom field's title in both entering and retrieving information on issues. The Field Description is displayed beneath the data entry field when entering new issues and editing existing issues, but not when browsing issues.
7. Select an appropriate '**Search Template**' (see above). Pre-configured search templates are available for each shipped custom field type. A description of each search template will appear next to the select list when you select one.
8. Select one or any number of '**Issue Types**' to which this custom field will be available. Alternatively, select '**Any issue type**' to make the custom field available to all Issue Types. You can change this in the future if you need to.
9. Select the applicable context, that is, the '**Project(s)**' to which the custom field will be available. Alternatively, select '**Global context**' to make the custom field available to all projects.
    - If issue types were chosen, the custom field will only appear for those issue types for the selected project(s).
10. Click the '**Finish**' button.
11. This will bring you to the screen association page:



12. Select a screen, or screen tab, on which to display your newly created custom field. You must associate a field with a screen before it will be displayed. New fields will be added to the end of a tab.
13. Click the '**Update**' button. You will return you to the View Custom Fields page, which displays a summary of all custom fields in your JIRA system. You can edit, delete or configure custom fields here. This page is also directly accessible from the menu bar to the left of all Administration pages. For details please see Configuring a Custom Field.

Next Steps

Once you have created your new custom field, you can configure its:

- default value
- options (for custom fields of type Select List, Multi Select or Cascading Select)
- context (see above)

For details, see Configuring a Custom Field.

Configuring a Custom Field

You can modify each of the custom fields in your JIRA system by changing the following:

- **Name** — the label that appears to the left of the custom field when it is displayed to a user. See below.
- **Description** — the Help text that appears below the custom field when it is displayed in the Simple Search column. See below.
- **Search Template** — the mechanism for making a custom field searchable. See below.
- **Default Value** — the default value of the custom field when it is first displayed to a user. See below.
- **Options** *(for Select and Multi-Select fields only)* — the values from which a user can choose. See below.
- **Context** — the combination of project(s) and issue type(s) for which a given **Default Value** and **Options** will apply. See below.
  You can create multiple Contexts, allowing you to specify different Default Values and Options for different combinations of projects and/or issue types.
- **Screens** — the screen(s) on which the custom field will appear when an issue is created, edited or transitioned through workflow. See below (also see Defining a Screen).
- **Renderers** — *(for certain types of fields only)* — see Configuring Renderers and Specifying Field Behaviour.
- **Hide/Show** — see Specifying Field Behaviour.
- **Required/Optional** — see Specifying Field Behaviour.

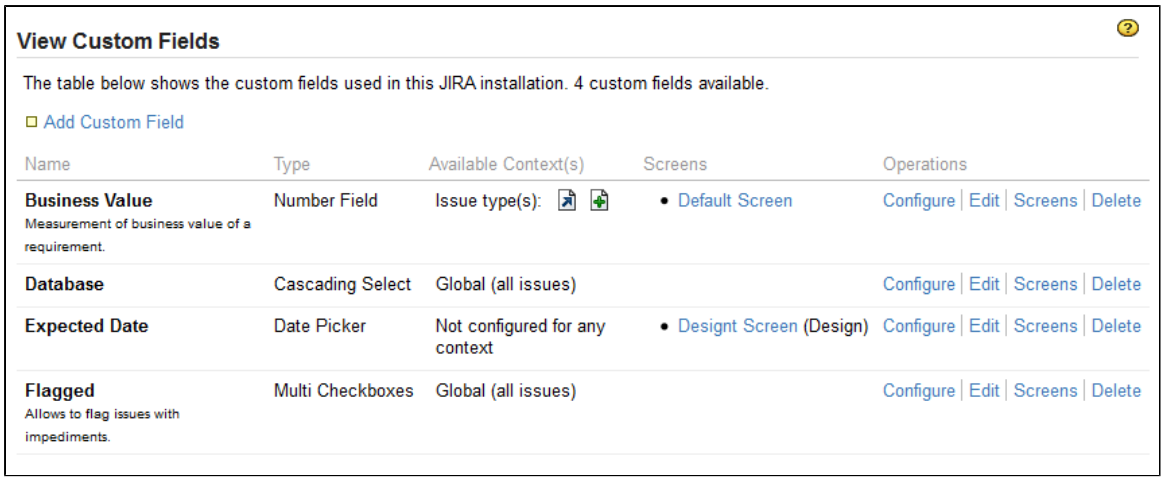
On this page:

- Viewing All Custom Fields
- Editing a Custom Field
  - Name
  - Description
  - Search Template
- Configuring a Custom Field
  - Context
  - Default Value
  - Options
- Choosing Screens

Viewing All Custom Fields

To view all the custom fields in your JIRA system:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
   Screenshot 1: The 'View Custom Fields' screen



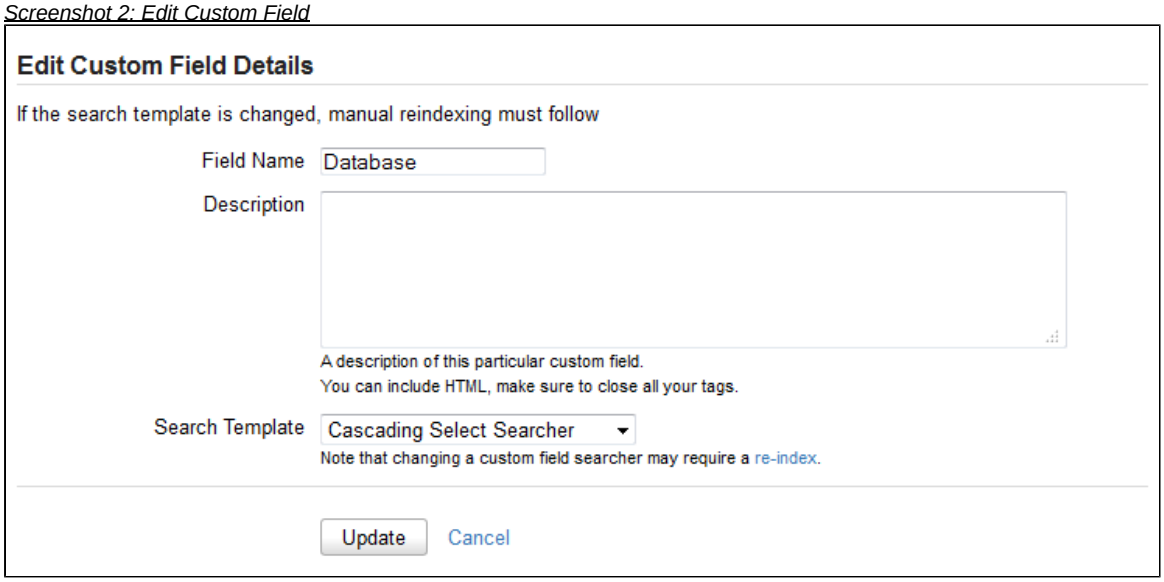
3. From the 'View Custom Fields' page, you can:
    - Edit a custom field's 'Name', 'Description' or 'Search Template' — see below.
    - Configure a custom field's 'Options', 'Default Value' or 'Context' — see below.
    - Place a custom field on a particular screen(s) — see below.

Editing a Custom Field

Editing a custom field allows you to change its Name (label), Description (Help text) and Search Template.

To edit a custom field:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Locate the relevant custom field and click the '**Edit**' link in the right-hand column. You can then edit the custom field's 'Name' (label), 'Description' (Help text) and 'Search Template' (below).
   Screenshot 2: Edit Custom Field



Name

The **Name** is the label that appears to the left of the custom field when it is displayed to a user. You can edit the Name as described above.

Description

The **Description** is the Help text that appears below the custom field when it is displayed in the Simple Search column. You can edit the Description as described above.

Note that the Help text which appears below the custom field when it is displayed on a screen (ie. when an issue is being created, edited or transitioned through a workflow) is specified via the *field configuration* — see Specifying Field Behaviour.

Search Template

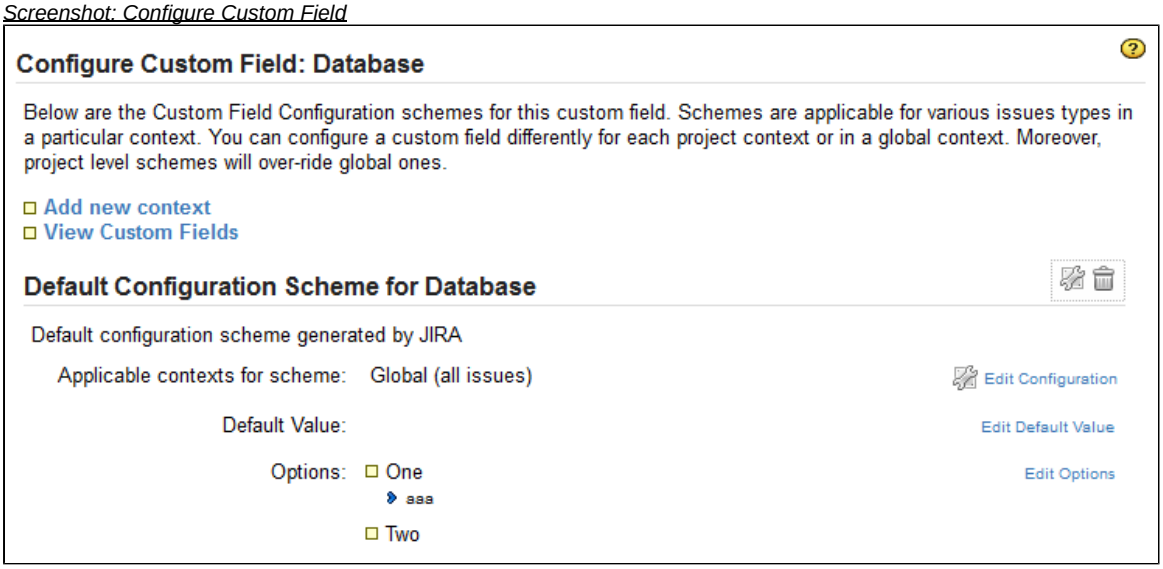
Search Templates are responsible for indexing a custom field as well as making it searchable via Simple Search and Advanced Search (note that custom fields are not searchable via Quick Search). Each of the default custom field types has a related preconfigured search template. You can choose a different Search Template as described above.

Configuring a Custom Field

A *custom field context* (also known as a *custom field configuration scheme*[1]) specifies the Default Value and Options for the custom field, and the issue types and projects to which the Default Value and Options will apply. You can create multiple contexts if you need to associate different Default Values and Options with particular projects or issue types.

Each custom field has a context named "**Default Configuration Scheme for ...**" which was created automatically by JIRA when you initially added your custom field, e.g.:

Screenshot: Configure Custom Field

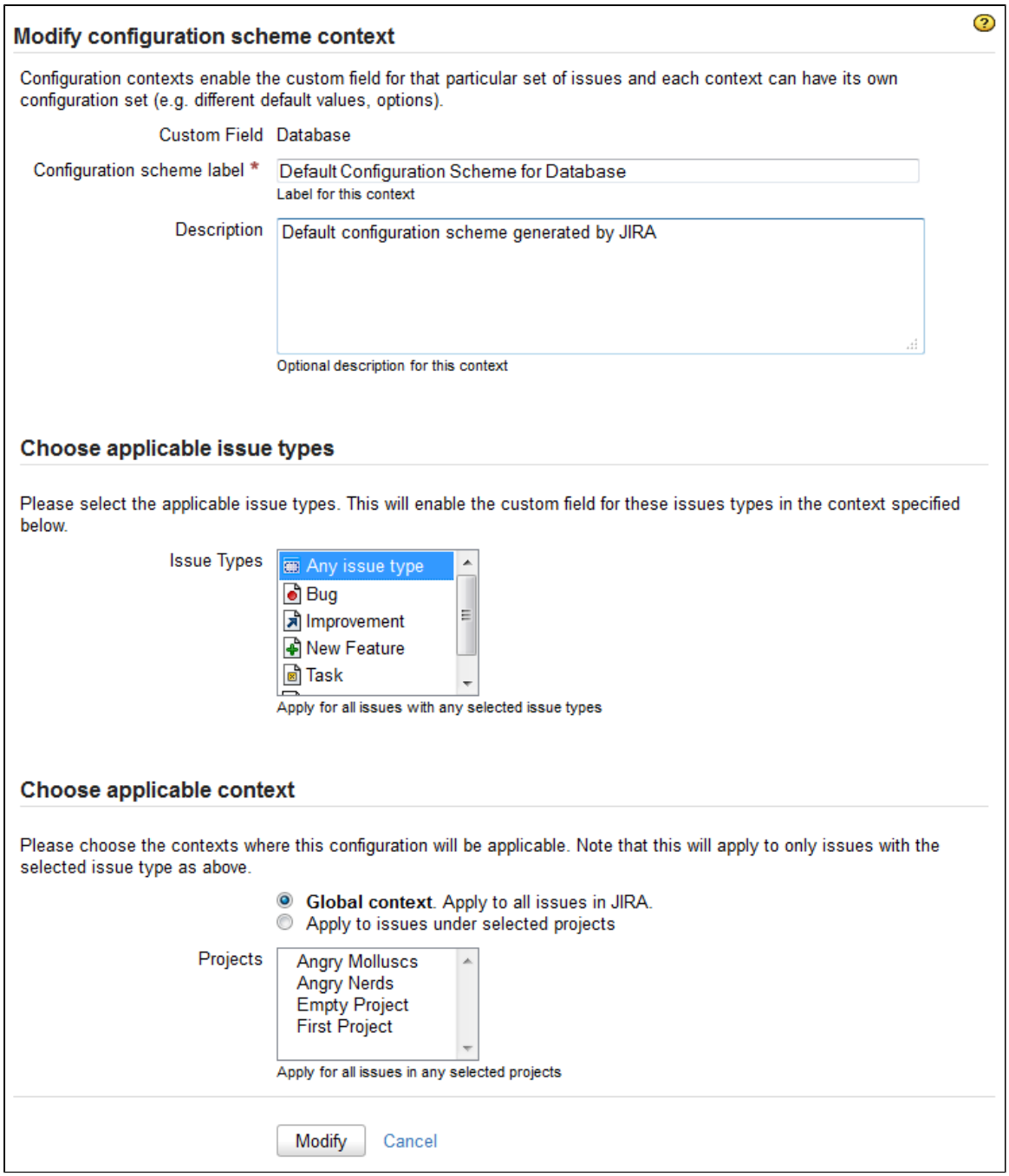


[1] The *custom field configuration scheme* is **not** related to the *field configuration scheme*.

Context

To change the project(s) and issue type(s) for which a given 'Default Value' and 'Options' will apply:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Locate the relevant custom field and click the '**Configure**' link in the right-hand column. The 'Configure Custom Field' page will be displayed (see above).
4. Locate the relevant context (there will usually only be one, named 'Default Configuration Scheme for ...') and click the '**Edit Configuration**' link in the right-hand column. The 'Modify configuration scheme context' screen will be displayed (see below).
   Screenshot 3: Modify configuration scheme context



5. Under 'Choose applicable issue types', select the issue type(s) to which you want this Default Value and Options to apply. You can select '**Any issue types**' if you wish.
6. Under 'Choose applicable contexts', select the project(s) to which you want this Default Value and Options to apply. Note that this will apply to only issues with the selected issue type(s) as above.

Adding a new Context

Adding a new context allows you to configure a custom field differently for different combinations of issue types and projects.

To add a new context:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Locate the relevant custom field and click the '**Configure**' link in the right-hand column. The 'Configure Custom Field' page will be displayed (see above).
4. Click the '**Add new context**' link (near the top of the screen). The 'Add configuration scheme context' page will be displayed (see below).
   - Under 'Add configuration scheme context', enter a 'Label' and 'Description' for your new context — these are used for administrative purposes only and will not be shown to your end-users.
   - Under 'Choose applicable issue types', select the issue type(s) to which you want this Default Value and Options to apply. You can select '**Any issue types**' if you wish.
   - Under 'Choose applicable contexts', select the project(s) to which you want this Default Value and Options to apply. Note that this will apply to only issues with the selected issue type(s) as above.

Default Value

To edit the default value that a custom field contains when it is first displayed to a user:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Locate the relevant custom field and click the '**Configure**' link in the right-hand column. The '**Configure Custom Field**' page will be displayed (see above).
4. Locate the relevant context (there will usually only be one, named 'Default Configuration Scheme for ...') and click the '**Edit Default Value**' link in the right-hand column. The '**Set Custom Field Defaults**' page will be displayed and will be particular to the custom field type:
   - *(For a Select List or Multi-Select List)* Select the appropriate default value from the drop-down list.
     To clear the default of a select field, click on the current default so it is no longer highlighted and then save, as described here: Unable to De-select Default Value for Multi Select Custom Field.
   - *(For a Cascading Select List)* Select the appropriate default values from the drop-down lists (one for each level).
   - *(For a Date field)* Specify a date, or tick the check-box to make the current date the default.
   - *(For other types of fields)* Type the appropriate default values from the drop-down lists (one for each level).
     Certain types of custom fields, such as calculated custom fields, may not allow for defaults to be selected and will not have the '**Edit Default Value**' link.

Options

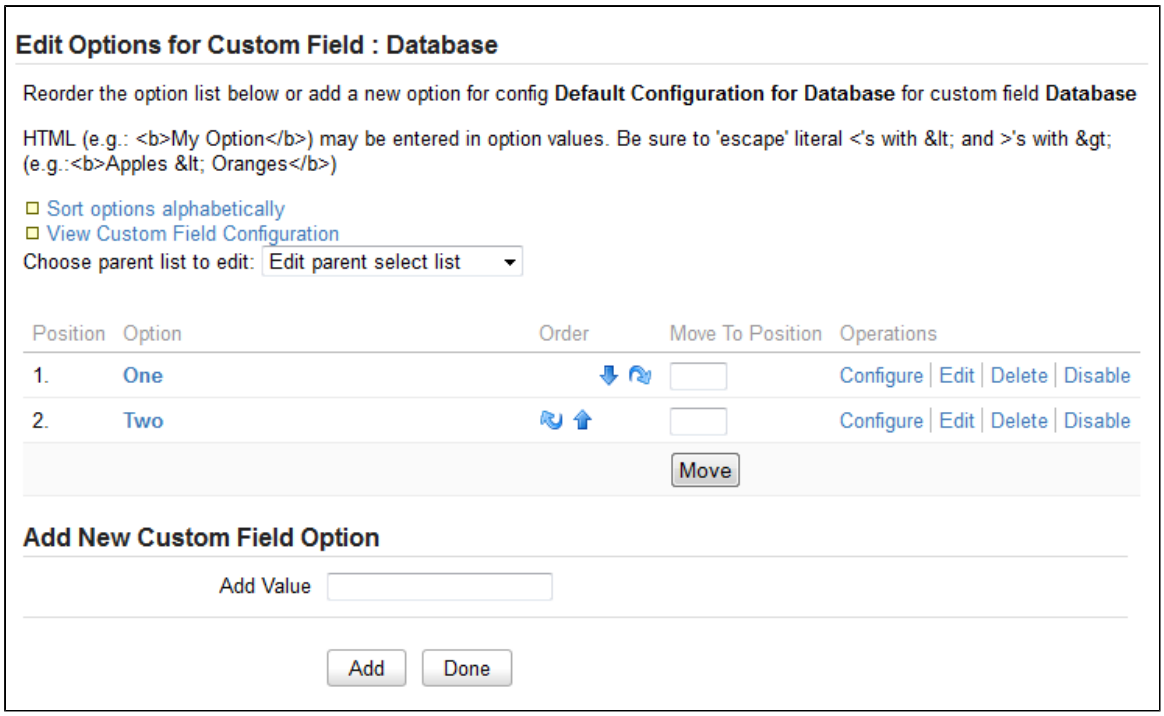
You can specify option values for custom fields of the following types:

- Select lists
- Multi select lists
- Cascading selects lists
- Radio buttons
- Multi checkboxes

You can add, remove, re-order, sort the options alphabetically, and edit the text of an option value. You can also have HTML in an option value — be sure to use complete tag pairs, and check that the HTML will display correctly.

To edit a custom field's options:

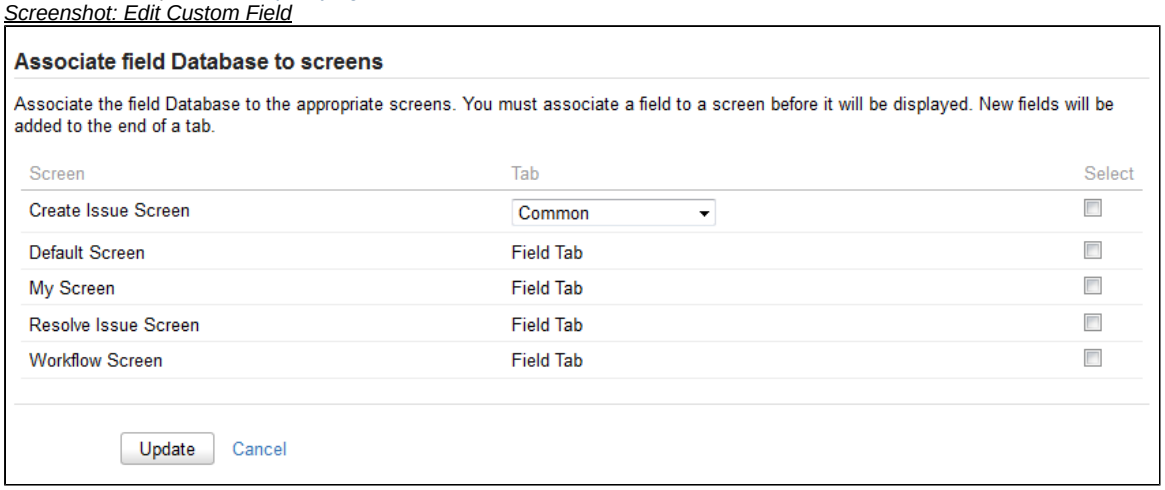
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Locate the relevant custom field and click the '**Configure**' link in the right-hand column. The '**Configure Custom Field**' page will be displayed (see above).
4. Locate the relevant context (there will usually only be one, named 'Default Configuration Scheme for ...'), and click the '**Options**' link in the right-hand column. The 'Edit Custom Field Options' page will be displayed (see below). Here you can:
   - Select from the 'Edit parent select list' drop-down to choose which list to edit. *(For a Cascading Select List only)*
   - Click '**Sort alphabetically**' to automatically re-order the options alphabetically.
   - Click the arrows in the 'Order' column, or specify a number and click the '**Move**' button, to re-order the options manually.
   - Click '**Edit**' to change the text of an option.
   - Click '**Disable**' to hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.
   - Click '**Delete**' to remove an option. (This will only be possible for options that have not been used.)
     Screenshot: Edit Custom Field Options



Choosing Screens

To choose the Screens on which a custom field will appear:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Custom Fields'** (tab) to open the 'View Custom Fields' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'custom fields'**
3. Locate the relevant custom field and click the '**Screens**' link in the right-hand column. The 'Associate field to Screens' page will be displayed (see below).
4. Select the check boxes of the screens on which you wish to display this custom field.
   Note that field visibility depends on the *field configuration* (which is **not** related to the *custom field configuration scheme* described above). Refer to Specifying Field Behaviour for more information.
   Screenshot: Edit Custom Field

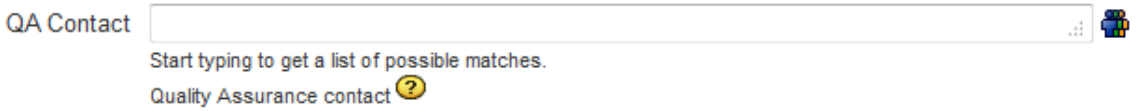


Creating Help for a Custom Field

To provide online help for a custom field, use HTML or Javascript in the field's description. E.g. you can have a simple link to an external help page:

```
<a href="http://www.mycompany.com/jirahelp/fieldhelp.html">get help</a>
```

Or using Javascript, you can have help text right in the field:



where clicking the help icon makes hidden help text appear:

This can be done by entering the following as the field's description:

```
Quality Assurance contact
<script type="text/javascript">
  function showHelp() {
        var listenersDiv = document.getElementById("qaFieldHelp");
        if (listenersDiv.style.display == 'none') {
          listenersDiv.style.display = '';
        } else {
          listenersDiv.style.display='none';
        }
  }
</script>
<a href="#"  onclick="showHelp(); return false;"><img src="/images/icons/ico_help.png"/></a>
<div id="qaFieldHelp" style="display:none">
  The QA Contact is a member of the QA department responsible for taking this issue through
testing.
  They will be notified by email of this and subsequent issue state transitions.
</div>
```

(Incidentally, Javascript in descriptions can also be used to set field values.)

Specifying Field Behaviour

What is a 'Field Configuration'?

A **Field Configuration** is a set of definitions for all fields, including both standard fields and custom fields.

A Field Configuration allows you to specify field behaviour. For each field, a Field Configuration specifies:

- the **description** that appears under the field when an issue is edited
- whether the field is **hidden** or **visible**
- whether the field is **required** (i.e. the field will be validated to ensure it has been given a value) or **optional**
- (for text fields only) which **renderer** to use

You can create multiple Field Configurations (see below). You can then associate different Field Configurations with different issue types by creating a Field Configuration Scheme. Once created, a Field Configuration Scheme can be associated with one or more projects, allowing you to control field behaviour on a per project, per issue type basis.

For more information, please see the Overview Diagram.

On this page:

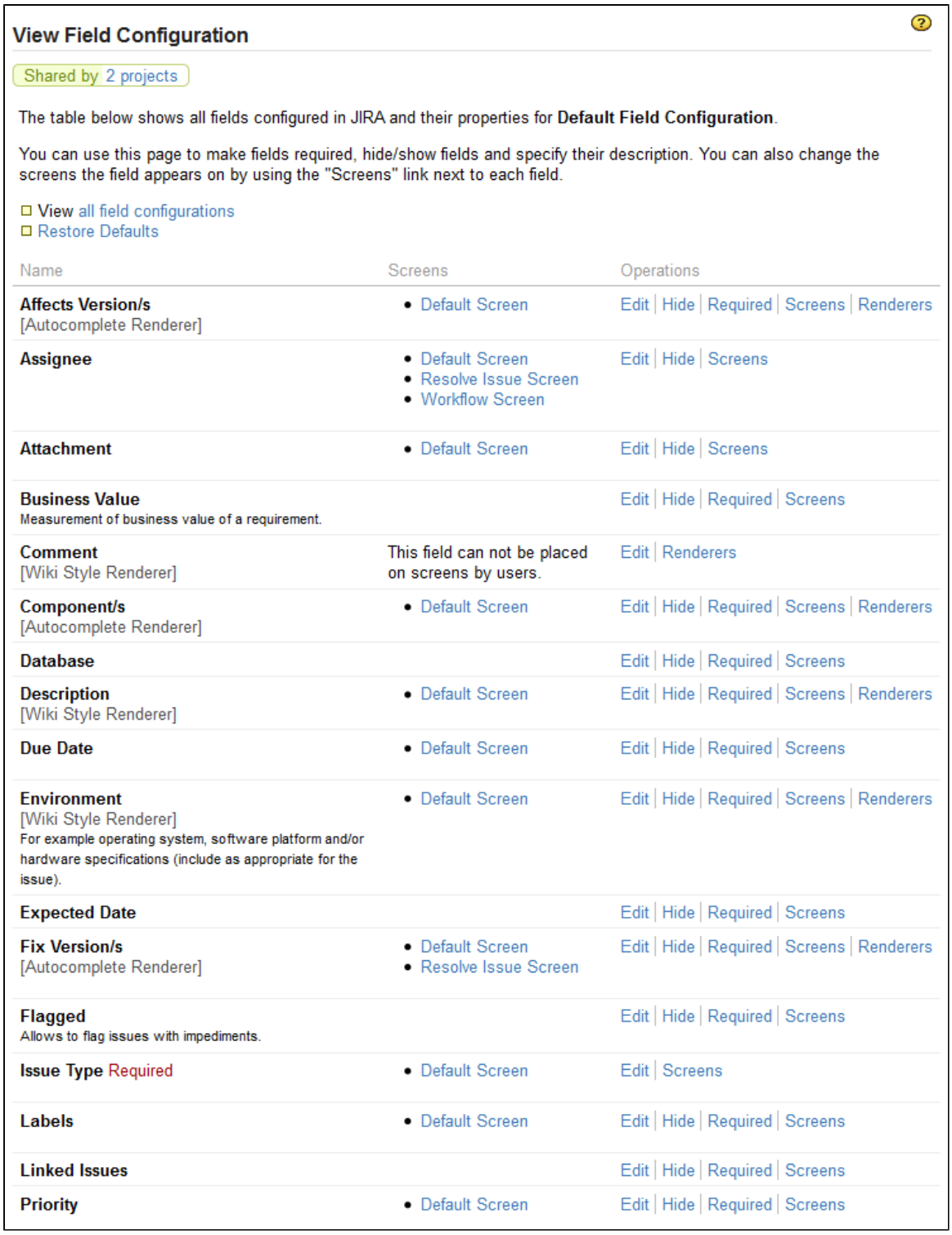
- What is a 'Field Configuration'?
- Editing a Field Configuration
  - Editing a Field's Description
  - Hiding or Showing a Field
  - Making a Field Required or Optional
  - Changing a Field's Renderer
- Managing Multiple Field Configurations
  - About the 'Default Field Configuration'
  - Adding a Field Configuration
  - Editing a Field Configuration
  - Deleting a Field Configuration
  - Copying a Field Configuration
- Activating a Field Configuration

Editing a Field Configuration

To change the behaviour of fields:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Locate the Field Configuration of interest and click the '**Configure**' link to open the 'View Field Configuration' page (below), which lists all system and custom fields in a table.
   The 'Edit' link only allows you to change the Name and Description of the Field Configuration, not of individual fields. Also note that the 'Edit' link is not available for the 'Default Field Configuration'.
4. In the 'Operations' column, you can perform the following actions for any field:
   - Edit — change the field's description (i.e. help text).
   - Hide/Show — hide the field from view or show it.
   - Require/Optional — set a field to be required (so that whenever a field is edited it must be given a value) or optional.
   - Renderers — change a field's renderer (see Configuring Renderers for more information).

   Screenshot 1: The 'View Field Configuration' screen



View Field Configuration

Shared by 2 projects

The table below shows all fields configured in JIRA and their properties for **Default Field Configuration**.

You can use this page to make fields required, hide/show fields and specify their description. You can also change the screens the field appears on by using the "Screens" link next to each field.

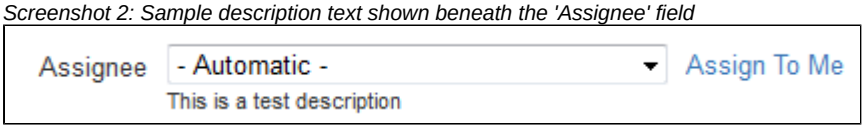
- View all field configurations
- Restore Defaults

Name	Screens	Operations
Affects Version/s [Autocomplete Renderer]	• Default Screen	Edit | Hide | Required | Screens | Renderers
Assignee	• Default Screen • Resolve Issue Screen • Workflow Screen	Edit | Hide | Screens
Attachment	• Default Screen	Edit | Hide | Screens
Business Value Measurement of business value of a requirement.		Edit | Hide | Required | Screens
Comment [Wiki Style Renderer]	This field can not be placed on screens by users.	Edit | Renderers
Component/s [Autocomplete Renderer]	• Default Screen	Edit | Hide | Required | Screens | Renderers
Database		Edit | Hide | Required | Screens
Description [Wiki Style Renderer]	• Default Screen	Edit | Hide | Required | Screens | Renderers
Due Date	• Default Screen	Edit | Hide | Required | Screens
Environment [Wiki Style Renderer] For example operating system, software platform and/or hardware specifications (include as appropriate for the issue).	• Default Screen	Edit | Hide | Required | Screens | Renderers
Expected Date		Edit | Hide | Required | Screens
Fix Version/s [Autocomplete Renderer]	• Default Screen • Resolve Issue Screen	Edit | Hide | Required | Screens | Renderers
Flagged Allows to flag issues with impediments.		Edit | Hide | Required | Screens
Issue Type Required	• Default Screen	Edit | Screens
Labels	• Default Screen	Edit | Hide | Required | Screens
Linked Issues		Edit | Hide | Required | Screens
Priority	• Default Screen	Edit | Hide | Required | Screens

Editing a Field's Description

Fields can be given descriptions to better identify the meaning of the field. These descriptions are usually displayed under the field on the creation of an issue and whenever it is edited, e.g.:

Screenshot 2: Sample description text shown beneath the 'Assignee' field



To edit the description of a field:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**

3. Locate the Field Configuration of interest and click the '**Configure**' link to open the 'View Field Configuration' page ([above|#View_Field_Configuration).
4. Click the '**Edit**' link next to the field you want to change to open the 'Edit Field Description' page.
5. Update the field's description and then click the '**Update**' button.

Hiding or Showing a Field

If your organisation or project has no use for a particular field, you have the option to hide it. Hiding a field will ensure that the the field does not appear on any Screens (i.e. issue operation Screens, workflow transition Screens) where the Field Configuration applies.

Please note:

- Hiding a field in the Field Configuration is distinct from not adding a field to a Screen. Fields hidden through the Field Configuration will be hidden in *all* applicable Screens, regardless of whether or not they have been added to the Screen.
- For fields that have a default value: If the field is hidden in the Field Configuration, then it will not receive a value when an issue is created, regardless of whether the field is present on the 'Create Issue' screen(s). (The following fields can have a default value: 'Resolution'; 'Status'; 'Priority'; 'Issue Type'; custom fields.)
- The fields 'Summary' and 'Issue Type' cannot be hidden and as such there is no 'Hide' option available for these fields.
- If you hide the '**Fix For Version**' field, the Change Log and Road Map reports will not work.

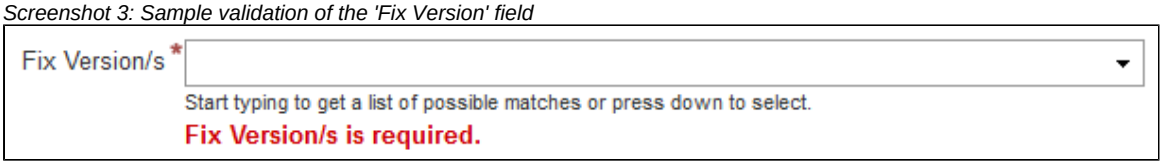
To hide a field:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Locate the Field Configuration of interest and click the '**Configure**' link to open the 'View Field Configuration' page ([above|#View_Field_Configuration).
4. Click the '**Hide**' link next to the field you no longer want. The field will then fade grey to signify that it has been hidden.
   You can make this field visible again at any time by clicking the '**Show**' link.

Making a Field Required or Optional

Certain fields within your organisation may be compulsory for issues. In this case you can set a field to be required, so that JIRA validates that the field has been given a value whenever an issue is edited. If a required field has not been given a value, JIRA will return an error informing the user that the field should be filled, e.g.:

Screenshot 3: Sample validation of the 'Fix Version' field



To make a field required:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Locate the Field Configuration of interest and click the '**Configure**' link to open the 'View Field Configuration' page ([above|#View_Field_Configuration).
4. Click the '**Required**' link next to the appropriate field. The text "Required" will appear next to the field.
5. You can make the field optional again by clicking the '**Optional**' link.

Please note:

- Fields that are hidden cannot be set to required. Making a hidden field required will make it "visible" as well.
  *If you set a field to "required", ensure that the field is present on your 'Create Issue' screen(s). Note that you can have different field configurations for different projects and issue types (see 'Associating field behaviour with Issue Types'); so you need to ensure that all "required" fields are present on the 'Create Issue' screens for all associated projects and issue types (see 'Associating screens with Projects and Issue Types'). There is a feature request at JRA-5783 to make a field required only on one transition. Please watch it for status updates.

Changing a Field's Renderer

JIRA *renderers* affect the display of a field's content, enabling you to choose a style which best suits your organisation and your users.

JIRA currently ships with the following renderers:

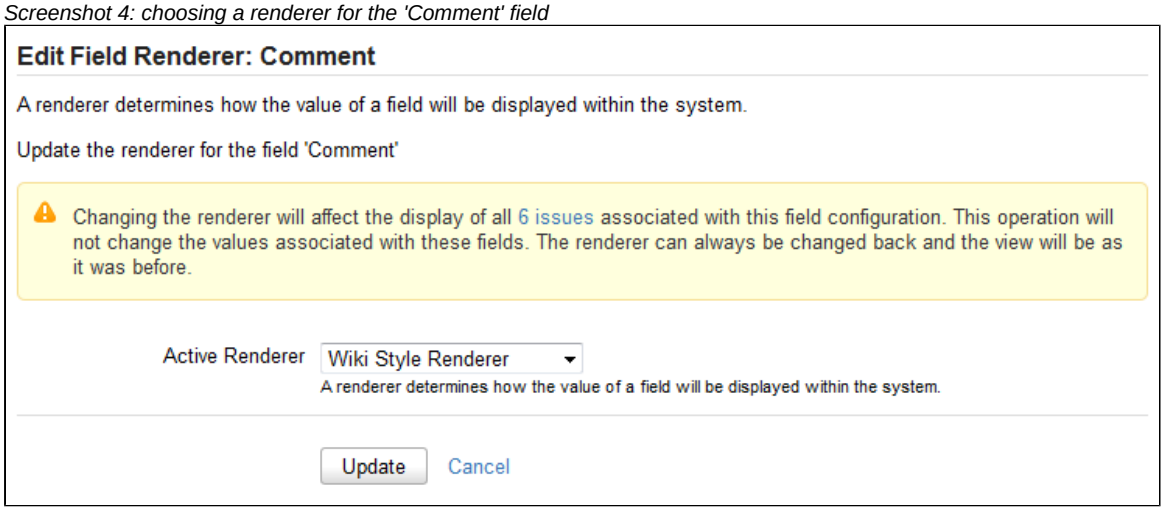
- for text fields:
  - the **Default Text Renderer**, which displays plain text; and
  - the **Wiki Style Renderer** (utilising the Confluence wiki engine), which displays rich text (HTML).
    To see how a 'Wiki Style Renderer' field will look when it is displayed to a user, please see Editing Rich-Text Fields.
- for multi-select fields:

- the **Autocomplete Renderer**, which allows the user to start typing text which is then 'autocompleted', or to select from a drop-down list of options; and
- the **Select List Renderer**, which simply provides a drop-down list of options.
(please note: for custom fields of type "Multi Select", only the Select List Renderer is available)

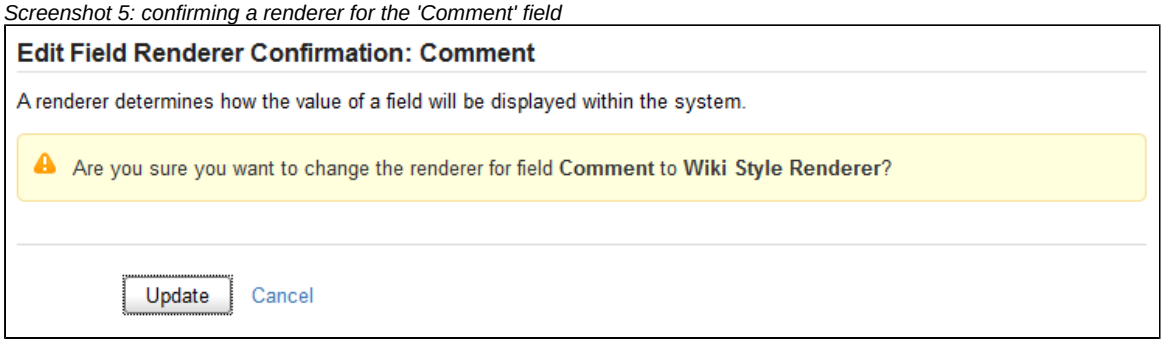
Before you change the renderer for a specific field, please read 'Configuring Renderers', paying particular attention to the section 'Implications for JIRA operations'.

To change the renderer for a specific field:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page.
Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Locate the Field Configuration of interest and click the '**Configure**' link to open the 'View Field Configuration' page ([above|#View_Field_Configuration). The 'Name' column indicates which renderers are currently enabled for all renderable fields, with the current renderer shown in brackets immediately below its field name.
4. Click the '**Renderers**' link for the field you want to change. This will take you to a page where you will have the option to select a renderer from all configured and available renderers:
Screenshot 4: choosing a renderer for the 'Comment' field



5. As shown above, this page will warn you if there are issues that will be affected by the change. If no issues will be affected then the warning does not show. From this page, choose the renderer you wish to use and click '**Update**'. You will be presented with a confirmation page:
Screenshot 5: confirming a renderer for the 'Comment' field



Changing the renderer only affects the *display* of the issue data that exists in the system. You can therefore toggle back and forth between renderer types safely.

Managing Multiple Field Configurations

You can create multiple field configurations for use on separate projects and issue types.

- Multiple field configurations are organised into Field Configuration Schemes, which associate field configurations with issue types.
- A scheme can then be associated with one or more projects, allowing you to control fields on a per project, per issue type basis. See Associating Field Behaviour with Issue Types for details.

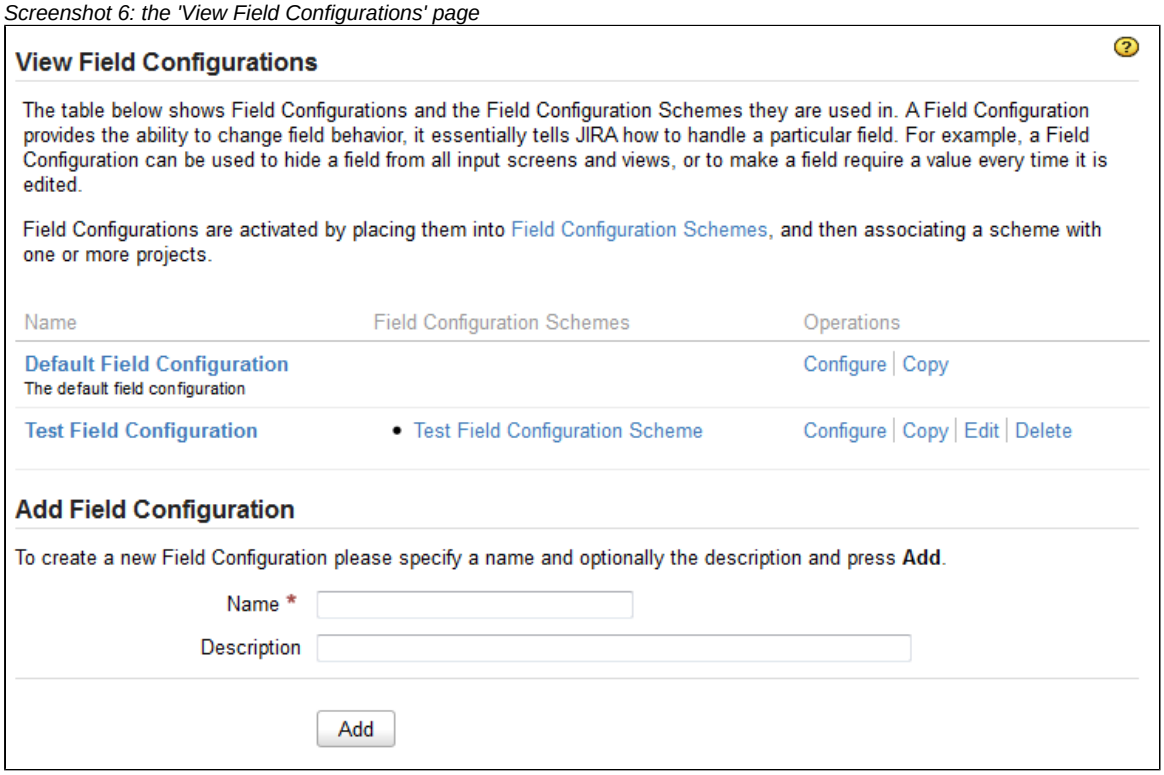
About the 'Default Field Configuration'

When JIRA is installed, the 'Default Field Configuration' is created automatically. All new projects are associated with this configuration. This configuration is also used for projects that are not associated with a Field Configuration Scheme.

Note that it is not possible to delete the Default Field Configuration.

Adding a Field Configuration

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page, which lists all your field configurations:
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
   Screenshot 6: the 'View Field Configurations' page



3. The '**Add**' new field configuration form is located at the bottom of the 'View Field Configurations' page. Enter the name of the new configuration on this form and optionally add a description.

Note that a newly created Field Configuration will not take effect until it has been activated (see below).

Editing a Field Configuration

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Click the '**Edit**' link next to the field configuration you wish to edit. The 'Edit Field Configuration' page opens, which allows you to edit the field configuration's name and description.

Deleting a Field Configuration

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Click the '**Delete**' link next to the field configuration you wish to delete.
   You will be prompted to confirm this operation.

Please note:

- The Default Field Configuration cannot be deleted.
- You can only delete a field configuration that is not associated with a Field Configuration Scheme.

Copying a Field Configuration

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configurations'** (tab) to open the 'View Field Configurations' page (above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'field configurations'**
3. Click the '**Copy**' link next to the field configuration you wish to copy.
4. Enter the name and description of the new field configuration
5. Click the '**Copy**' button. You will now be directed back the 'View Field Configuration' page and your copied field configuration added to the list.
   The field settings on the original and the copied field configurations will be identical.

Note that a newly created Field Configuration will not take effect until it has been activated (see below).

Activating a Field Configuration

To activate a Field Configuration:

1. configure a Field Configuration Scheme to associate the Field Configuration with appropriate issue types;
2. then associate the Field Configuration Scheme with a project.

For details of both procedures, see Associating Field Behaviour with Issue Types.

Associating Field Behaviour with Issue Types

What is a 'Field Configuration Scheme'?

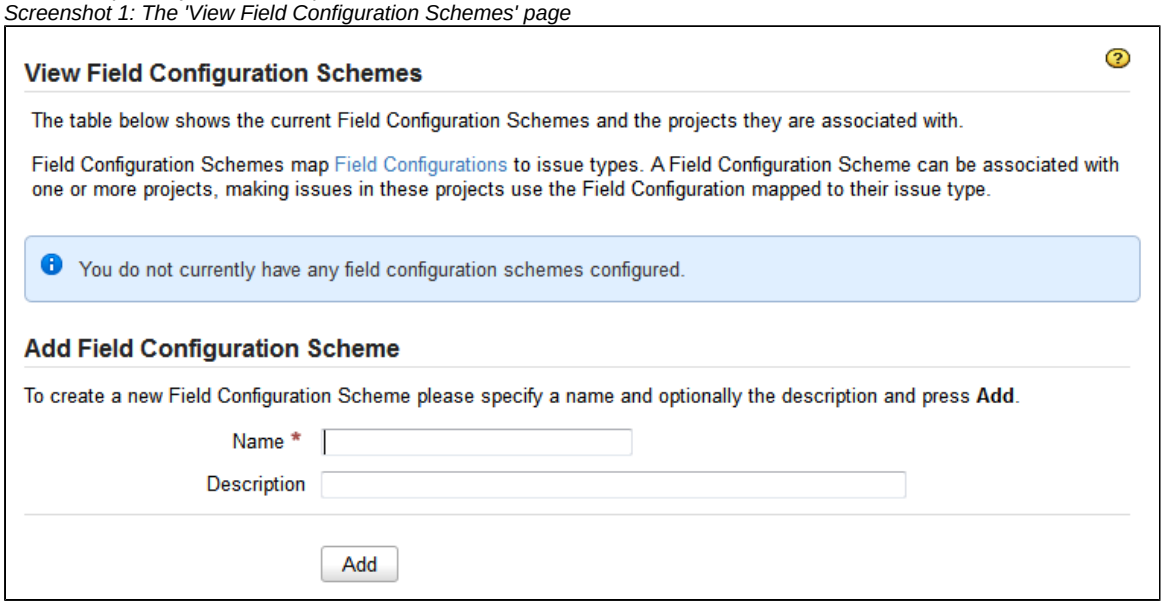
A **Field Configuration Scheme** maps Field Configurations to issue types. A Field Configuration Scheme can be associated with one or mor projects.

This means that you can define Field Configurations for a particular issue type in a given project. For example, it is possible to have a separate field configuration for the 'Bug' issue type and the 'Improvement' issue type for the 'Test' Project. For more information, please see the Overview Diagram.

Because field configuration schemes can be associated with more than one project, your administrative workload is minimised as you can reuse the same field configuration for issue type mappings across multiple projects.

Adding a Field Configuration Scheme

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page, which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. In the 'Add New Field Configuration Scheme' form located at the end of the page, enter the name of the new configuration on this form, and optionally add a description; then click the '**Add**' button.
   Screenshot 1: The 'View Field Configuration Schemes' page



On this page:

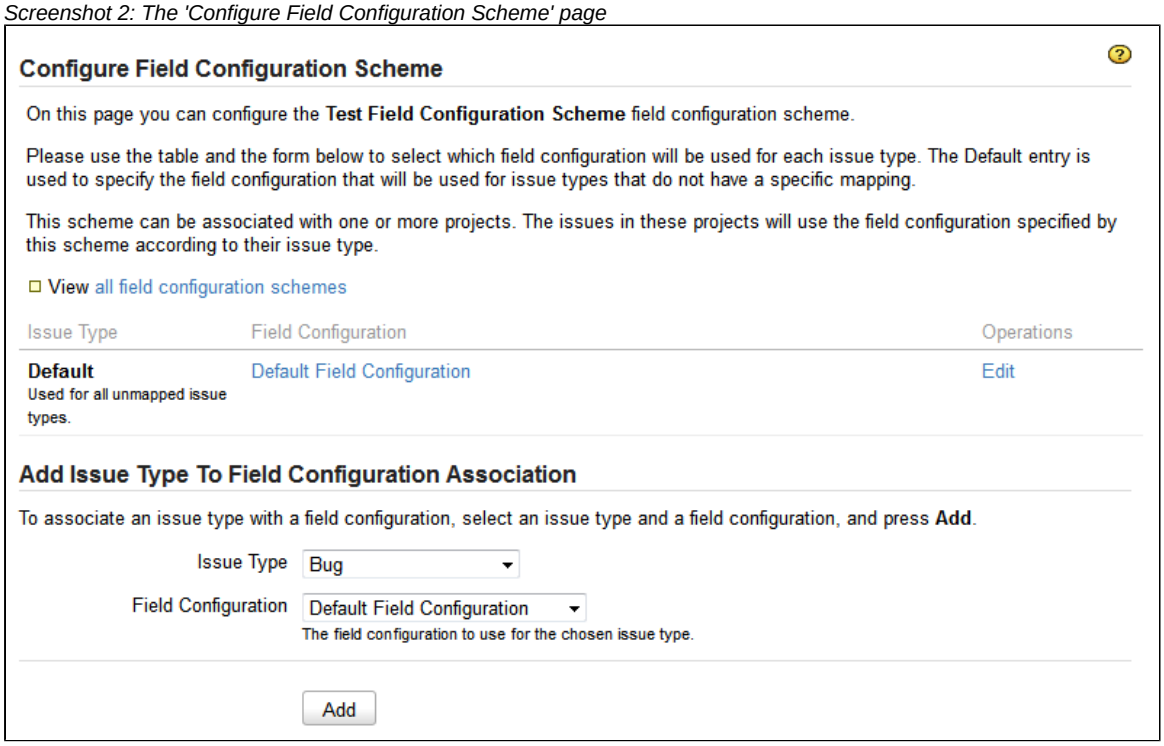
- What is a 'Field Configuration Scheme'?
- Adding a Field Configuration Scheme
  - Associating an Issue Type with a Field Configuration
  - Removing an association an Issue Type and a Field Configuration
  - Associating an Issue Type with a different Field Configuration
- Editing a Field Configuration Scheme
- Deleting a Field Configuration Scheme
- Copying a Field Configuration Scheme
- Associating a Field Configuration Scheme with a Project

Associating an Issue Type with a Field Configuration

1. Log in as a user with the '**JIRA Administrators**' global permission.

2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page (see above), which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Configure**' link for the Field Configuration Scheme that contains a Field Configuration which you wish to associate with an issue type. The 'Configure Field Configuration Scheme' page will appear, showing the scheme's current mappings of Field Configurations to issue types.
   If you have not added any field configurations, you will only have the Default Field Configuration to work with.
4. In the 'Add Issue Type To Field Configuration Association' form located at the end of the page, select the desired issue type and field configuration and click the '**Add**' button.

Screenshot 2: The 'Configure Field Configuration Scheme' page



Please note:

- An issue type can only have one association within a given configuration scheme.
- If an issue type does not have an association in the scheme, the field configuration associated with the *Default* entry in the scheme will be used for issues of that type.

Removing an association an Issue Type and a Field Configuration

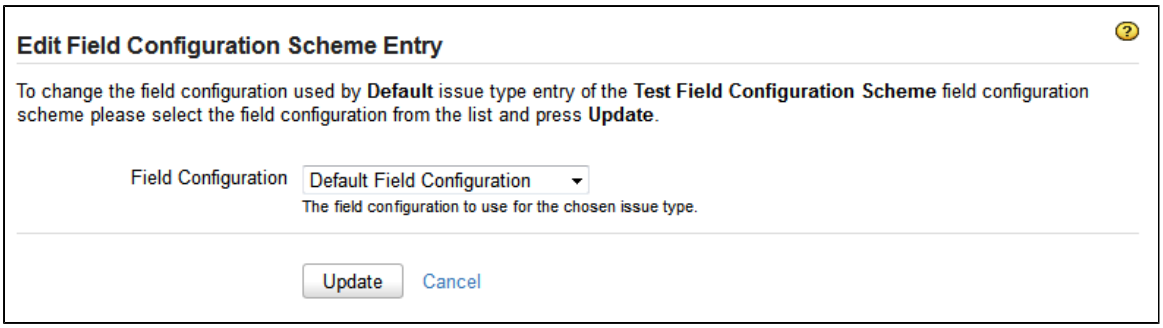
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page (see above), which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Configure**' link for the Field Configuration Scheme that contains a Field Configuration which you wish to no longer associate with an issue type. The 'Configure Field Configuration Scheme' page will appear, showing the scheme's current mappings of Field Configurations to issue types.
   If you have not added any field configurations, you will only have the Default Field Configuration to work with.
4. Click the '**Remove**' link next to the issue type you wish to remove from the scheme.

Please note: The *Default* entry cannot be removed from the scheme.

Associating an Issue Type with a different Field Configuration

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page (see above), which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Configure**' link for the Field Configuration Scheme that contains a Field Configuration which you wish to associate with a different issue type. The 'Configure Field Configuration Scheme' page will appear, showing the scheme's current mappings of Field Configurations to issue types.
   If you have not added any field configurations, you will only have the Default Field Configuration to work with.
4. Click the '**Edit**' link next to the issue type whose field configuration you wish to change.
5. Select the new field configuration you would like to associate with this issue type and click the '**Update**' button.

Screenshot 3: Changing the field configuration with which an issue type is associated



Editing a Field Configuration Scheme

To change the name or description of a Field Configuration Scheme:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page (see above), which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Edit**' link next to the desired field configuration scheme. You will now see the 'Edit Field Configuration Scheme' page.
4. Change the name and/or description as necessary, and click the '**Update**' button.

Deleting a Field Configuration Scheme

To delete a Field Configuration Scheme:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page (see above), which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Delete**' link next to the desired field configuration scheme. You will be prompted to confirm your deletion

Copying a Field Configuration Scheme

To copy a Field Configuration Scheme:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Fields' > 'Field Configuration Schemes'** to open the 'View Field Configuration Schemes' page (see above), which lists all your field configuration schemes (if any exist).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Copy**' link next to the field configuration scheme you wish to copy. This will bring you to the 'Copy Field Configuration Scheme' page.
4. Enter the name and description of the new field configuration scheme.
5. Click the '**Copy**' button. You will now be directed back the View Field Configuration Scheme page with your new scheme added. The new scheme will have the same configuration as the copied scheme.

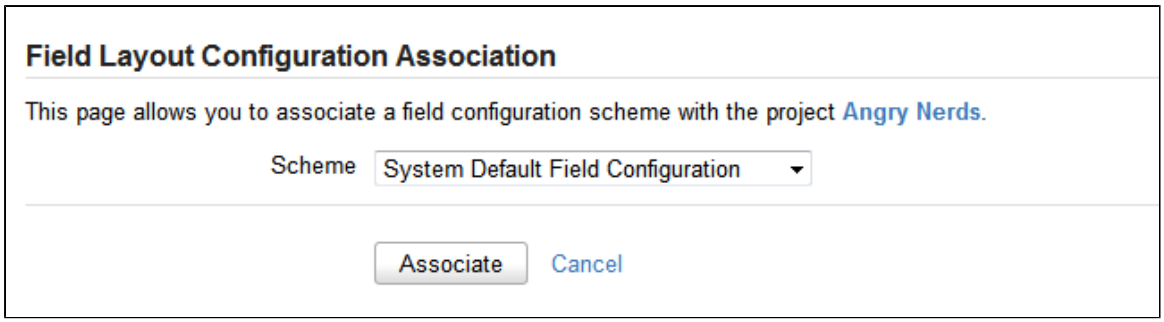
Associating a Field Configuration Scheme with a Project

To activate a Field Configuration Scheme, you need to associate it with a project. An association means that the Field Configuration Scheme will now be applied to the chosen project. The issues in that project will use the Field Configuration that is mapped to their issue type by the scheme. Note that you can use Issue Type Schemes to associate issue types with a project.

Note that newly created projects are not associated with any Field Configuration Schemes, and hence use the Default Field Configuration for all issues.

To associate a Field Configuration Scheme with a project:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click **'Projects'** and select the project of interest, then click **'Administer Project'**. You will now see the 'Project Summary' screen (see Defining a Project).
3. On the lower left, in the 'Fields' section, click the name of the current Field Configuration Scheme.
4. Click the '**Actions**' dropdown menu and choose '**Use a different scheme**'.
5. In the resulting 'Field Layout Configuration Association' page, select the scheme you want to associate with this project.
   Screenshot 4: The 'Field Layout Configuration Association' page



Selecting *None* will make all the issues in the project use the Default Field Configuration.

6. Click the '**Associate**' button. You will be returned to the 'Project Summary' page, with the project now associated with the selected Field Configuration Scheme.

Configuring Renderers

Overview

JIRA *renderers* affect the display of a field's content, enabling you to choose a style which best suits your organisation and your users.

JIRA currently ships with the following renderers:

- for text fields:
  - the **Default Text Renderer**, which displays plain text; and
  - the **Wiki Style Renderer** (utilising the Confluence wiki engine), which displays rich text (HTML).
    To see how a 'Wiki Style Renderer' field will look when it is displayed to a user, please see Editing Rich-Text Fields.
- for multi-select fields:
  - the **Autocomplete Renderer**, which allows the user to start typing text which is then 'autocompleted', or to select from a drop-down list of options; and
  - the **Select List Renderer**, which simply provides a drop-down list of options.
    (please note: for custom fields of type "Multi Select", only the Select List Renderer is available)

Renderers are configured on a per field basis. To configure a renderer for a particular field, see 'Specifying Field Behaviour'. Note that you can configure the same field differently for different projects and issue types — see 'Associating Field Behaviour with Issue Types'.

Renderers are implemented as JIRA plugins, meaning that any renderer can be easily added to or removed from use within JIRA. This includes any custom renderers that may be developed. For details see 'Configuring Renderers' (below).

Please read Implications for JIRA operations below before configuring renderers.

Renderers affect the rendering (view) of a field's value. This means that you can migrate to a different renderer without affecting your issue data; only the view will be changed. It also means that if you do not like the way your issues look using the new renderer, you can simply switch back with no impact on your issue data.

On this page:

- Overview
- Renderable Fields
- Renderer Types
  - Default Text Renderer
  - Wiki Style Renderer
  - Autocomplete Renderer
  - Select List Renderer
- Implications for JIRA operations
  - Bulk Move
  - Bulk Edit
  - Email Notifications
  - Excel View
  - RSS/XML View
  - Editing a Renderable Custom Field's Default Value
- Configuring Renderers
  - Applying a Renderer to a Field
  - Enabling a Renderer Plugin
  - Configuring a Renderer Plugin

Renderable Fields

Potentially any field within JIRA can be a renderable field, but this only really makes sense in the case of text-based fields (for the Default Text Renderer and the Wiki Style Renderer) and multi-select fields (for the Autocomplete Renderer and the Select List Rendered). The following table shows the JIRA fields that are renderable out-of-the-box:

Field	Available Renderers
Description	Wiki Style Renderer (default), Default Text Renderer.
Comment	Wiki Style Renderer (default), Default Text Renderer.
Environment	Wiki Style Renderer (default), Default Text Renderer.
Component	Autocomplete Renderer (default), Select List Renderer.
Affects Version	Autocomplete Renderer (default), Select List Renderer.
Fix Version	Autocomplete Renderer (default), Select List Renderer.
Custom field of type "Free Text Field (unlimited text)"	Wiki Style Renderer, Default Text Renderer.
Custom field of type "Text Field"	Wiki Style Renderer, Default Text Renderer.
Custom field of type "Multi Select"	Select List Renderer.
Custom field of type "Version Picker"	Autocomplete Renderer (default), Select List Renderer.

Renderer Types

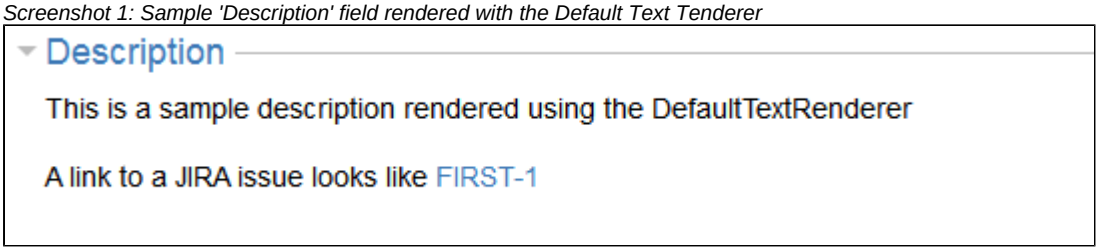
JIRA ships with the following renderers:

- for text fields: Wiki Style Renderer and Default Text Renderer
- for multi-select fields: Autocomplete Renderer and Select List Renderer

Default Text Renderer

The Default Text Renderer renders a field's content as plain text, with the following additional auto-linking feature: if the text contains text that resolves to a JIRA issue key then an HTML link will be generated that points to that issue. Below is a sample of how some description text looks when rendered through the Default Text Renderer.

Screenshot 1: Sample 'Description' field rendered with the Default Text Tenderer



It is not possible to disable the Default Text Renderer plugin as it is required for the system to function properly. If a text field is setup to use a renderer that is later disabled, the field will revert to using the Default Text Renderer.

Wiki Style Renderer

The Wiki Style Renderer allows a user to enter wiki markup to produce html content, as described in 'Editing Rich-Text Fields' in the *JIRA User's Guide.*

This renderer uses the Confluence wiki renderer engine and therefore uses the Confluence wiki notation. The Confluence notation is easy to learn and allows for:

- Italic, bold and underlined text.
- Multiple levels of headings to organise your document.
- Bullets, numbering, tables and quotations.
- Images, screenshots, and emoticons.
- Powerful mini-applications using macros.
  A full notation guide can be found here.

> The Wiki Style Renderer can only be used with JDK 1.4 and up. The renderer will not run on JDK 1.3.

Please note that some fields may require further field behavior configurations to be enabled — see Choosing a Renderer|Specifying Field Behaviour].

Wiki Style Renderer Macro Support

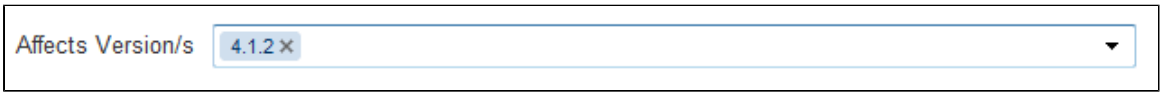
The Wiki Style Renderer supports pluggable macros in the same way that Confluence does. Macros provide an easy and powerful extension point to the wiki markup language. JIRA ships with a number of macros as described in the *JIRA User's Guide.*

JIRA and Confluence can share macros, but keep in mind that many Confluence macros are very specific to the Confluence application and will therefore not run within JIRA. For example, the Children macro in Confluence shows links to all of a Page's child pages. JIRA has no concept of 'page' and therefore this macro will not function in JIRA.

Autocomplete Renderer

The Autocomplete Renderer allows the user to start typing text which is then 'autocompleted', or to select from a drop-down list of options:

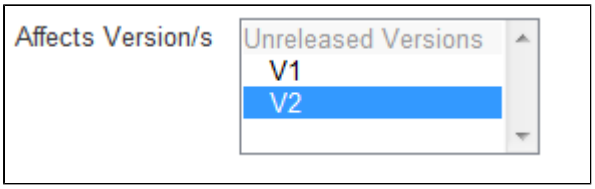
Screenshot 2: Sample field using the Autocomplete Renderer



Select List Renderer

The Select List Renderer provides a drop-down list of options:

Screenshot 3: Sample field using the Select List Renderer



Implications for JIRA operations

The fact that JIRA allows you to configure different renderers across different projects/issue types for the same field has implications for bulk operations. Also, since the Wiki Style Renderer inherently creates HTML as its end product, there are implications as to how this will behave when issue data is viewed outside JIRA's web front-end.

Bulk Move

When performing a bulk move operation you can either move issues to an environment (project/issue type) where the renderer types for the fields are the same or where they will be different.

If the renderer types are the same

If the renderer types for where you are moving to are the same then you will not notice any changes to the way the issues data is displayed once the move has occurred and the move operation will not prompt you with any warnings.

If the renderer types are different

When bulk moving issues to an environment (project/issue type) that has a different renderer type defined for one of the fields being affected by the move, if any of the issues have a non empty value associated with the field, the move operation will you with a warning so that you are aware of the change. The warning does not affect the move operation in any way but it is there to alert you to the fact that the moved issues' affected fields may look different in their new project/issue type.

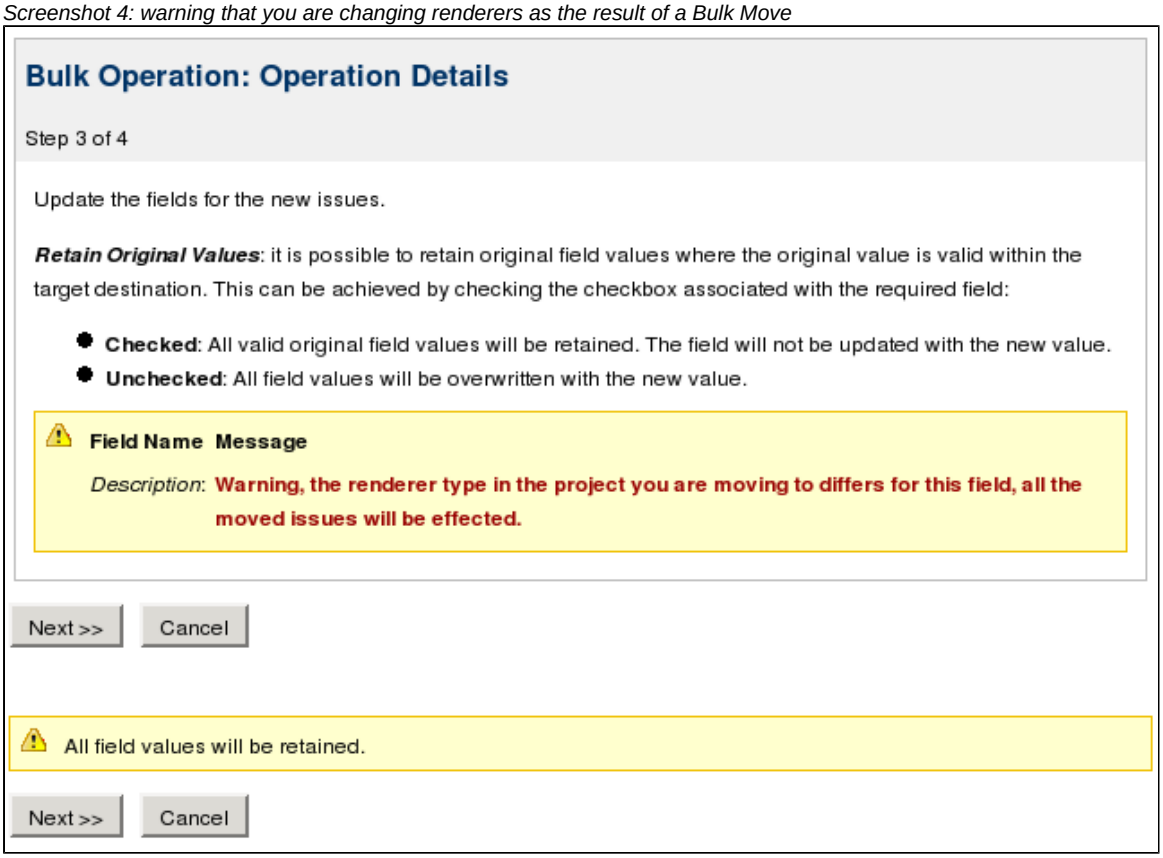
This is best illustrated with an example. Let's say you have project 'A' which is configured to use the Wiki Style Renderer for the Description field. Let's say you also have a project 'B' which is configured to use the Default Text Renderer for the Description field. You have three issues that exist in project 'A' and you want to perform a bulk move of the three issues to project 'B'. If none of the issues in project 'A' have a

value set for the Description field they will be moved and you will not notice any changes since there is no value to render. If one of the issues has the following value in its Description:

```
{color:green}green text{color}

*this is a test issue*
```

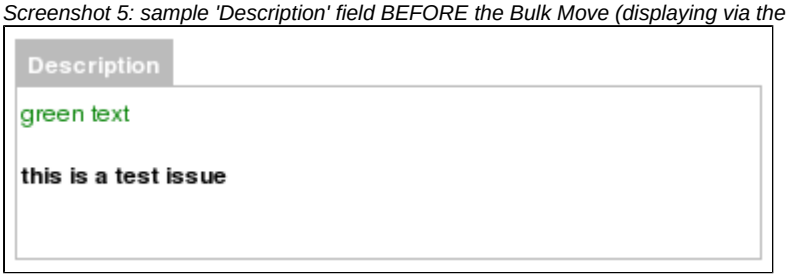
You would be presented with the following screen in the bulk move to alert you that you are changing renderers as a result of the move:

Screenshot 4: warning that you are changing renderers as the result of a Bulk Move



The move operation does nothing to affect the data itself, so after the move the wiki markup will display through the Default Text Renderer. In our example the before and after look like this:

Screenshot 5: sample 'Description' field BEFORE the Bulk Move (displaying via the Wiki Style Renderer)



Screenshot 6: sample 'Description' field AFTER the Bulk Move (displaying via the Default Text Renderer)

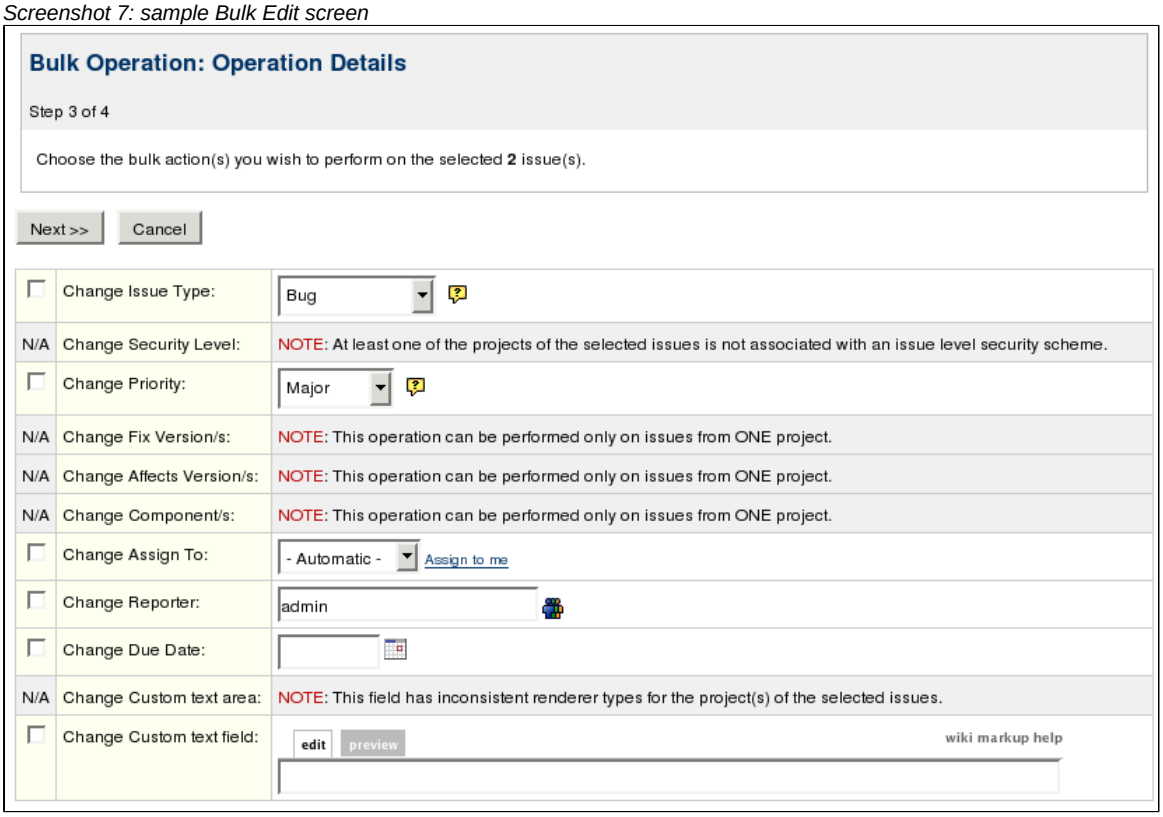
Bulk Edit

When performing a bulk edit operation the only renderable fields you may be able to bulk edit are instances of the Text Field, and Free Text Field (unlimited text) custom fields. The bulk edit operation does not allow you to bulk edit the description, environment, or comment fields.

You will only be allowed to bulk edit a renderable field if all the issues selected for edit use the same renderer type. If the renderer type differs for any of the selected issues you will be presented with an error message.

This is best illustrated with an example. Let's say you have two global custom fields, 'Custom text area' and 'Custom text field', whose types are as their names imply. Let's say you have project 'A' which is configured to use the Wiki Style Renderer for both of the fields. Let's say you also have a project 'B' which is configured to use the Default Text Renderer for the 'Custom text area' field and the Wiki Style Renderer for the 'Custom text field'. Let's also say that you have one issue in each project. If you were to perform a bulk edit operation on the two issues in these projects you will be presented with the screenshot below:

Screenshot 7: sample Bulk Edit screen



You will notice that for the 'Custom text area' field you are presented with a warning that the field has inconsistent renderer types and that it is not available to be selected for bulk edit. This is because the fields do not share the same renderer in the two issues. You will also notice that for the 'Custom text field' field you are presented with an editable input that allows for wiki preview. This is because the field shares the same renderer in the two issues.

Email Notifications

JIRA allows for extensive configuration in relation to email notifications. JIRA can send out two types of emails, HTML and text (see 'Email Formatting').

HTML Emails

When using the Atlassian Wiki Renderer, the rendered content (i.e. exactly what you see on the 'View Issue' page) will be sent out in the emails. This will create emails which are as rich as the content makes it. If using the Wiki Style Renderer, this is the preferred type of email since it is a real representation of the wiki markup.

Text Emails

When using the Atlassian Wiki Renderer, the actual wiki markup (unrendered) will be displayed in text emails for fields that use the Wiki Style Renderer. This is obviously less readable than the rendered version of the markup, but because the markup's syntax is quite simple the text does remain easy to read.

Excel View

JIRA allows the Issue Navigator view to be exported to an Excel spreadsheet. If any of the fields being exported to Excel are using the Wiki Style Renderer, the value exported to the cell in Excel will be the original wiki markup. Attempting to display complex HTML within a cell in Excel adds rows and columns that make using the data for formulas very difficult.

The unrendered wiki markup will be shown in Excel cells for fields that use the Wiki Style Renderer.

RSS/XML View

JIRA allows the Issue Navigator view to be exported to RSS/XML. If a field is using the Default Text Renderer its values will be exported in a CDATA section within the generated XML. If a field is using the Wiki Style Renderer, its rendered value will be XML escaped and included in the generated XML. If the XML view is being used as an RSS feed, most RSS readers will render the generated HTML so you will see the rich content within your RSS reader.

If you would like to have this view feed out the raw values (unrendered) then you can send an additional request parameter 'rssMode=raw'. If the original link looks like this:

```
http://localhost:8080/browse/AAA-1?decorator=none&view=rss
```

Then the URL to have the raw values placed inside a CDATA should look like this:

```
http://localhost:8080/browse/AAA-1?decorator=none&view=rss&rssMode=raw
```

Editing a Renderable Custom Field's Default Value

When editing a renderable custom field's default value, even if it is only ever configured to use the Wiki Style Renderer you will not be presented with the 'Edit' and 'Preview' tabs. Unfortunately it is not possible, in that context, to tell which renderer should be used for editing. This said, if you enter a default value using wiki markup then this will render correctly in environments (project/issue type) where the field has been configured to use the Wiki Style Renderer.

Configuring Renderers

Applying a Renderer to a Field

To enable a renderer for a particular field, edit the Field Configuration and choose the appropriate renderer for the field. For details, see Specifying Field Behaviour.

Enabling a Renderer Plugin

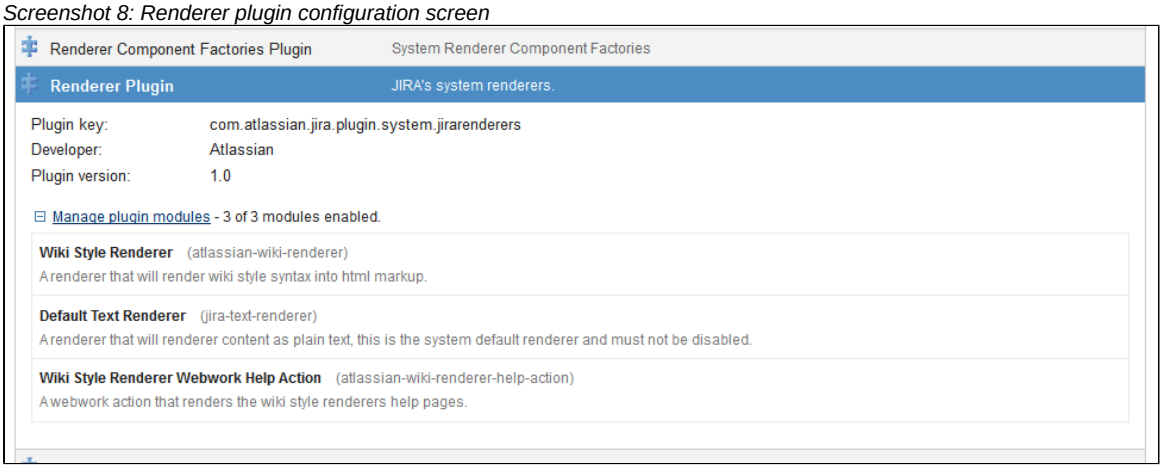
Renderers within JIRA are implemented as JIRA plugins. The macros that the Wiki Style Renderer uses are also implemented as JIRA plugins. For general information on plugins please see the JIRA Plugin Guide.

Note that plugins are configured at a site-wide level — it is not possible to configure plugins at a project/issue type level.

Configuring a Renderer Plugin

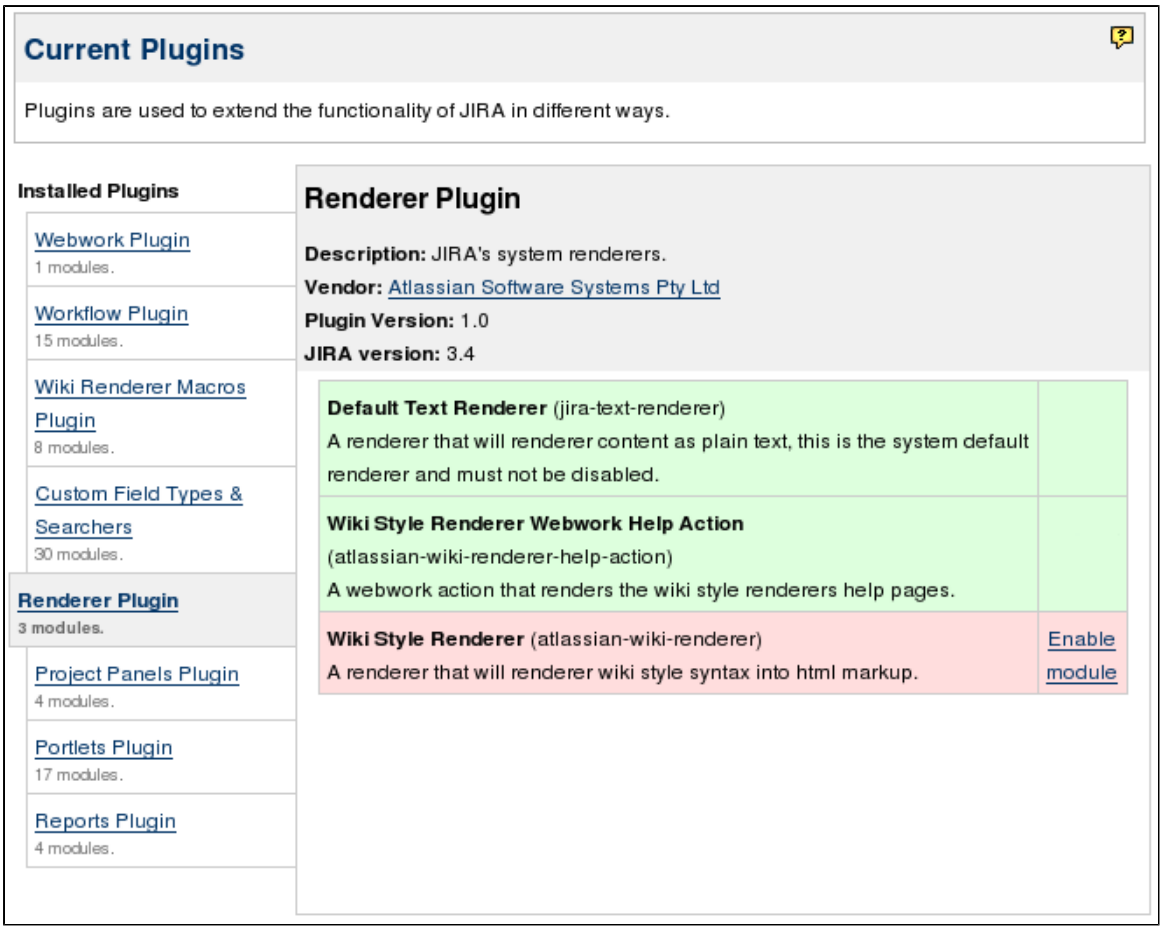
Renderers and their dependant components, except for the Default Text Renderer, can be enabled/disabled via the plugin administration menus. Go to 'Administration' > 'Plugins' and then click on the option 'Renderer Plugin' to display the following screen.

Screenshot 8: Renderer plugin configuration screen



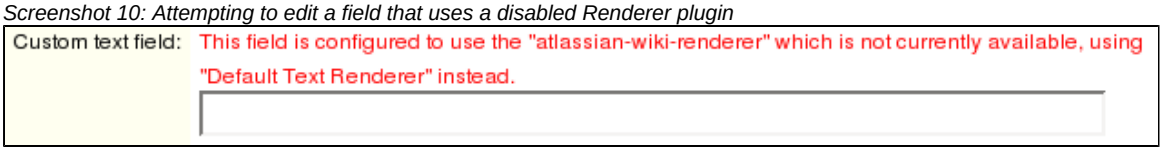
> The plugin titled 'Wiki Style Renderer Webwork Help Action' is a front-end helper for showing the Atlassian wiki renderer notation guide and it cannot be disabled.

From this screen you will see all the configured Renderers within JIRA. At the moment only two renderers exist but if more are created you will see their configuration here. If you click on the 'Disable Module' link for the 'Wiki Style Renderer' this will deactivate the renderer for the entire instance of JIRA.
Screenshot 9: Disabling a Renderer plugin



Any fields that are still setup to use the disabled renderer will fall back to the default text renderer and when you attempt to edit the field a warning message will alert you to the fact that you are configured to use a renderer that is not available.

Screenshot 10: Attempting to edit a field that uses a disabled Renderer plugin

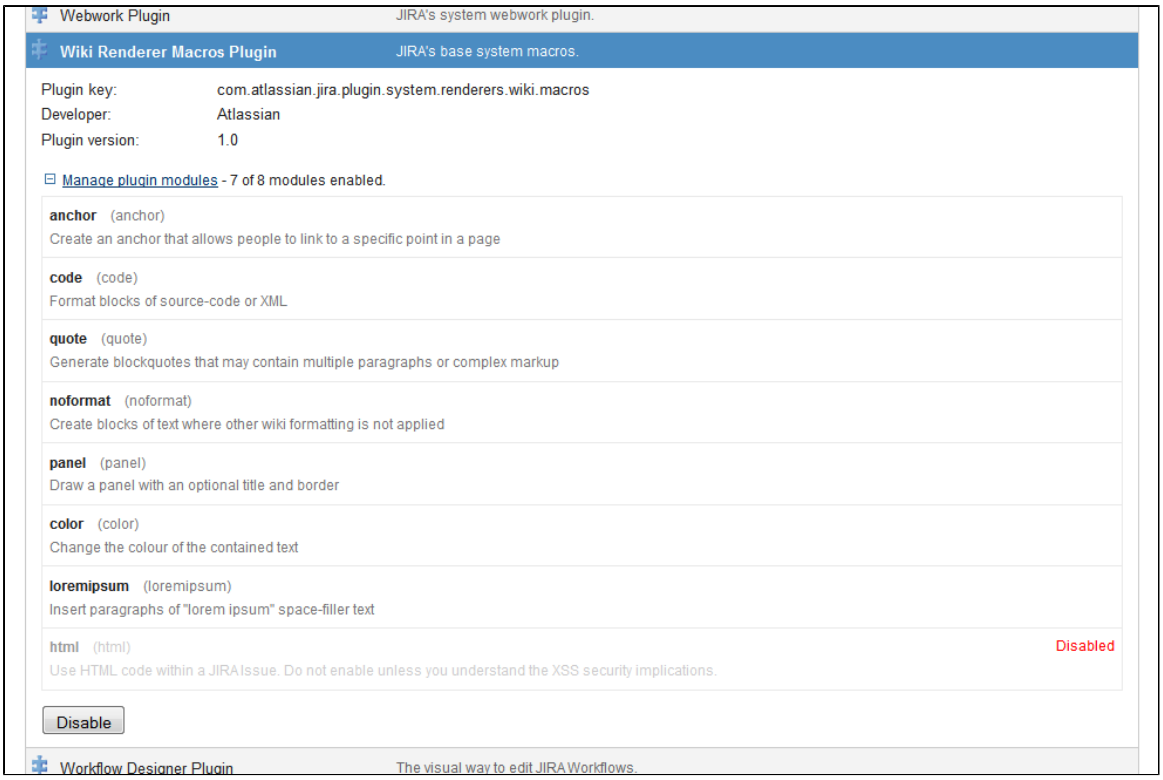


When a renderer is disabled it will not be available for selection when changing a field's renderer. To enable the renderer just click the 'Enable Module' link. Enabling/Disabling a renderer has no effect on the renderer settings in the field configurations so it is possible to disable and then re-enable a renderer without affecting any data.

Configuring Macro Plugins for the Wiki Style Renderer

The macros used by the Wiki Style Renderer can be enabled/disabled via the plugin administration menus. Go to 'Administration' > 'Plugins' and then click on the option 'Wiki Renderer Macros Plugin' to display the following screen.

Screenshot 11: Configuring Macro Plugins for the Wiki Style Renderer



From this screen you will see all the configured macros within JIRA. If a macro is disabled then it will not be available to the wiki renderer. If you deploy any additional macros that you wish to use, they must be enabled here to be available to the wiki renderer. For more information on writing plugins please see the documentation on Writing Macros.

Defining a Screen

What is a 'Screen'?

Screens group multiple issue fields. Using Screens, you can control which fields are displayed, and the fields' vertical display order, during issue operations (e.g. 'Create Issue' and 'Edit Issue') or workflow transitions (e.g. 'Resolve Issue'). You can also split fields on a Screen into multiple tabs.

Screens overlap slightly with Field Configurations in regards to field visibility. Note that when a Screen is displayed to a user (during issue creation, for example) the user will see only the issue fields that:

1. the user has permissions to edit (e.g. the 'Due Date' field can only be edited by users with the 'Schedule Issues' permission).
2. are present on the Screen that is associated with the 'Create Issue' operation for this issue.
3. are not hidden in the Field Configuration applicable to the issue.

A field may be present on a Screen, but if it is hidden in an appropriate Field Configuration, it will not be visible to the user when the Screen is displayed. Note also that, if a particular field needs to be hidden at all times, it is simpler to hide the field in an applicable Field Configuration rather than remove it from all Screens. For more information please see the Overview.

On this page:

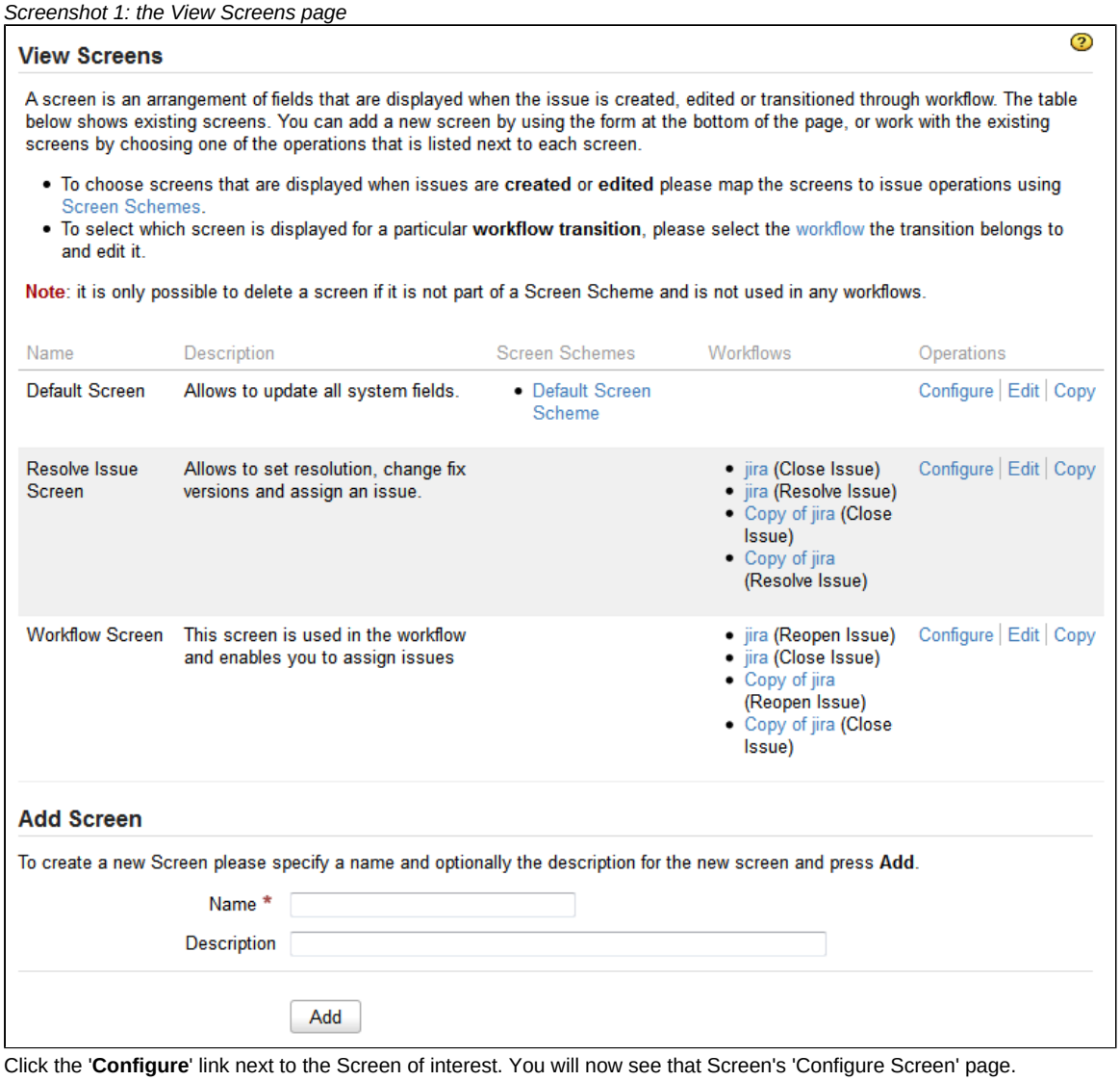
- What is a 'Screen'?
- Configuring a Screen's Fields
  - Adding a Field to Screen
  - Removing a Field from a Screen
  - Reordering Fields on a Screen
  - Adding Time Tracking capabilities to a screen
- Adding a Screen
- Editing a Screen's Details
- Copying a Screen
- Deleting a Screen
- Configuring Tabs
  - Adding a Tab
  - Moving fields between Tabs
  - Deleting a Tab
  - Renaming a Tab
  - Reordering Tabs
- Activating a Screen

Configuring a Screen's Fields

Adding a Field to Screen

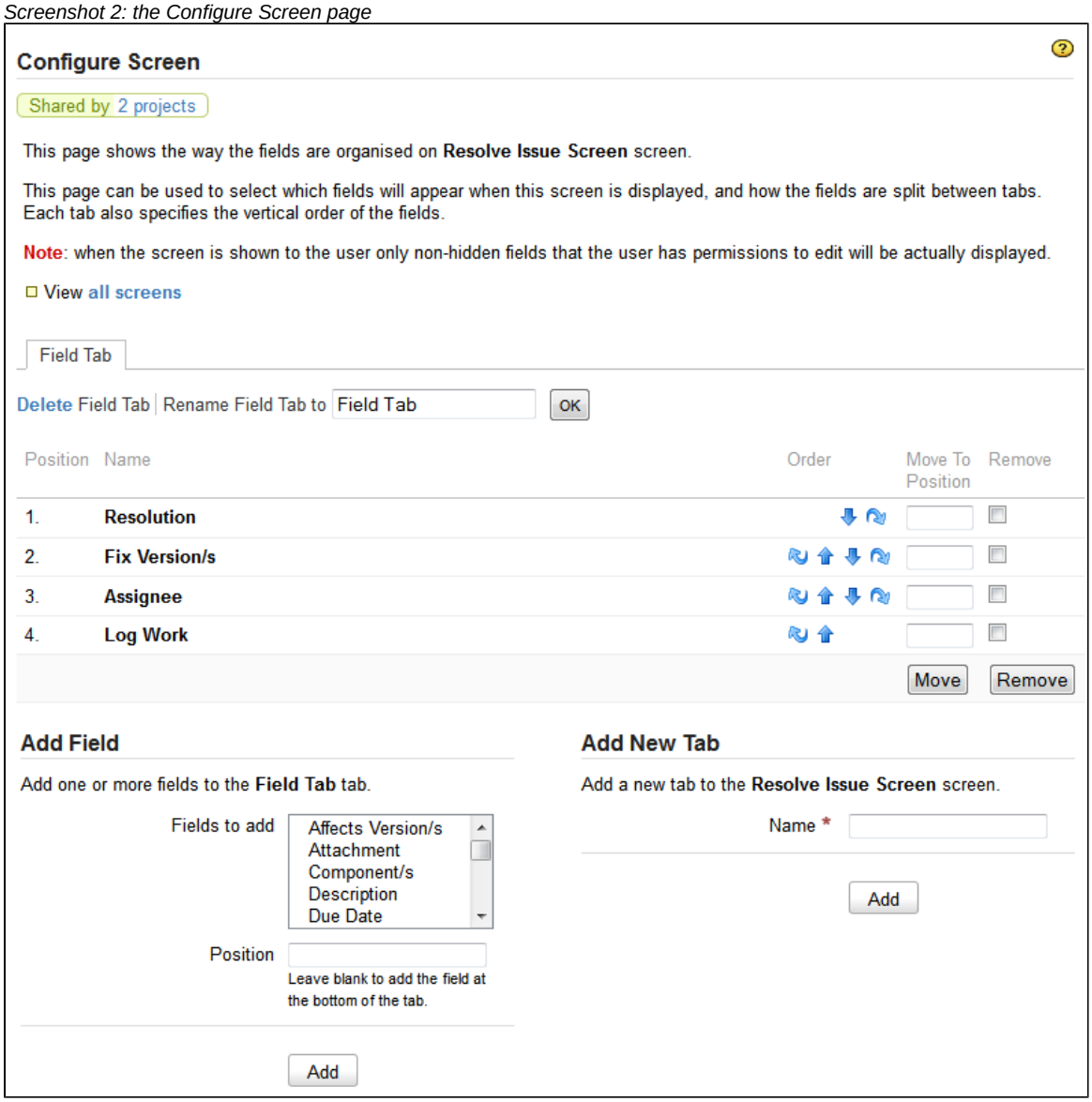
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
   Keyboard shortcut: **'g' + 'g' +** type **'sc'**
3. The 'View Screens' page will be displayed:

Screenshot 1: the View Screens page



4. Click the '**Configure**' link next to the Screen of interest. You will now see that Screen's 'Configure Screen' page.

Screenshot 2: the Configure Screen page



5. In the **Add Field** form, located at the bottom of the **Configure Screen** page:
6. Select the field/s that you wish to add to the screen from the '**Fields to add**' option.
7. If you wish, specify the position at which the field will be placed, by entering the position number in the '**Position**' option.
   If you have selected multiple fields and specified a position, the topmost field selected will be placed in the corresponding position and the other fields directly below it.
8. Click the '**Add**' button.

Removing a Field from a Screen

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. In the table of fields on the 'Defining a Screen page (see above), select the check-boxes next to the fields you wish to remove.
5. Click the '**Remove**' button located at the bottom of the table.
6. The fields will be removed from the Screen and will be made available in the **Add Field** form at the bottom of the screen, for subsequent re-addition if needed.

The '**Summary**' field is always required by the system. If your screen is being used for a 'create issue' operation, you will need to ensure that the '**Summary**' field is on the screen, or your users will be unable to create issues.

Reordering Fields on a Screen

To change the vertical display order of fields on a screen:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. In the table of fields on the 'Defining a Screen page (see above), in the table of fields on the **Configure Screen** page, specify the

number of the position to which a field should be moved inside the field's text box within the **Move to Position** column. (You can repeat this step for multiple fields by specifying a different position for each field you wish to move.)
5. Click the '**Move**' button located at the bottom of the table in the 'Move to Position' column.
   Alternatively, you can click on the arrows next to the desired field to move the field up, down, to the first position or to the last position.

Adding Time Tracking capabilities to a screen

You can add the ability to log work or specify/modify time estimates to a screen by adding the special **Log Work** or **Time Tracking** fields, respectively.

By adding both of these fields to a screen, your users will be able to log work *and* specify time estimates on that screen. For more information about how this works for a user, please refer to Logging work *and/or* specifying time estimates on the same JIRA screen.

To add abilities to log work and/or specify/modify time estimates on a screen, in the **Add Field** form located at the end of the Configure Screen page (see 'Adding a Field to a Screen above):

1. Select one or both of the following fields from the '**Fields to add**' option, depending on your requirements:
   - **Log Work** — adds a group of fields which provide the ability to log work
   - **Time Tracking** — adds a group of fields which provide the ability to specify/modify time estimates
2. Click the '**Add**' button.

If these fields cannot be seen in the **Fields to add** selection box and they have not already been added to the screen, then confirm that JIRA's Time Tracking feature has been activated. These fields will not be available to add to any screen if Time Tracking is deactivated.

If any screens have the **Log Work** or **Time Tracking** fields and JIRA's Time Tracking feature is subsequently deactivated, those screens will retain these fields until you specifically remove them. However, the fields will not be visible to the user until Time Tracking is reactivated.

Adding a Screen

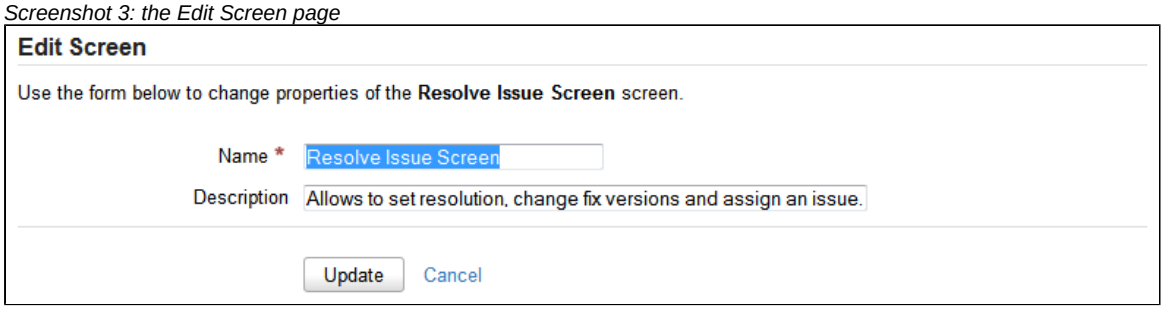
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). In the **Add Screen** form, located at the bottom of the **View Screens** page, enter the name of the new Screen (and optionally add description), then click the '**Add**' button.

A newly created Screen is not usable until it has been associated with either an issue operation (via a Screen Scheme) or a workflow transition. See 'Activating Screens' (below).

Editing a Screen's Details

To change a Screen's name and/or description,

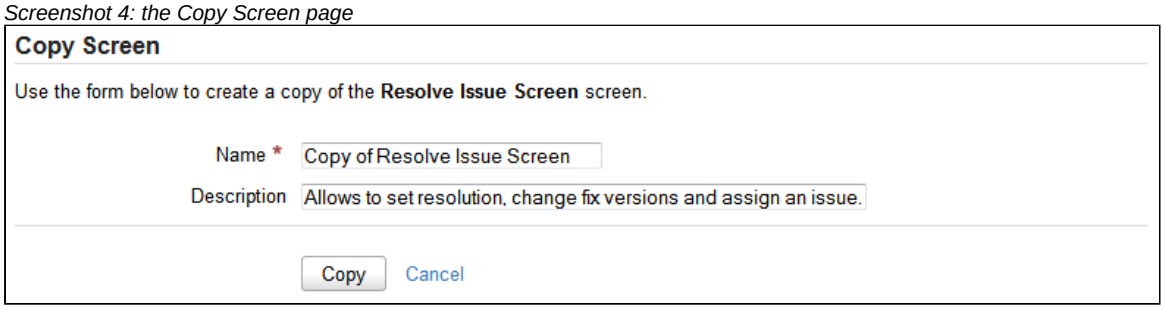
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Edit**' link next to the appropriate screen.
4. You will now be directed to the **Edit Screen** page where you can edit the name and/or description of the Screen.
   Screenshot 3: the Edit Screen page



Copying a Screen

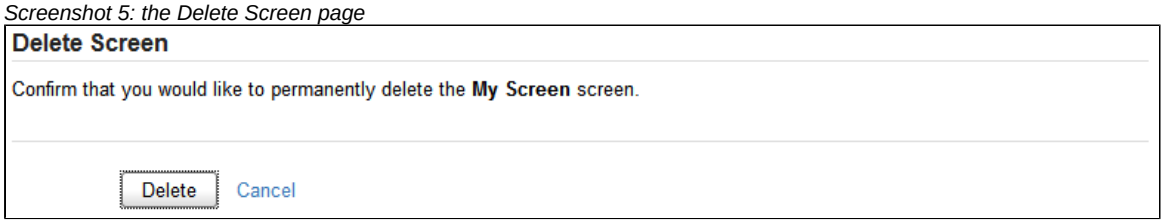
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Copy**' link next to the Screen you wish to copy. You will be directed to the **Copy Screen** page, where you can enter a name and a description for the new Screen:

Screenshot 4: the Copy Screen page



Deleting a Screen

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Delete**' link next to the Screen you wish to delete. You will be promipted to confirm your deletion
   Screenshot 5: the Delete Screen page

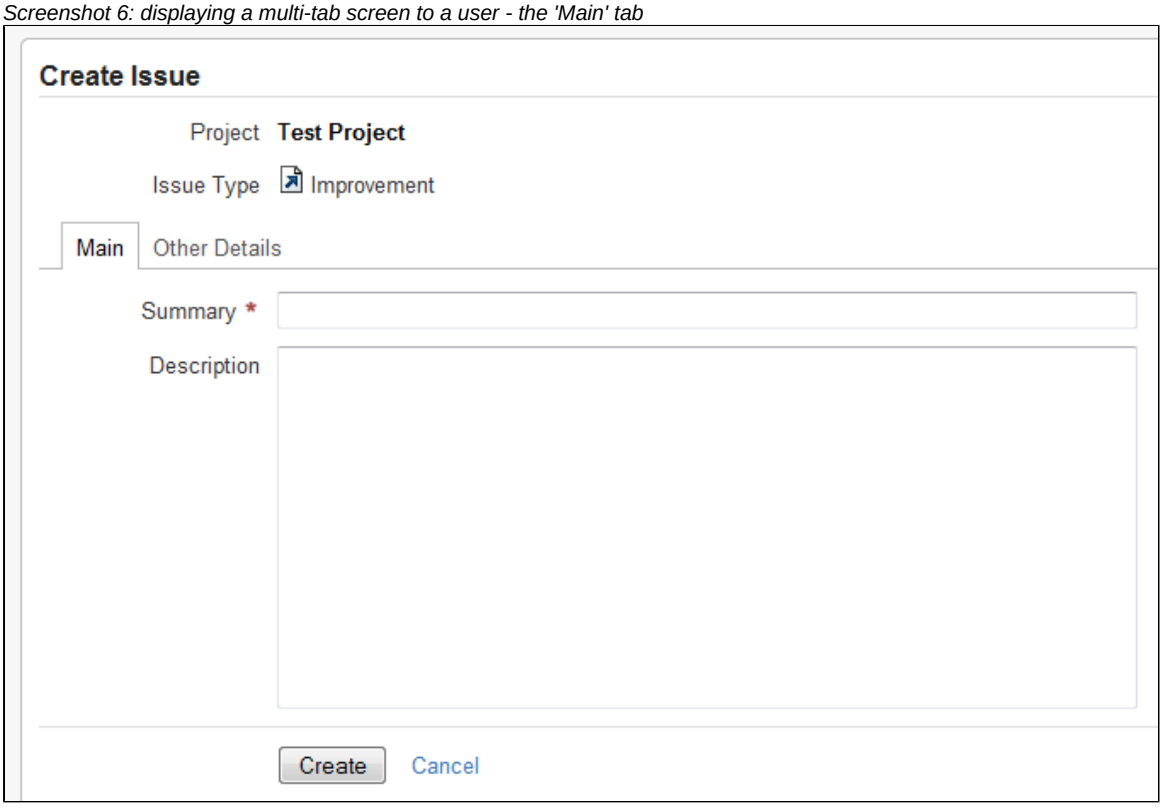


Screens that are associated with one or more Screen Schemes, or one or more workflow transitions, cannot be deleted.

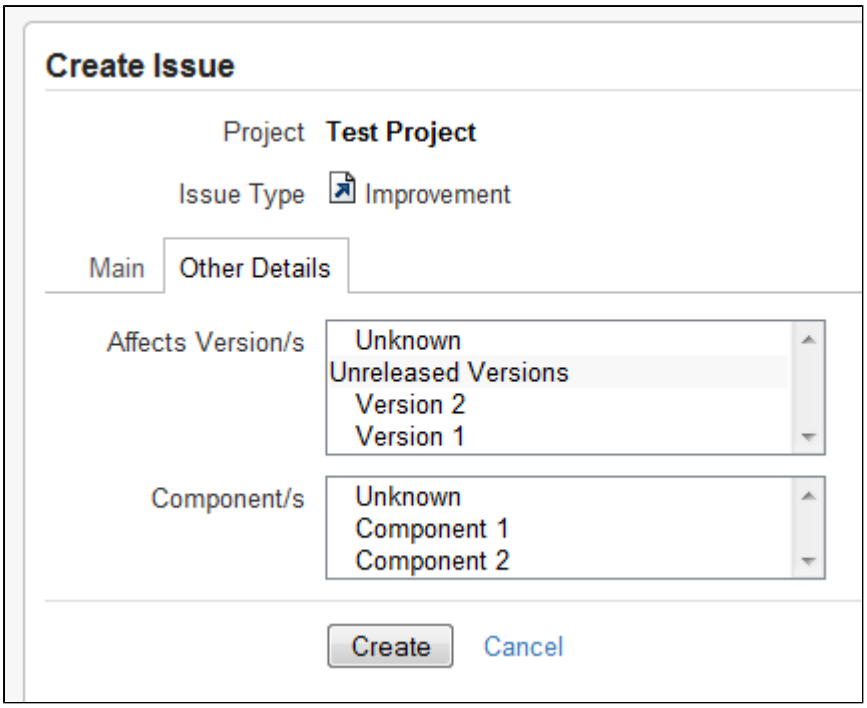
Configuring Tabs

Splitting a Screen into multiple tabs can help to group related fields. This functionality is very useful for organising complex Screens, as you can place less used fields onto separate tabs. For example, the following two screenshots show an example of configuring a simple Screen that only shows the issue 'Summary' and 'Description' on the first tab ('Main'), and 'Affected Versions' and 'Components' on the second tab ('Other Details'). You might want to put the 'Environment' field and the 'Attachments' field on their own tabs too. The final result could look like this when the screen is displayed to a user:

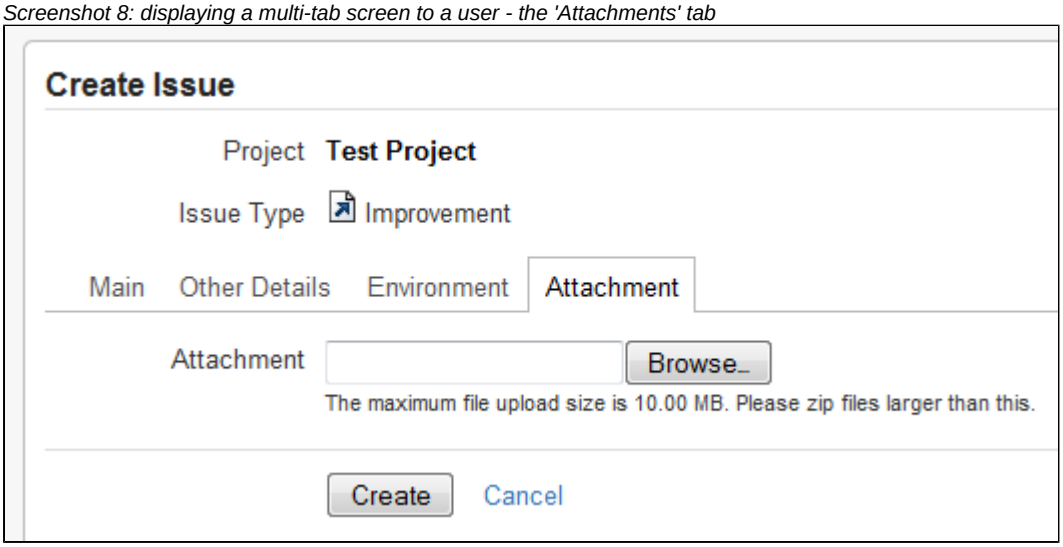
Screenshot 6: displaying a multi-tab screen to a user - the 'Main' tab



Screenshot 7: displaying a multi-tab screen to a user - the 'Other Details' tab



Screenshot 8: displaying a multi-tab screen to a user - the 'Attachments' tab



Adding a Tab

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. In the '**Add New Tab**' form, located at the bottom of the 'Configure-Screen' page (see above), enter the name of the new tab in the '**Name**' field and click the '**Add**' button.

Moving fields between Tabs

If your screen contains multiple tabs, you can move fields from one tab to another.

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. In the table of fields on the 'Configure-Screen' page (see above):
    a. For the field you wish to move, select the field's destination tab in the **Move to Tab** column. Repeat this for all the fields you wish to move.
    b. Click the '**Move**' button located at the bottom of the table in the **Move to Tab** column.

Please Note:

- Fields of type 'Date' can only be displayed in the 'Dates' area of the screen, even if they are custom fields.
- System fields on the default 'View Issue' screen (e.g. Summary, Security Level, Issue Type, etc.) are fixed and cannot be moved onto a separate tab. However, any other custom fields (except 'Date' fields) that have been added to the 'View Issue' screen can be moved onto a separate tab. This restriction only applies to the screen associated with the 'View Issue' operation, i.e. system fields can be moved onto other tabs for screens associated with operations such as 'Create Issue', 'Edit Issue', etc.

Deleting a Tab

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. The 'Configure-Screen' page will be displayed (see above). Navigate to the Tab you wish to remove.
5. Click the '**Delete**' tab link. You will be prompted to confirm your deletion.

Screenshot 9: deleting a tab

Renaming a Tab

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. The 'Configure-Screen' page will be displayed (see above). Navigate to the Tab you wish to remove.
5. The '**Rename**' text field is located in the top left of the **Configure Screen Tab** form.
6. Enter the new name of the Tab and click '**Enter**'.

Reordering Tabs

To configure the horizontal order of Tabs:

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens'**.
3. The 'View Screens' page will be displayed (see above). Click the '**Configure**' link next to the Screen of interest.
4. The 'Configure-Screen' page will be displayed (see above). Navigate to the Tab you wish to move.
5. Click the arrows (next to the name of the tab) to move that Tab left or right (in the direction of the arrow).

Activating a Screen

To make a Screen available to users, you can **either:**

- Associate the Screen with an **issue operation** (e.g. 'Create Issue'), via a **Screen Scheme** — see 'Associating Screens with Issue Operations'; **or**
- Associate the Screen with a **Workflow Transition** (e.g. 'Resolve Issue') — see 'Configuring Workflow'.

Associating a Screen with an Issue Operation

What is a 'Screen Scheme'?

A **Screen Scheme** allows you to choose which Screen will be shown to a JIRA user when they perform a particular *issue operation.* There are three issue operations for which you can choose a Screen:

- 'Create Issue' — the Screen that is shown when an issue is being created.
- 'Edit Issue' — the Screen that is shown when an issue is edited.
- 'View Issue' — the Screen that is shown when a user views an issue.

You can specify the same screen for each of these issue operations, or choose different screens for each operation.

Once you have created your Screen Scheme, you will need to activate it by associating it with a project and issue types via an Issue Type Screen Scheme.

On this page:

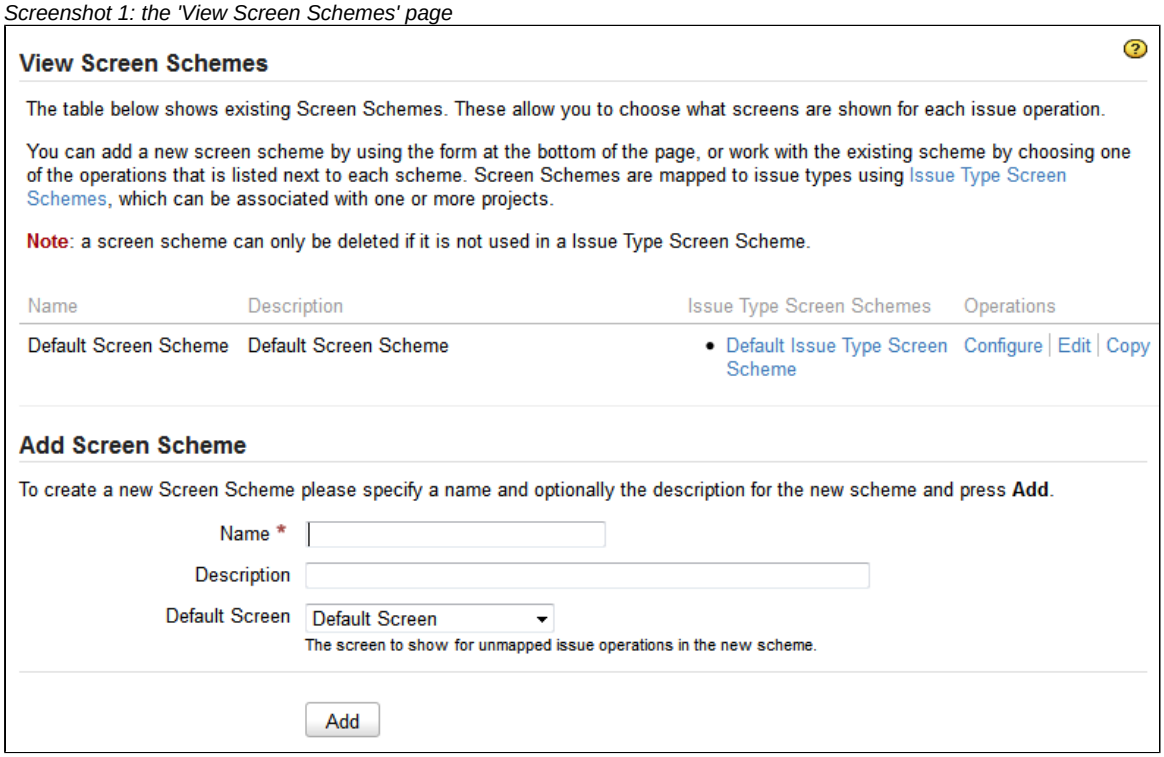
- What is a 'Screen Scheme'?
- Configuring a Screen Scheme
  - Associating a Screen with an Issue Operation
  - Editing an Association
  - Deleting an Association
- Adding a Screen Scheme
- Editing a Screen Scheme's details
- Deleting a Screen Scheme
- Copying a Screen Scheme
- Activating a Screen Scheme

Configuring a Screen Scheme

Associating a Screen with an Issue Operation

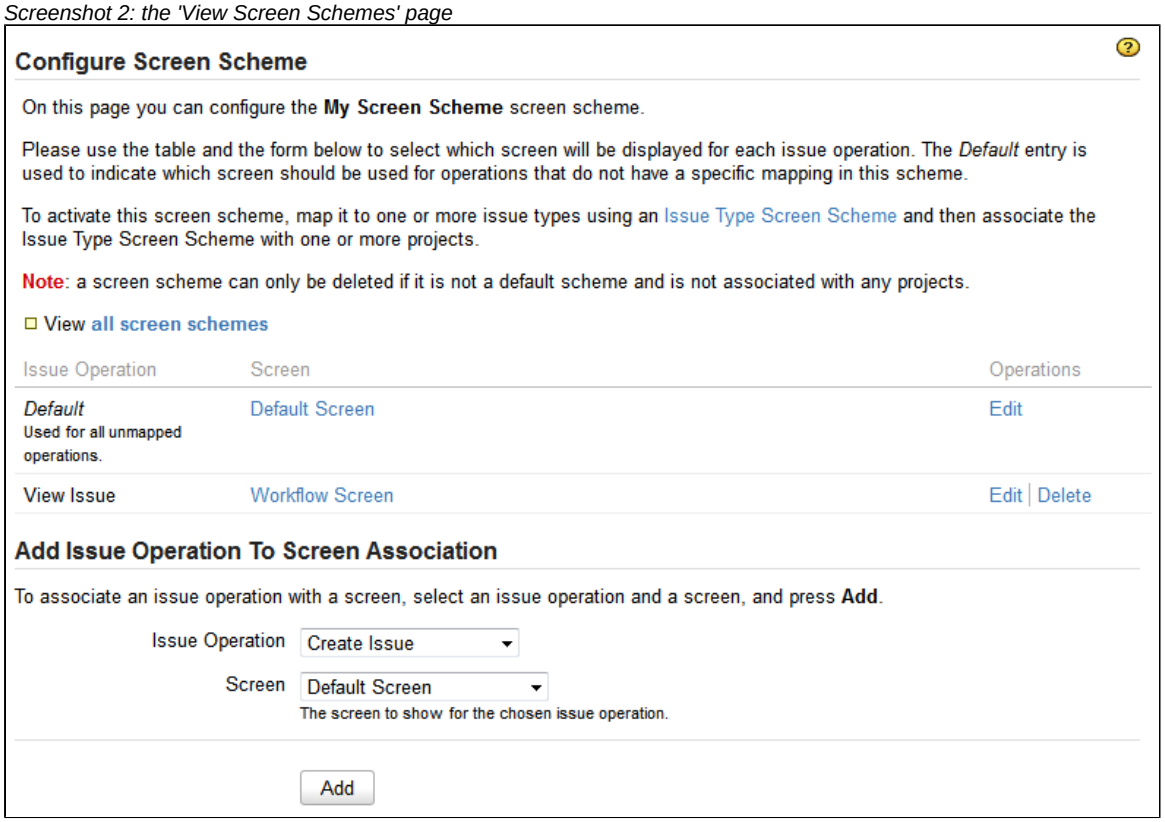
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
   Keyboard shortcut: **'g' + 'g' +** type **'sc'**
3. The 'View Screen Schemes' page will be displayed:

   Screenshot 1: the 'View Screen Schemes' page



4. Locate the Screen Scheme in which you are interested, and click the '**Configure**' link next to it. The 'Configure Screen Scheme' page will be displayed:

Screenshot 2: the 'View Screen Schemes' page



5. The '**Add Issue Operation to Screen Association**' is located at the bottom of the 'Configure Screen Scheme' page.
    - Select the Issue Operation with which you wish to associate a Screen.
    - Select the desired Screen.

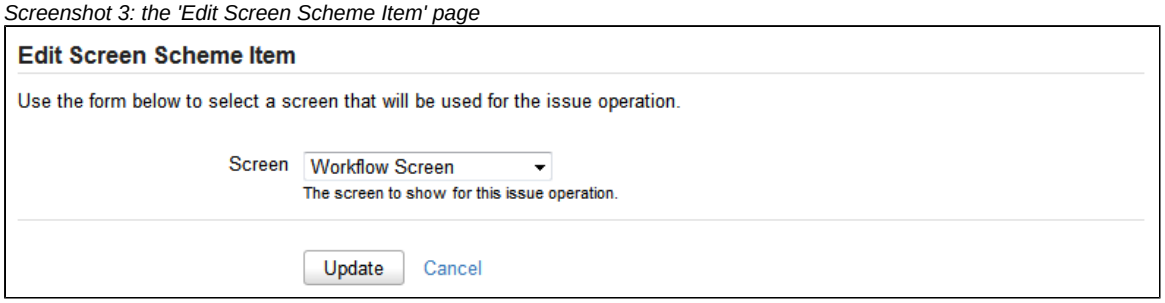
Note:

- There can only be one association for an issue operation per Screen Scheme. If all operations have been associated with a Screen, use the 'Edit' link next to each operation to change the Screen it is associated with.
- If an issue operation does not have a specific mapping to a Screen, the screen that is associated with the *Default* entry will be used for that operation. The *Default* entry cannot be deleted from a Screen Scheme. You can click the "Edit" link next to the *Default* entry to change the Screen that is associated with it.
- The 'View Issue' operation only allows you to control the layout of **custom fields** in the middle of the 'View Issue' page. The 'View Issue' page ignores all the non-custom fields on the Screen.

Editing an Association

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
3. The 'View Screen Schemes' page will be displayed (see above).
4. Locate the Screen Scheme in which you are interested, and click the 'Configure*' link next to it. The 'Configure Screen Scheme' page will be displayed (see above).
5. On the 'Configure Screen Scheme' page, click the '**Edit**' link next to the issue operation you wish to edit. The 'Edit Screen Scheme Item' page will be displayed:

Screenshot 3: the 'Edit Screen Scheme Item' page



6. Select the desired screen and click '**Update**'.

Deleting an Association

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
3. The 'View Screen Schemes' page will be displayed (see above).
4. Locate the Screen Scheme in which you are interested, and click the 'Configure*' link next to it. The 'Configure Screen Scheme'

page will be displayed (see above).
5. On the 'Configure Screen Scheme' page, click the '**Delete**' link next to the issue operation you wish to remove.

Adding a Screen Scheme

Depending on your requirements, you may want to create multiple Screen Schemes, and associate them with different projects/issue types.

To create a new screen scheme,

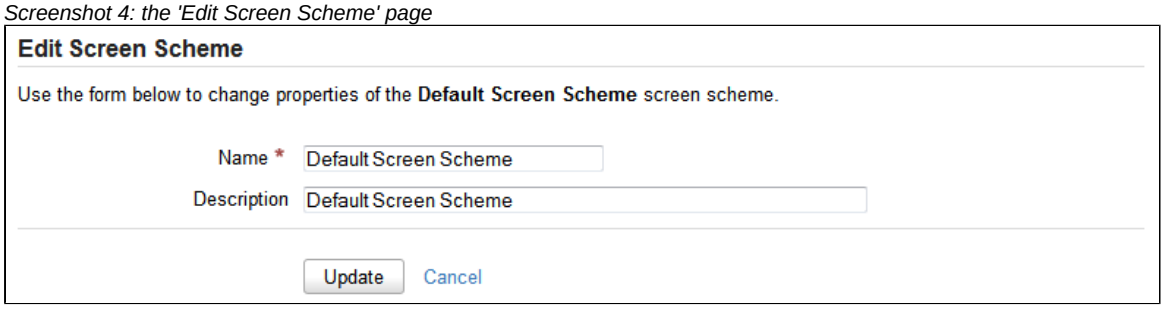
1. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
   Keyboard shortcut: **'g' + 'g' +** type **'sc'**
2. The 'View Screen Schemes' page will be displayed (see above).
3. The '**Add Screen Scheme**' form is located at the bottom of the 'View Screen Schemes' page. Enter the name of the new field screen and select a default screen. The default screen will be used for issue operations that do not have a screen associated with them.

Editing a Screen Scheme's details

To change a screen scheme's name, description or default screen,

1. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
2. The 'View Screen Schemes' page will be displayed (see above). Click the '**Edit**' link next to the desired Screen Scheme.
3. You will now be directed to the 'Edit Screen Scheme' page where you can edit the Screen Scheme's name and description and the Screen that is associated with the *Default Entry* of the scheme.

Screenshot 4: the 'Edit Screen Scheme' page

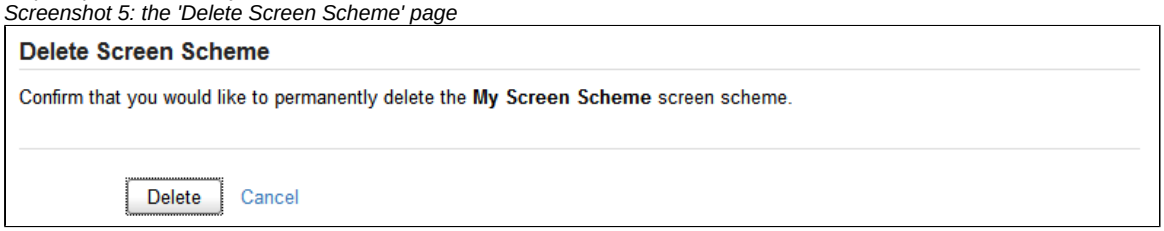


Deleting a Screen Scheme

Note that Screen Schemes that are associated with an Issue Type Screen Scheme cannot be deleted. You will first need to edit the Issue Type Screen Scheme and remove the Screen Scheme.

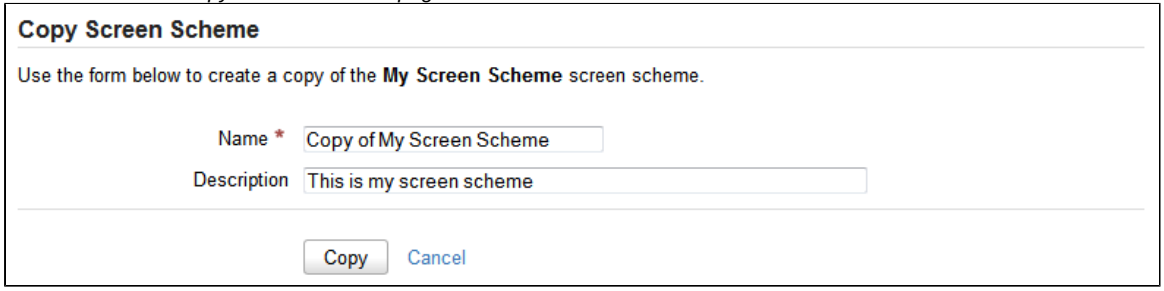
To delete a screen scheme,

1. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
2. The 'View Screen Schemes' page will be displayed (see above). Click the '**Delete**' link next to the desired Screen Scheme. You will be prompted to confirm your deletion.
   Screenshot 5: the 'Delete Screen Scheme' page



Copying a Screen Scheme

1. Select **'Administration' > 'Issues' > 'Screens' > 'Screen Schemes'**.
2. The 'View Screen Schemes' page will be displayed (see above). Click the '**Copy**' link next to the Screen Scheme you wish to copy
3. You will now be directed to the 'Copy Screen Scheme' page. Enter the name and description of the new Screen Scheme and click the '**Copy**' button.
   Screenshot 6: the 'Copy Screen Scheme' page



Activating a Screen Scheme

To activate a Screen Scheme, you need to associate it with one or more projects and issue types, using Issue Type Screen Schemes. To activate a Screen Scheme,

1. configure an Issue Type Screen Scheme to use the Screen Scheme; then
2. associate the Issue Type Screen Scheme with a project.

For details of both procedures, see 'Associating screens with Issue Types'.

Associating a Screen with an Issue Type

What is an 'Issue Type Screen Scheme'?

An **Issue Type Screen Scheme** associates a Screen Scheme with issue types, allowing you to specify different Screens for the same operation (e.g. 'Create Issue') in the same project for different issues types. For more information please see the Overview Diagram.

By default, your JIRA system contains an Issue Type Screen Scheme called 'Default Issue Type Screen Scheme'. You may want to edit this scheme, or copy it to make a new one.

On this page:

- What is an 'Issue Type Screen Scheme'?
- Configuring an Issue Type Screen Scheme
  - Associating an Issue Type with a Screen Scheme
  - Editing an Association
  - Deleting an Association
- Adding an Issue Type Screen Scheme
- Editing an Issue Type Screen Scheme
- Deleting an Issue Type Screen Scheme
- Copying an Issue Type Screen Scheme
- Associating an Issue Type Screen Scheme with a Project

Configuring an Issue Type Screen Scheme

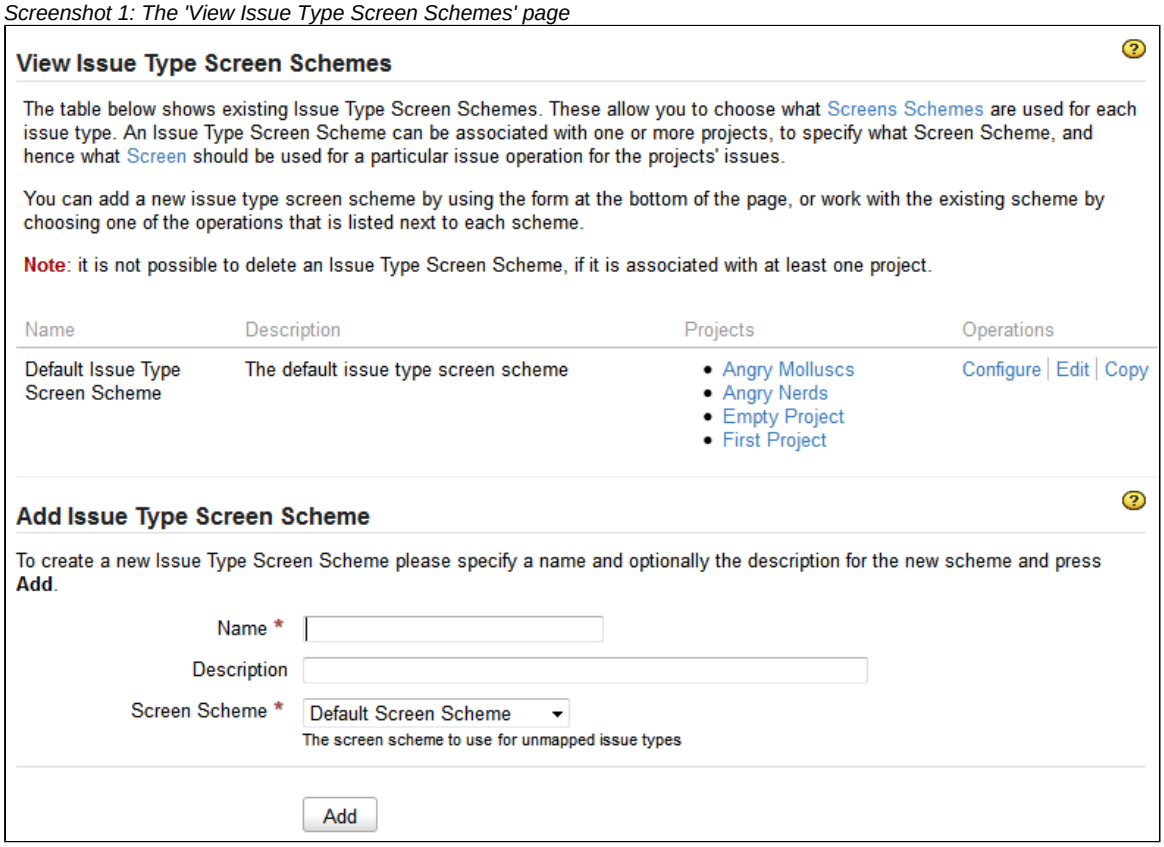
The configuration of an Issue Type Screen Scheme involves associating an issue type(s) with a particular Screen Scheme. For example, associating the 'Bug' issue type with the 'Default Screen Scheme' and then associating the 'Improvement' issue type with the 'Improvement Screen Scheme'.

Associating an Issue Type with a Screen Scheme

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the '**View Issue Type Screen Schemes**' page:

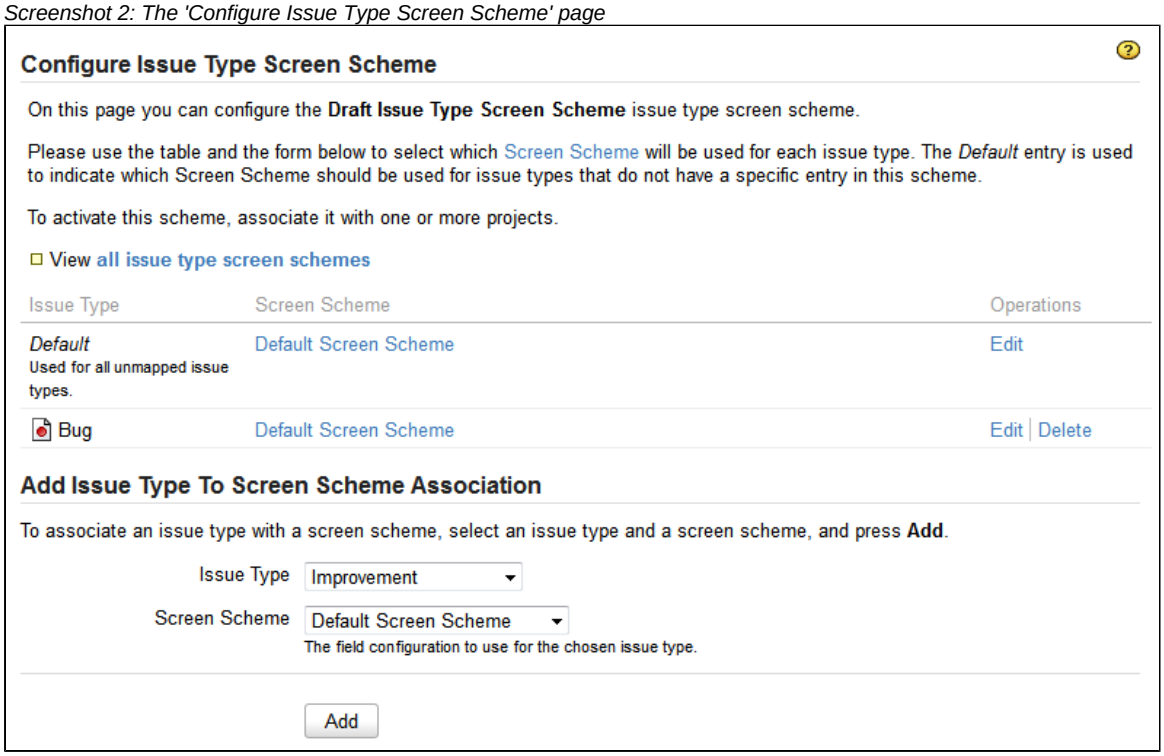
Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**

Screenshot 1: The 'View Issue Type Screen Schemes' page



3. Click the '**Configure**' link next to the desired Issue Type Screen Scheme, which opens the 'Configure Issue Type Screen Scheme' page:

Screenshot 2: The 'Configure Issue Type Screen Scheme' page



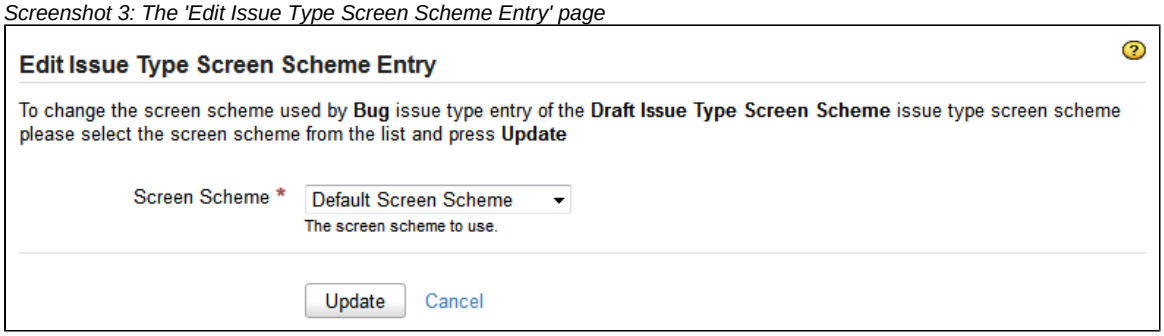
4. Scroll down to the 'Add Issue Type to Screen Scheme Association' form, located at the bottom of the 'Configure Issue Type Screen Scheme' page.
5. Select an issue type you wish to associate a Screen Scheme with.
6. Select the desired scheme.
7. Click the '**Add**' button and the new association will be added to the association list above.

Please note

- There can only be one association for each issue type. If all issue types have been associated with a Screen Scheme you can use the '**Edit**' link next to each entry to change the Screen Scheme that is associated with it.
- If there is no specific entry for an issue type, the Screen Scheme associated with the *Default* entry will be used.

Editing an Association

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the 'View Issue Type Screen Schemes' page (see above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Configure**' link next to the desired Issue Type Screen Scheme, which opens the 'Configure Issue Type Screen Scheme' page (see above).
4. Click the '**Edit**' link next to the issue type you wish to edit, which displays the 'Edit Issue Type Screen Scheme Entry' page.
   Screenshot 3: The 'Edit Issue Type Screen Scheme Entry' page



5. Select the screen whose association you wish to change, and click the '**Update**' button.

Deleting an Association

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the 'View Issue Type Screen Schemes' page (see above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Configure**' link next to the desired Issue Type Screen Scheme, which opens the 'Configure Issue Type Screen Scheme' page (see above).
4. Click the '**Delete**' link next to the issue operation you wish to remove.

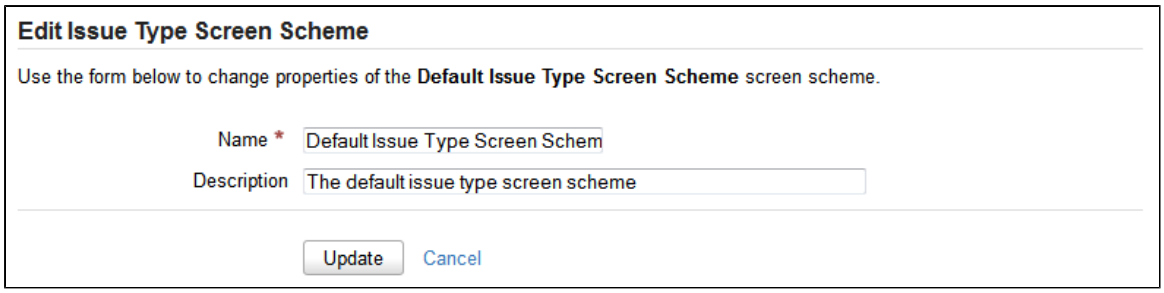
The *Default* entry is used for all issue types that do not have a specific entry in the scheme. It cannot be deleted.

Adding an Issue Type Screen Scheme

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the 'View Issue Type Screen Schemes' page (see above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. In the 'Add Issue Type Screen Scheme' form (located at the bottom of the 'View Issue Type Screen Schemes' page — see above), enter the name for the new scheme. You can optionally add a description.
4. Select a Screen Scheme for the *Default* entry in the new scheme. The '*Default* entry' will be used for issue types that do not have a specific mapping in the scheme.
5. Click the '**Add**' button. The screen will automatically update the Issue Type Screen Schemes list with the new Issue Type Screen Scheme.

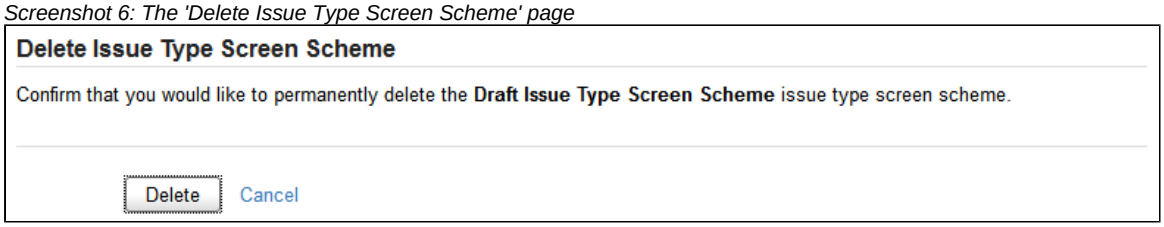
Editing an Issue Type Screen Scheme

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the 'View Issue Type Screen Schemes' page (see above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Edit**' link next to the desired Issue Type Screen Scheme to open the 'Edit Issue Type Screen Scheme' page, where you can edit the Issue Type Screen Scheme's name and description as well as the Screen Scheme of the *Default* entry.
4. Click the '**Update**' button, which returns you to the 'View Issue Type Screen Schemes' page, with your updates now applied to the Issue Type Screen Schemes list.
   Screenshot 5: The 'Edit Issue Type Screen Scheme' page



Deleting an Issue Type Screen Scheme

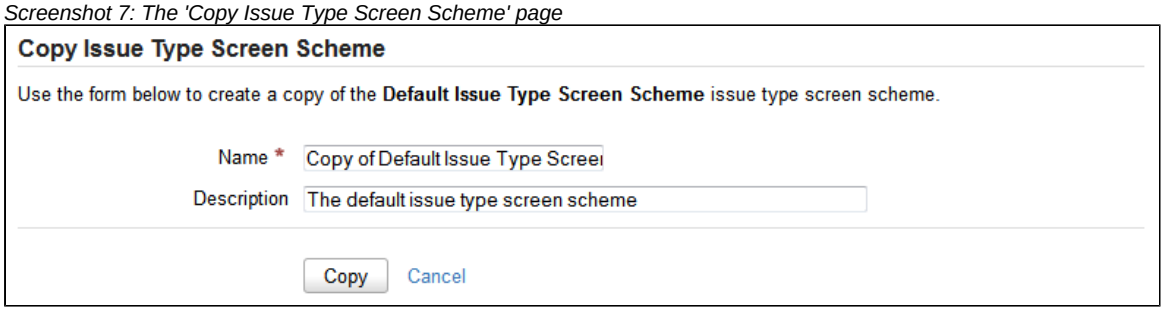
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the 'View Issue Type Screen Schemes' page (see above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Delete**' link next to the Issue Type Screen Scheme you wish to delete.
   Screenshot 6: The 'Delete Issue Type Screen Scheme' page



Issue Type Screen Schemes that are associated with a project cannot be deleted.

Copying an Issue Type Screen Scheme

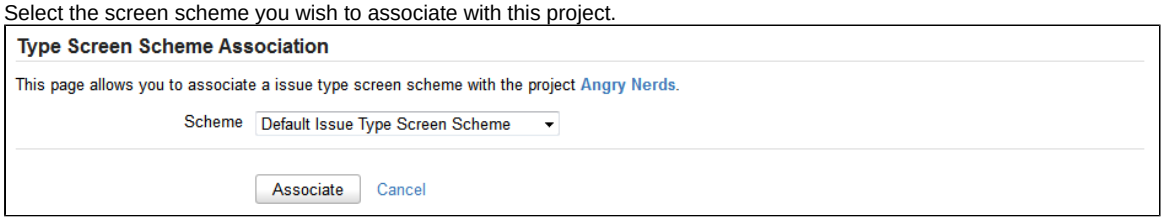
1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'Issues' > 'Screens' > 'Issue Type Screen Schemes'** to open the 'View Issue Type Screen Schemes' page (see above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'issue type screen schemes'**
3. Click the '**Copy**' link next to the field screen you wish to copy, which opens the 'Copy Issue Type Screen Scheme' page.
4. Enter the name and description of the new Issue Type Screen Scheme and click the '**Copy**' button.
   Screenshot 7: The 'Copy Issue Type Screen Scheme' page



Associating an Issue Type Screen Scheme with a Project

Once you have created and configured an Issue Type Screen Scheme to your desired settings, you can now associate the scheme with a Project. This will apply your chosen Screen Scheme to each issue type within the selected project.

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click **'Projects'** and select the project of interest. You will now see the 'Project Summary' screen (see Defining a Project).
3. On the lower left, in the 'Screens' section, click the name of the current Issue Type Screen Scheme.
4. Click the '**Actions**' dropdown menu and choose '**Use a different scheme**'.
5. Select the screen scheme you wish to associate with this project.



6. Click the '**Associate**' button.

To control which issue types apply to a project, please see 'Associating Issue Types with Projects'.

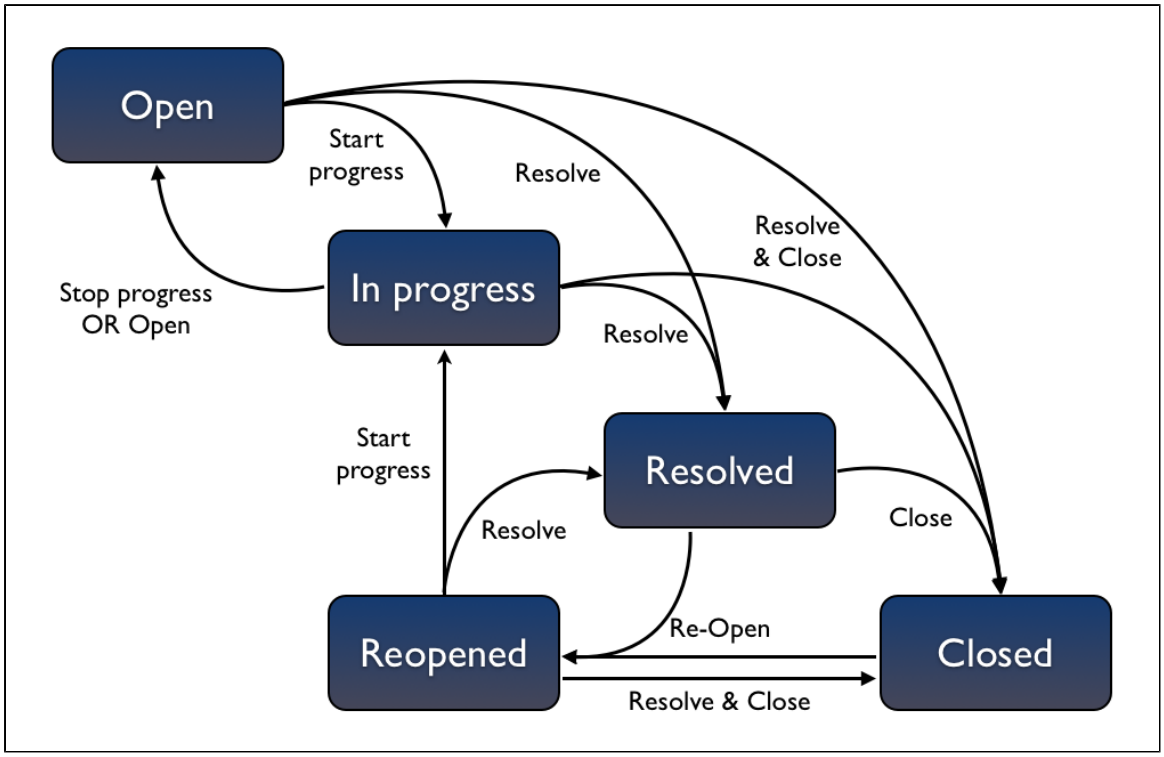
Configuring Workflow

About workflow

A JIRA *workflow* is the set of *steps (or statuses)* and *transitions* that an issue goes through during its lifecycle. Workflows typically represent business processes.

JIRA ships with a default workflow called 'jira'. This default workflow cannot be edited, but you can customise the issue lifecycle by creating additional workflows. Each workflow can be associated with particular projects and (optionally) particular issue type(s).

JIRA's default workflow

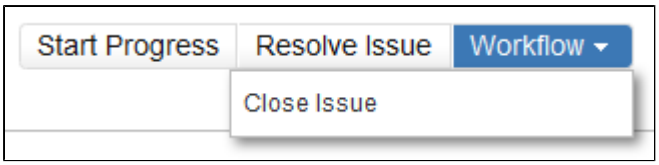


About steps and transitions

JIRA workflows consist of **steps** and **transitions**:

- A **step** represents a workflow's current status for an issue. An issue can exist in one step only at any point in time. Each workflow step corresponds to a **linked status**. When an issue is moved into a particular step, its **status** field is updated to the value of the step's **linked status**. In the diagram above, the blue boxes represent steps/statuses.
  When defining a step, you can optionally specify **properties**, one of which allows you to make an issue uneditable while it is in that step.
  ✅ Although steps and statuses are treated separately when administering workflows, they are effectively synonymous concepts in JIRA and as such, it usually helps to name a step after its linked status.

- A **transition** is a link between two steps. A transition allows an issue to move from one step to another step. For an issue to be able to progress from one particular step to another, a transition must exist that links those two steps. Note that a transition is a *one-way* link, so if an issue needs to move back and forth between two steps, two transitions need to be created. In the diagram above, the arrows represent transitions.
  The available workflow transitions for an issue are listed on the issue's 'view issue' page. A user can execute a transition (i.e. move the issue through workflow) by clicking one of the available links, e.g.:



When defining a transition, you can optionally specify:

- A **screen** to be displayed to the user — this is useful if you need the user to provide input before completing the transition.
- **Conditions** — these control who can perform a transition (i.e. who can see the transition link on the 'view issue' page).
- **Validators** — these check that any user-supplied input is valid before performing the transition.
- **Post functions**— these perform particular actions after the transition is complete, e.g.:
  - Assign the issue to a particular user.
  - Send an email notification.
  - Update a field in the issue.

On this page:

- About workflow
  - About steps and transitions
  - About 'Open' and 'Closed' issues
- Creating a workflow
- Editing a workflow
  - Editing an inactive workflow
  - Editing an active workflow
    - Limitations when editing an active workflow
- Using the 'Workflow Designer'
  - Using the main toolbar
  - Using the 'Statuses' panel
  - Using the 'Global Transitions' panel
  - Using the workflow design area
    - Editing or deleting annotations
  - Accessing the 'Workflow Designer'
- Working with steps
  - Adding a step
  - Editing a step
  - Using step properties
  - Deleting a step
- Working with transitions
  - Adding a transition
  - Editing or deleting a transition
  - Using a screen with a transition
  - Applying conditions to transitions
    - JIRA's built-in conditions
    - Adding a condition
    - Combining conditions into groups
  - Applying validators to transitions
    - Adding a validator
  - Applying post functions to transitions
    - Adding a post function
    - Using a post function to set a field
    - Using a post function to send a notification
  - Working with transition properties
  - Customising workflow transitions on the 'view issue' page
    - Changing the number of transition buttons
    - Changing the order of transition buttons and 'workflow' menu items
  - Using 'common transitions'
- Using XML to create a workflow
  - Importing an XML workflow into JIRA
  - Copying a workflow between systems

See also:

- Activating Workflow
- Adding a Custom Event
- Configuring the Initial Status
- Workflow Properties

About 'Open' and 'Closed' issues

Within JIRA (e.g. in the **Assigned To Me** gadget and other gadgets), an issue is determined to be **Open** or **Closed** based on the value of its **Resolution** field — not its **Status** field.

- An issue is determined to be **Open** if its **Resolution** field has not been set.
- An issue is determined to be **Closed** if its **Resolution** field has a value (e.g. **Fixed**, **Cannot Reproduce**).

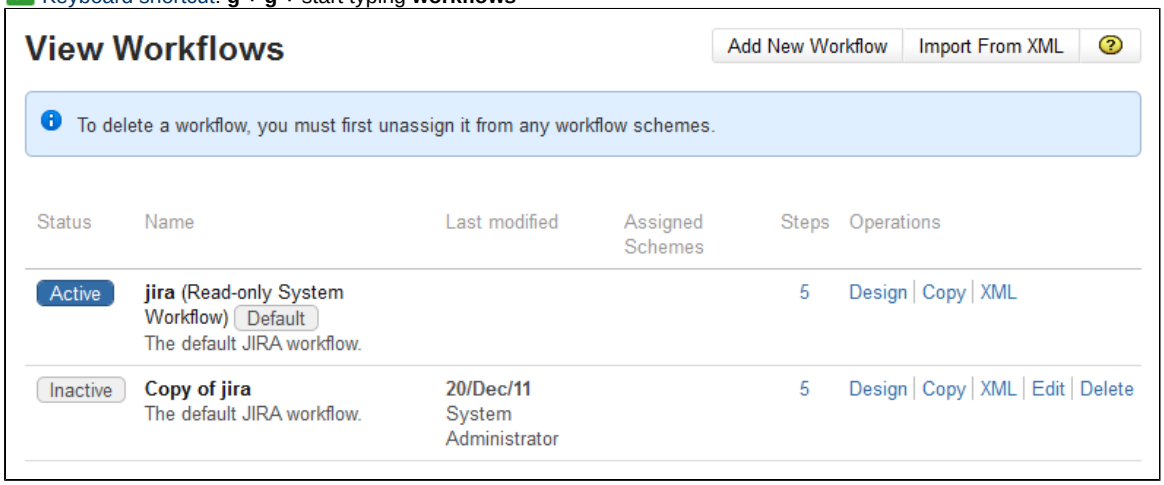
This is true regardless of the current value of the issue's **Status** field (**Open**, **In Progress**, etc).

So if you need your workflow to force an issue to be **Open** or **Closed**, you will need to set the issue's **Resolution** field during a transition. There are two ways to do this:

- Set the **Resolution** field automatically via a post function.
- Prompt the user to choose a **Resolution** via a screen.

Creating a workflow

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system.
   Keyboard shortcut: **g + g +** start typing **workflows**



   Some of the features in the screenshot above, such as **Import From XML**, will not available unless you have logged in as a user with the **JIRA System Administrators** global permission.
3. Create a new workflow in JIRA using either of the following methods:
   - **Create a 'blank' workflow** by first clicking the **Add New Workflow** button and in the resulting **Add New Workflow** dialog box:
     a. Type a **Name** (usually 2-3 words) to identify your new workflow.
     b. *(Optional)* Type a detailed **Description** of your new workflow.
     c. Click the **Add** button. The **Workflow Designer** will open, showing your new workflow containing one step, called **Open**, which has an incoming transition called **Create**.
   - **Copy an existing workflow** (useful if your new workflow can created by applying modifications to an existing workflow), by clicking the **Copy** link next to the existing workflow and in the resulting **Copy Workflow** dialog box:
     a. Type a **Workflow Name** (usually 2-3 words) to identify your new workflow.
     b. *(Optional)* Type a detailed **Description** of your new workflow.
     c. Click the **Copy** button. Your new workflow will contain the same steps and transitions as the workflow you copied.
        If you are copying the default workflow (above) and wish to rename the workflow transition buttons on the 'view issue' page, you must delete the `jira.i18n.title` and `jira.i18n.description` properties from all transitions in the copied workflow. Otherwise, the default names (i.e. values of these properties) will persist. Read more about transition properties.
4. Once you have created your new workflow you may want to customise it using the **Workflow Designer** by adding and/or editing steps and transitions (below) — especially if you have created a blank workflow.
5. When you have finished customising your new workflow, see Activating workflow for details on how to use it with a JIRA project.

Editing a workflow

Editing a workflow means that you are modifying the steps and transitions that make up a workflow. Read more about steps and transitions on this page.

The process for editing a workflow differs depending on whether you are editing an **inactive workflow** or an **active workflow**. Restrictions are placed on the modifications you can make to an active workflow, due to the impact the changes will have on projects and/or issue types that the workflow is applied to.

Editing an inactive workflow

An inactive workflow is a workflow that is not currently being used by any projects.

Because there are no issues currently transitioning through an inactive workflow, you can simply use the **Workflow Designer** directly to edit the workflow's steps and transitions (below).

Editing an active workflow

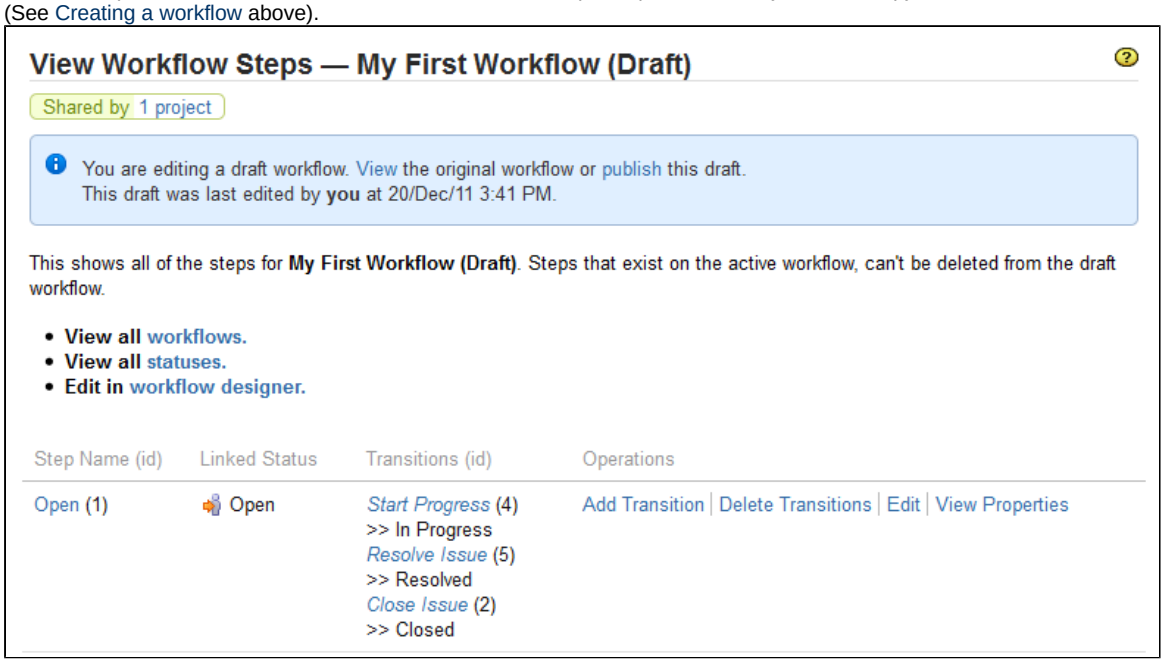
An active workflow is a workflow that is currently being used by one or more projects.

To edit an active workflow, you must first create a 'draft' of it. You can then:

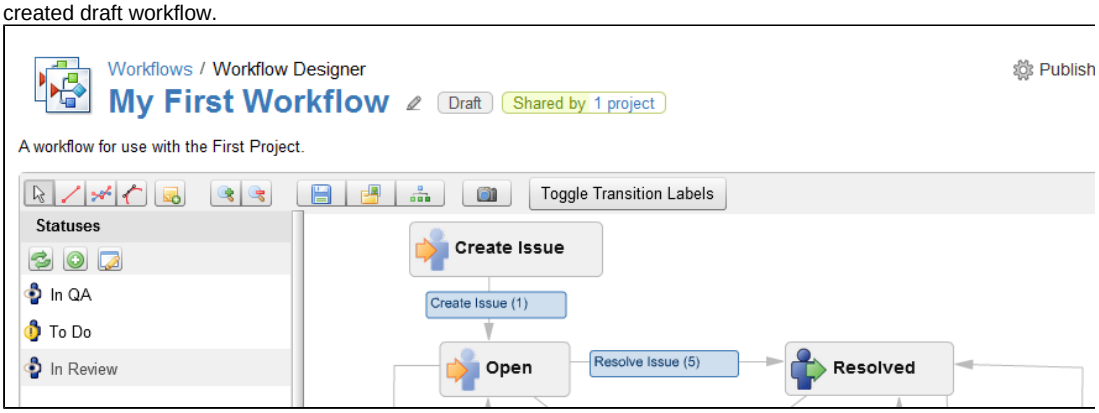
- Make quick edits to your live draft with the benefit of real-time validations
- Publish your changes, with the option of saving your original workflow as an inactive backup.

To edit an active workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Do one of the following to create your draft:
   - Click the **Create Draft** link next to the active workflow you wish to edit. The **View Workflow Steps** page of your newly created draft workflow is displayed.
     It is not possible to create a draft of the default workflow (above). You can only create a copy of the default workflow. (See Creating a workflow above).

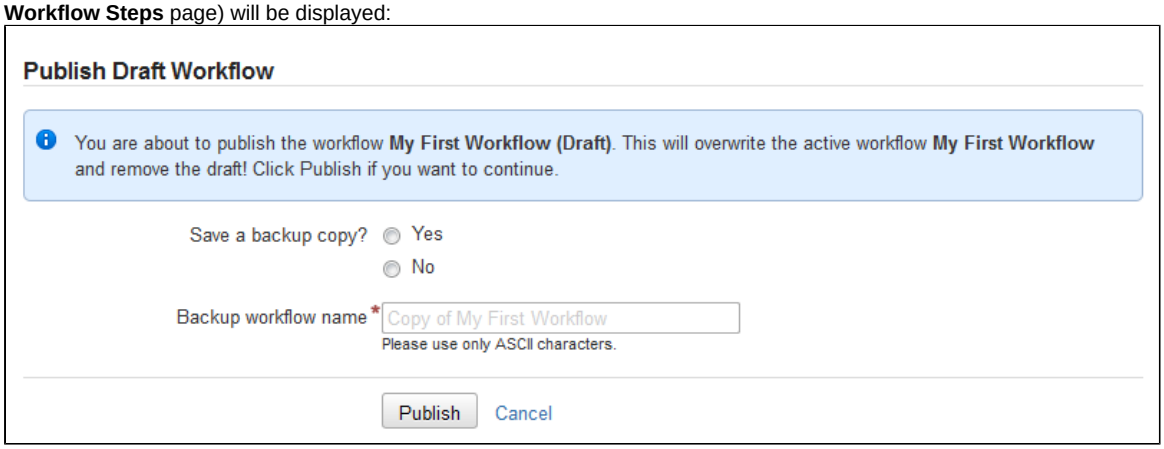


   - Click the **Design** link next to the active workflow you wish to edit. The **Workflow Designer** page (below) is displayed.
     - Click **Create Draft** near the top of the page. The **Workflow Designer** page will be reloaded, showing your newly created draft workflow.



   - Click the hyperlinked number (under the **Steps** column) next to the active workflow you wish to edit. The **View Workflow Steps** page is displayed.
     - Click the '**Create** a draft workflow' link in the information message box near the top of the page. The **View Workflow Steps** page will be reloaded, showing your newly created draft workflow.
4. After using the **Workflow Designer** or the **View Workflow Steps** pages to add and/or edit your draft workflow's steps and transitions (below), click **Publish** at the top right of the **Workflow Designer** (or the '**publish** this draft' link in the information

message box near the top of the **View Workflow Steps** page). A confirmation dialog box (or confirmation page if you used the **View Workflow Steps** page) will be displayed:



5. Select whether you wish to save the original workflow as an inactive copy. If you choose to retain the original workflow, enter a name for the inactive copy.
6. Click **Publish** to publish your draft (i.e. commit your changes to the active workflow).

Please Note: After creating a draft of an active workflow, you will be able to edit the draft workflow as described in the sections below. Any changes that you make to the draft will not affect the active workflow until you publish the draft.

Limitations when editing an active workflow

Please note that the following limitations apply when editing an active workflow (i.e. a draft workflow):

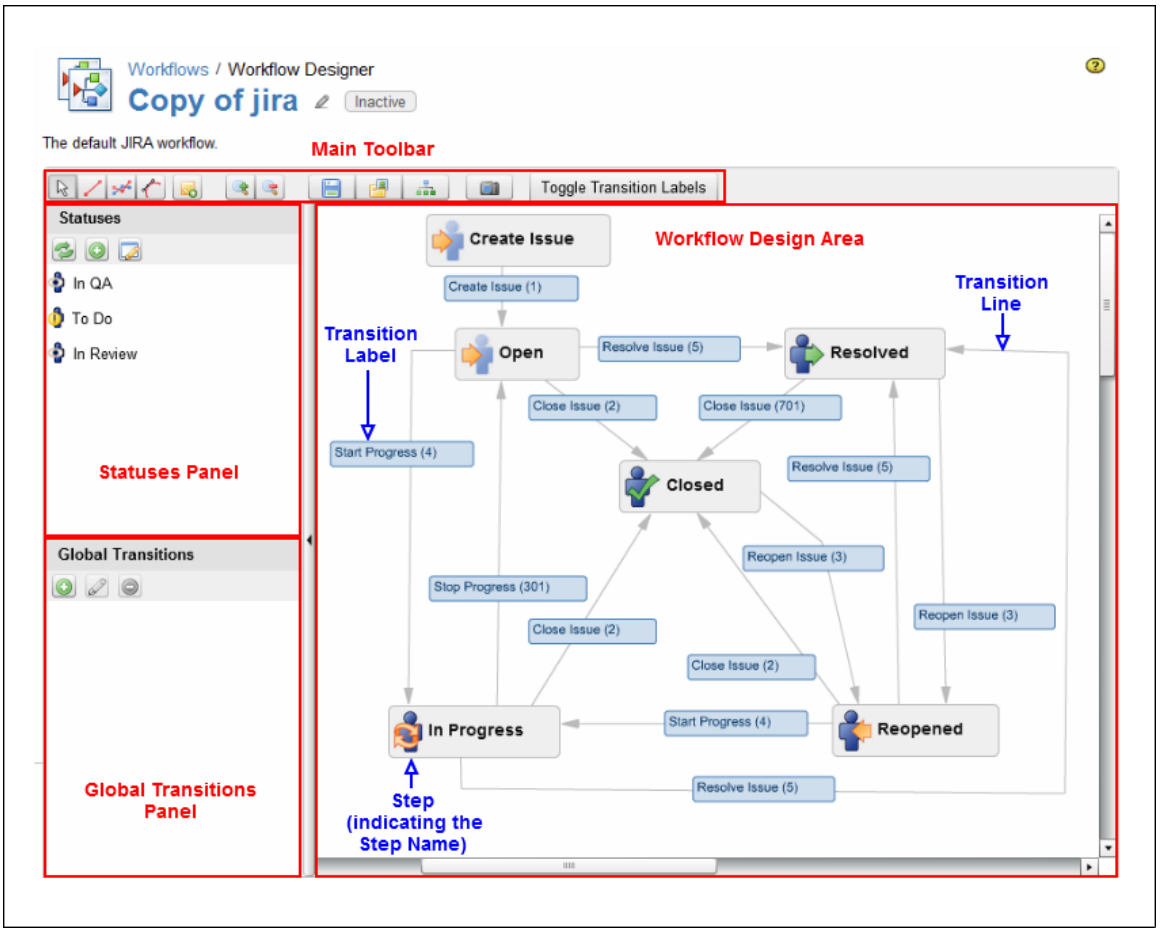
- Workflow steps cannot be deleted.
- A step's associated Status cannot be edited.
- If a step has no outgoing transitions, it cannot have any new outgoing transitions added.
- A step's Step ID cannot be changed.

If you wish to make any of the modifications listed above, then you will need to copy the workflow (see Creating a workflow above), modify the copy and then activate it. Be aware that working with a copy of an active workflow, is ***significantly slower than editing an active workflow***, particularly for large JIRA sites.

Using the 'Workflow Designer'

The **Workflow Designer** allows you to visualise the entire layout of your workflow as well as create and edit a workflow's steps and transitions visually.

The **Workflow Designer**'s interface consists of the four areas indicated in the following diagram.



Please Note:

- The **Statuses** and **Global Transitions** panels are expandable. If these panels are not visible (i.e. collapsed), you can expand these panels by clicking the vertical bar (containing a small arrow mid-way along the bar) on the left-hand edge of the workflow design area.
- Keyboard shortcuts are not available in the **Workflow Designer**.

Using the main toolbar

The main toolbar contains tools (on buttons) for visually arranging your workflow's steps/statuses and transitions on the workflow design area.

Button	Description
	Select Tool Selects, moves or edits any item on the workflow design area. Selected items are highlighted in red. Click away from any item in the workflow design area to clear the selection.
	Create Transition — Straight Line Creates a transition with a single straight line. See Adding a transition (below) for details.
	Create Transition — Polygonal Line Creates a transition with multiple straight lines. See Adding a transition (below) for details.
	Create Transition — Bezier Line Creates a transition with a curved line. See Adding a transition (below) for details.
	Create Annotation Adds a 'sticky note' to the workflow design area into which you can write a short description or annotation to add to your workflow's layout. Annotations only appear in the workflow design area and are not visible when viewing workflows from the 'view issue' page (by clicking the **View Workflow** link on that page).

	Zoom In
	Zoom Out
	Save Layout Saves the current position of all items on the workflow design area (including steps, transitions and annotations). If you have moved items around on the workflow design area and wish to save their positional layout, do not forget to use this feature before leaving the **Workflow Designer**.
	Load Saved Layout Loads the previously saved positional layout of all items on the workflow design area.
	Auto Layout Repositions all items on the workflow design area automatically.
	Save Snapshot Image Saves an image of the current position of all items on the workflow design area in PNG format.
	Toggle Transition Labels Switches between hiding or revealing transition labels, each of which indicate their **Transition (id)**. To be able to edit all aspects of a transition (including its conditions, validators, post functions and properties), transition labels must be visible. If transition labels are hidden, you can only edit the **Transition Name**, **Description**, **Destination Step** and **Transition View** of a transition by double-clicking its transition line. See the **Add Workflow Transition** page (below) for details.

Using the 'Statuses' panel

The **Statuses** panel lists all available JIRA statuses which have not already been associated with a step on the workflow design area. It also provides tools (on buttons) for adding new statuses or editing existing ones.

Button	Description
	Refresh List Refreshes the statuses list.
	Add New Status Adds a new global status to JIRA. See Defining 'Status' field values for details.
	Status Editor Opens the **Status Editor** dialog box which allows you to edit an existing global status in JIRA. This dialog box is functionality very similar in to the **View Statuses** page. See Defining 'Status' field values for details.

You can use the **Statuses** panel to add a step to your workflow. See Adding a step (below) for details.

Using the 'Global Transitions' panel

The **Global Transitions** panel lists all global transitions used in your workflow. A global transition is one in which the destination step of the transition has all other steps in the workflow as incoming steps, but provides the added advantage of only requiring you to edit or update this transition in one place.

Please Note:

- Global transitions are similar to (but not the same as) 'common transitions'. See Using 'common transitions' (below) for details.
- To avoid clutter, any global transitions added to your workflow do not appear on the workflow design area. However, they do appear in the **Global Transitions** panel as well as every step of your workflow's **View Workflow Steps** page (below).

The **Global Transitions** panel also provides tools (on buttons) for adding new global transitions, as well as editing or deleting existing ones.

Button	Description

	Add New Transition Adds a new global transition to the workflow. See Adding a transition (below) for details.
	Edit Selected Transition Edits the currently selected global transition in the global transitions panel. See Editing a transition (below) for details.
	Delete Selected Transition Deletes the currently selected global transition in the global transitions panel, from the workflow.

Using the workflow design area

The workflow design area shows the layout of your workflow's steps as well as transitions, consisting of transition lines and labels.

Use the **Select Tool** (above) to move these items around workflow design area.

Please Note:

- A workflow depicted in the workflow design area is what regular JIRA users will see when viewing the workflow from the 'view issue' page (by clicking the **View Workflow** link on that page).
- If significant changes have been made to items in the workflow design area, do not forget to use the **Save Layout** tool (above) to save the layout.

Editing or deleting annotations

Once you have added an annotation using the **Create Annotation** tool (above), you can do the following to edit or delete the annotation:

- To resize an annotation, drag the edges or corners of the 'sticky note'.
- To edit the text inside an annotation, simply click inside the annotation and begin modifying the text.
- To remove an annotation, click the **X** at the top-right of the 'sticky note'. You will be prompted to confirm this action.

Accessing the 'Workflow Designer'

To access the 'Workflow Designer' page for any workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system.
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Click the **Design** link associated with the relevant workflow. The **Workflow Designer** page is displayed, showing your workflow's steps and transitions.

Working with steps

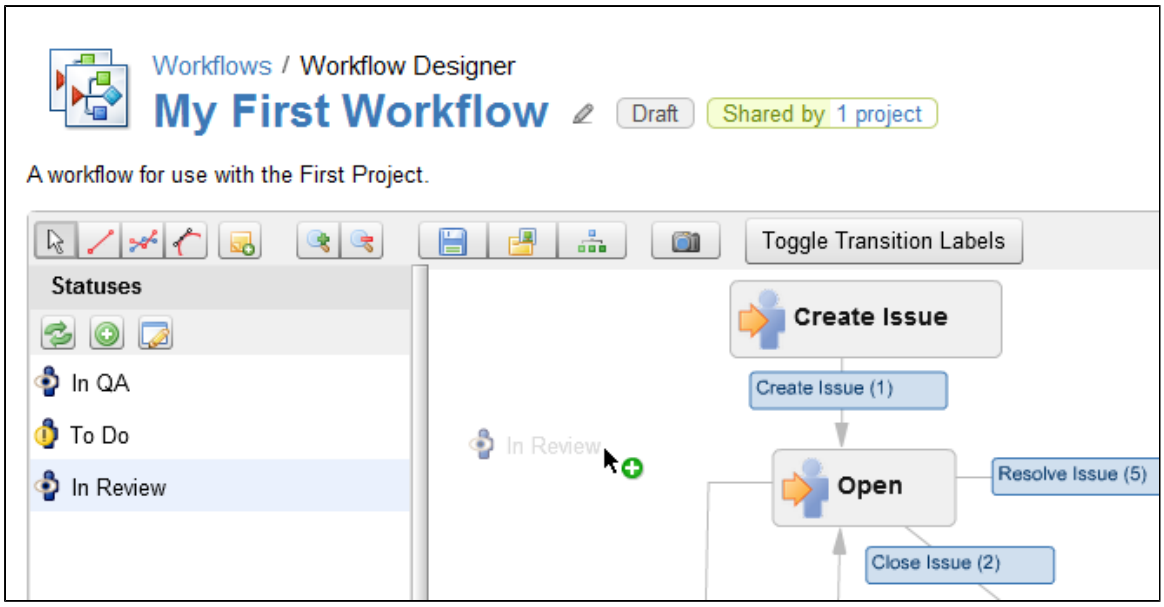
Adding a step

To add a new step to a workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to add the new step to the workflow.

Using the 'Workflow Designer' page to add the new step

1. On the **View Workflows** page, click the **Design** link next to the workflow that you wish to add a step to. The **Workflow Designer** page is displayed, showing your workflow's existing steps as well as the lines and labels of transitions between these steps (in the workflow design area).
2. Drag an available status from the **Statuses** panel list and drop it onto the workflow design area. A new step will be added to your workflow.



Please Note:

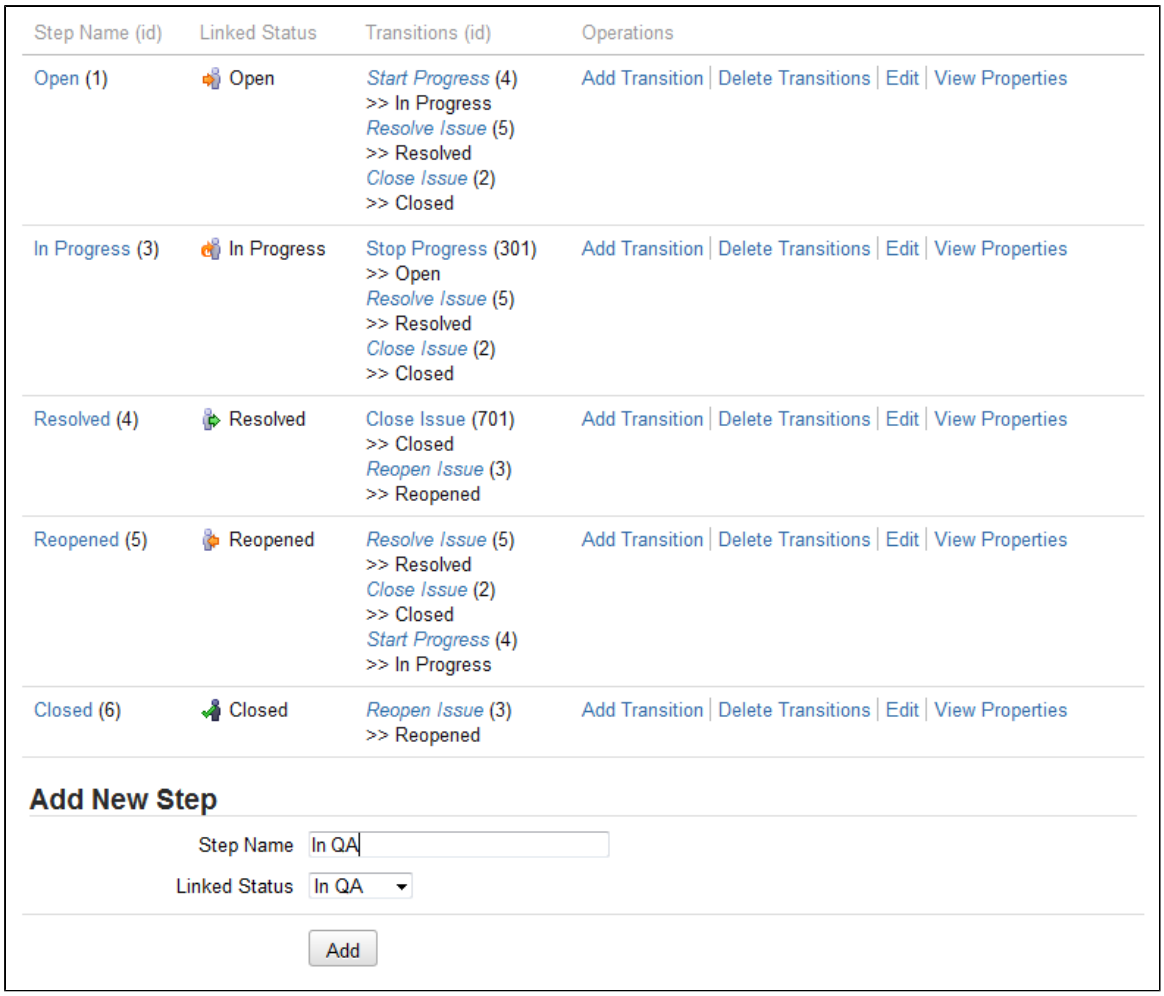
- The name of the status, which is equivalent to the **Linked Status** indicated on the **View Workflow Steps** page (below) is assigned to the step's **Step Name (id)**. This step name is depicted on step you dragged from the **Statuses** panel to the workflow design area.
- Because steps on the workflow design area depict step names, if use the **Statuses** panel on the **Workflow Designer** (above) to change the name of a status that has already been added to the workflow design area, the step name will remain unchanged in this area. To change the name of a step, see Editing a step (below).

Using the 'View Workflow Steps' page to add the new step

1. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow that you wish to add a step to. The **View Workflow Steps** page is displayed, showing the existing steps that make up the workflow and each step's **Linked Status** and **Outgoing Transitions** (under **Transitions (id)**).

   Please Note:

   - The **Add New Step** form appears below the list of steps. However, this will only be shown if the workflow is inactive or you are editing the draft of an active workflow.
   - If no fields appear in this form, then all available statuses defined in your JIRA installation have been used in your workflow and you will need to define a new status.



2. In the **Step Name** field, type a short name for the step.
   It is often useful to use the name of the corresponding linked status (in the following step).
3. In the **Linked Status** field, select the status that corresponds to this step.
   Each status can only correspond to one step in each workflow. Hence, if all statuses are linked to steps in this workflow, you may need to define a new status.
4. Click the **Add** button. The **View Workflow Steps** page will now show your new step in the list.

Be aware that some gadgets (such as **Assigned To Me** and **In Progress**) will not display data for issues in and after the **Resolved** step.

Editing a step

To edit an existing step in a workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Click the hyperlinked number (under the **Steps** column) next to the workflow whose step you wish to edit. The **View Workflow Steps** page is displayed, showing the steps that make up the workflow (as shown under Adding a step above).
   - On the **View Workflow Steps**page, you can click the following link of any step:
     - **Add Transition** — to add an **Outgoing Transition** to that step. See the **Add Workflow Transition** page (below) for details.
     - **Delete Transitions** — to delete one or more **Outgoing Transitions** of that step. This link is only available if the step has at least one outgoing transition.
     - **Edit** — to edit the step's **Name** or **Linked Status**.
     - **View Properties** — to view and edit the step's **Properties**. See Using step properties (below) for details.
     - **Delete Step** — to do just that. This link is only available if the step has no incoming transitions.
   - Alternatively, you can access the **View Workflow Step** page to edit a step.
     To access this page, click the linked name of the step you wish to edit in the **Step Name (id)** column (e.g. **Open** or **In QA** created in Adding a step above). The step's **View Workflow Step** page is displayed.

- On the **View Workflow Step** page, the following information is shown about the step:
  - On the left of the **View Workflow Step** page:
    - **Linked Status** — the status (usable globally) to which your specific workflow step is linked.
  - In the **Workflow Browser** section on the right:
    - **Incoming Transitions** — that is, transitions whose **Destination Step** is this step. To allow issues to move into this step, there must be at least one incoming transition.
    - **Outgoing Transitions** — that is, transitions whose **Originating Step** is this step. To allow issues to move out of this step, there must be at least one outgoing transition.
- On the **View Workflow Step** page, you can:
  - Click '**Add** outgoing transition' to do just that to the step. See the **Add Workflow Transition** page (below) for details.
  - Click '**Delete** outgoing transitions' to delete one or more **Outgoing Transitions** of the step. This option is only available if the step has at least one outgoing transition indicated in the **Workflow Browser** section.
  - Click '**Edit** step' to edit the step's **Step Name** or **Linked Status**.
  - Click '**Delete** Step' to do just that. This option is only available if the step has no incoming transitions indicated in the **Workflow Browser** section. See Deleting a step (below) for details.
  - Click '**View** step's properties' to view and edit the step's **Properties**. See Using step properties (below) for details.
  - View and edit any of the step's **Incoming Transitions** or **Outgoing Transitions**, by clicking the name of a transition in the **Workflow Browser** section. See Adding a transition, Adding a validator and Adding a post function (below) for details.

Using step properties

You can use a workflow step's properties to prevent issues from being edited when they are at that particular workflow step. For example, in the default workflow (above), issues in the **Closed** step/status cannot be edited, even by users who have the '**Edit Issue**' permission.

Please Note:

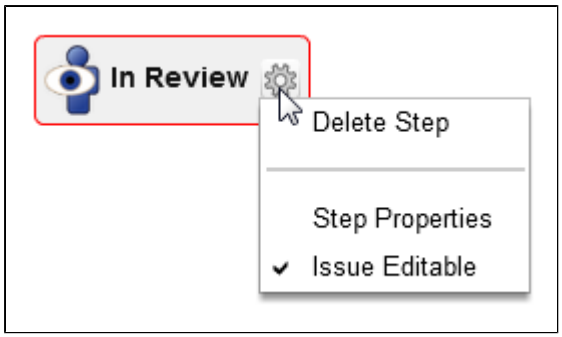
- Issues which cannot be edited cannot be updated using Bulk edit either.
- You can only edit the properties of a workflow's step if that workflow is editable (i.e. if that workflow is either inactive or a draft of an active workflow). See Editing an inactive workflow for more information.

To stop issues from being editable in a particular workflow step, set the step's `jira.issue.editable` property to `false` as follows:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to set the workflow step's property.

Using the 'Workflow Designer' page to set the step's property

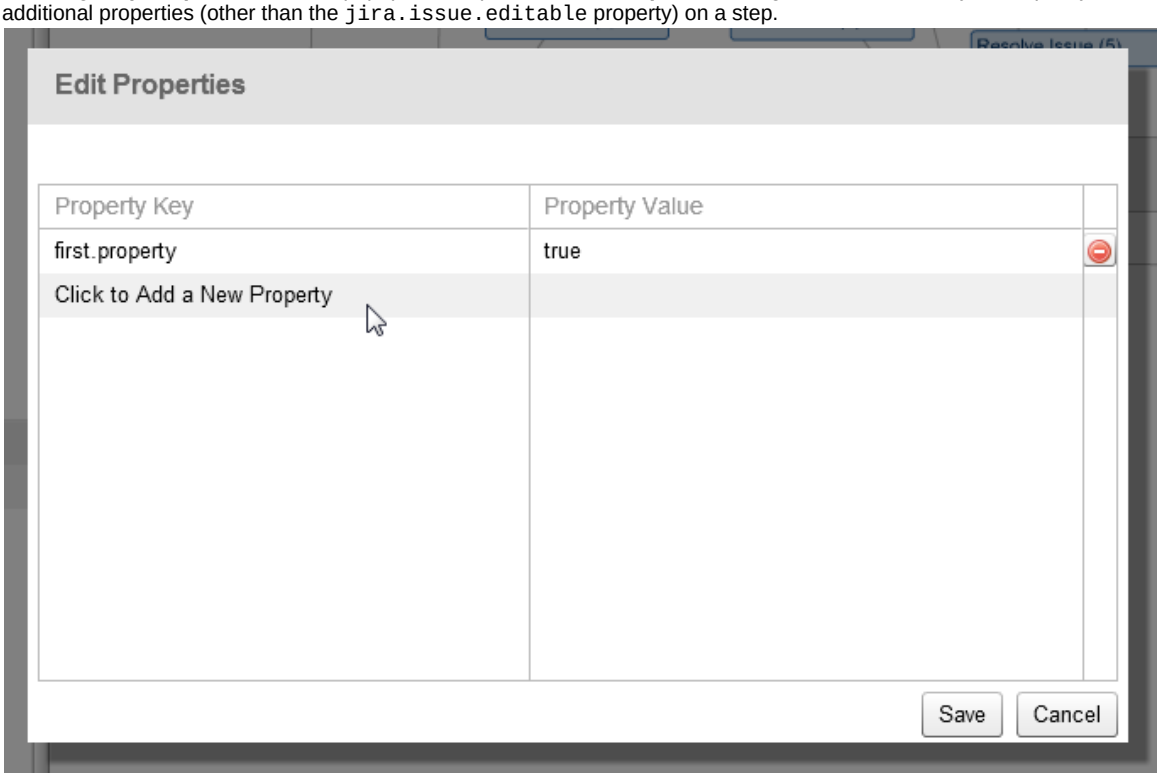
1. On the **View Workflows** page, click the **Design** link next to the workflow whose step you wish to make uneditable. The **Workflow Designer** page is displayed, showing the steps that make up your workflow in the workflow design area (as shown under Using the 'Workflow Designer' above).
2. In the workflow design area, move the mouse pointer over the relevant step and click the cog icon that appears to the right of the step to reveal a popup menu.



3. From the popup menu, select **Issue Editable** to remove the tick from that menu item. This action sets the step's `jira.issue.editable` property and its value to `false`.

Please Note:

- Selecting **Issue Editable** from the popup menu again (to add the tick) remove the step's `jira.issue.editable` property (or sets the value of the `jira.issue.editable` property key to `true`).
- Selecting **Step Properties** from the popup menu opens the **Edit Properties** dialog box, which allows you to specify additional properties (other than the `jira.issue.editable` property) on a step.



On the **Edit Properties** dialog box, you can:

- Add a new property to the step.
- Edit a property's key or value, by simply clicking the property's key or value to begin editing it.
- Delete a property, by clicking the icon to the right of the property and its value.

Using the 'View Workflow Steps' page to set the step's property

1. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose step you wish to make uneditable. The **View Workflow Steps** page is displayed, showing the steps that make up the workflow (as shown under Adding a step above).
2. Click the **View Properties** link that corresponds to the relevant step. The **View Workflow Step Properties** page is displayed, showing the step's existing properties (if any). The **Add New Property** form appears below the list of existing properties (if any have already been defined).

   The **Add New Property** form will not appear if the step's workflow is not editable.
3. In the **Property Key** field, type: `jira.issue.editable`
4. In the **Property Value** field, type: `false`
5. Click the **Add** button.

It is not possible to edit a step's properties on this page. To change any property's key or value (or both), you must first delete the property you wish to change and add the new updated property.

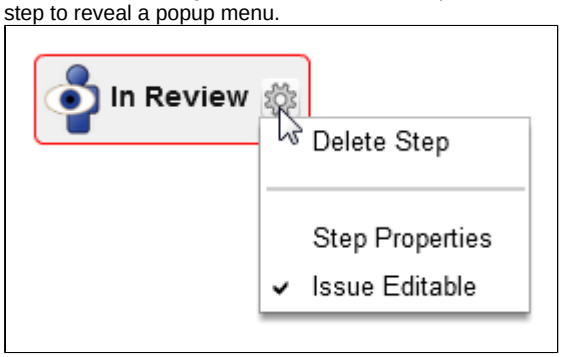
Deleting a step

Please Note: A step can only be deleted if it has no incoming transitions. Also note that you cannot delete a step from an active workflow — see Limitations for a workaround.

To delete a step from a workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to delete the step from the workflow.

Using the 'Workflow Designer' page to delete the step

1. Click the **Design** link next to the workflow whose step you wish to delete. The **Workflow Designer** page is displayed, showing the steps that make up your workflow in the workflow design area (as shown under Using the 'Workflow Designer' above).
2. In the workflow design area, move the mouse pointer over the relevant step and click the cog icon that appears to the right of the step to reveal a popup menu.
   

3. From the popup menu, select **Delete Step** to remove the step from the workflow.

Using the 'View Workflow Steps' page to delete the step

1. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose step you wish to delete. The **View Workflow Steps** page is displayed, showing the steps that make up the workflow (as shown under Adding a step above).
2. Click the **Delete Step** link that corresponds to the relevant step.
   This link will only be shown if the step has no incoming transitions (unlike the equivalent **Delete Step** function in the **Workflow Designer**, which when used on a step with incoming transitions, will also remove all those incoming transitions).

Working with transitions

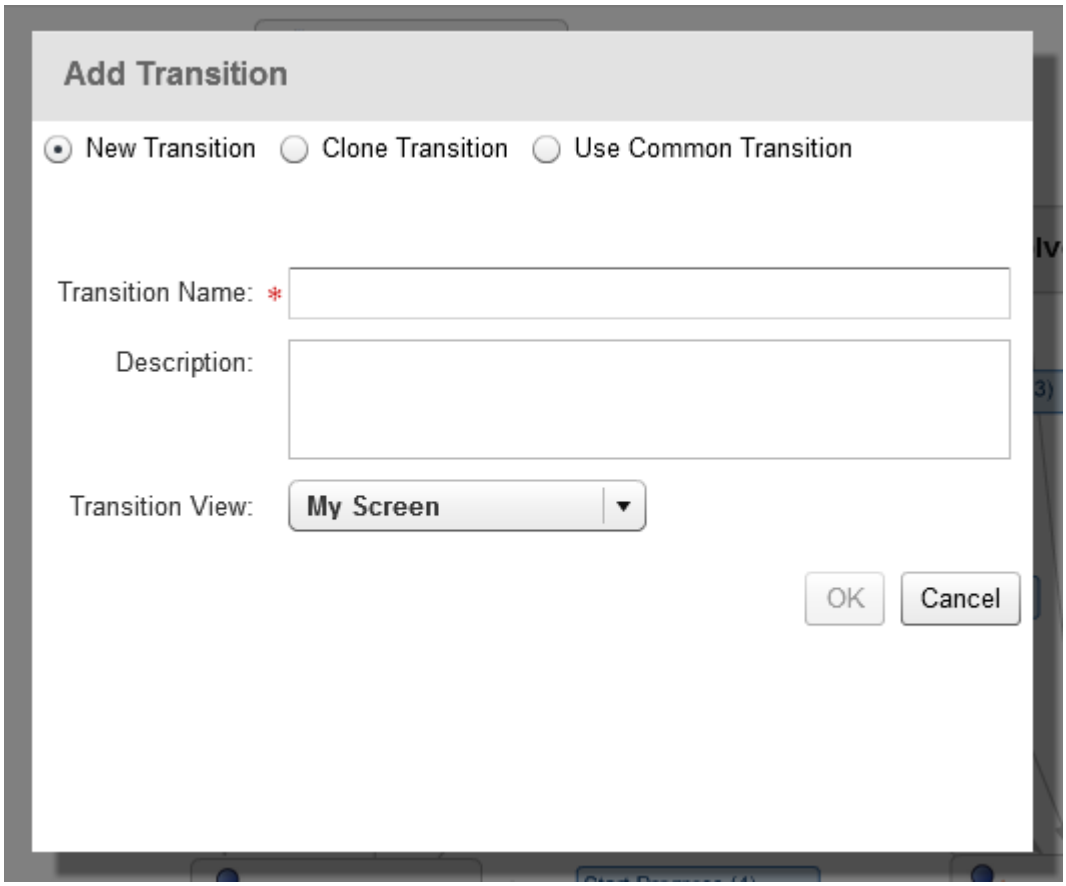
Adding a transition

To add a transition to a workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to add the transition to the workflow.
   The **Workflow Designer** provides more options for creating your new transition.

Using the 'Workflow Designer' page to add the transition

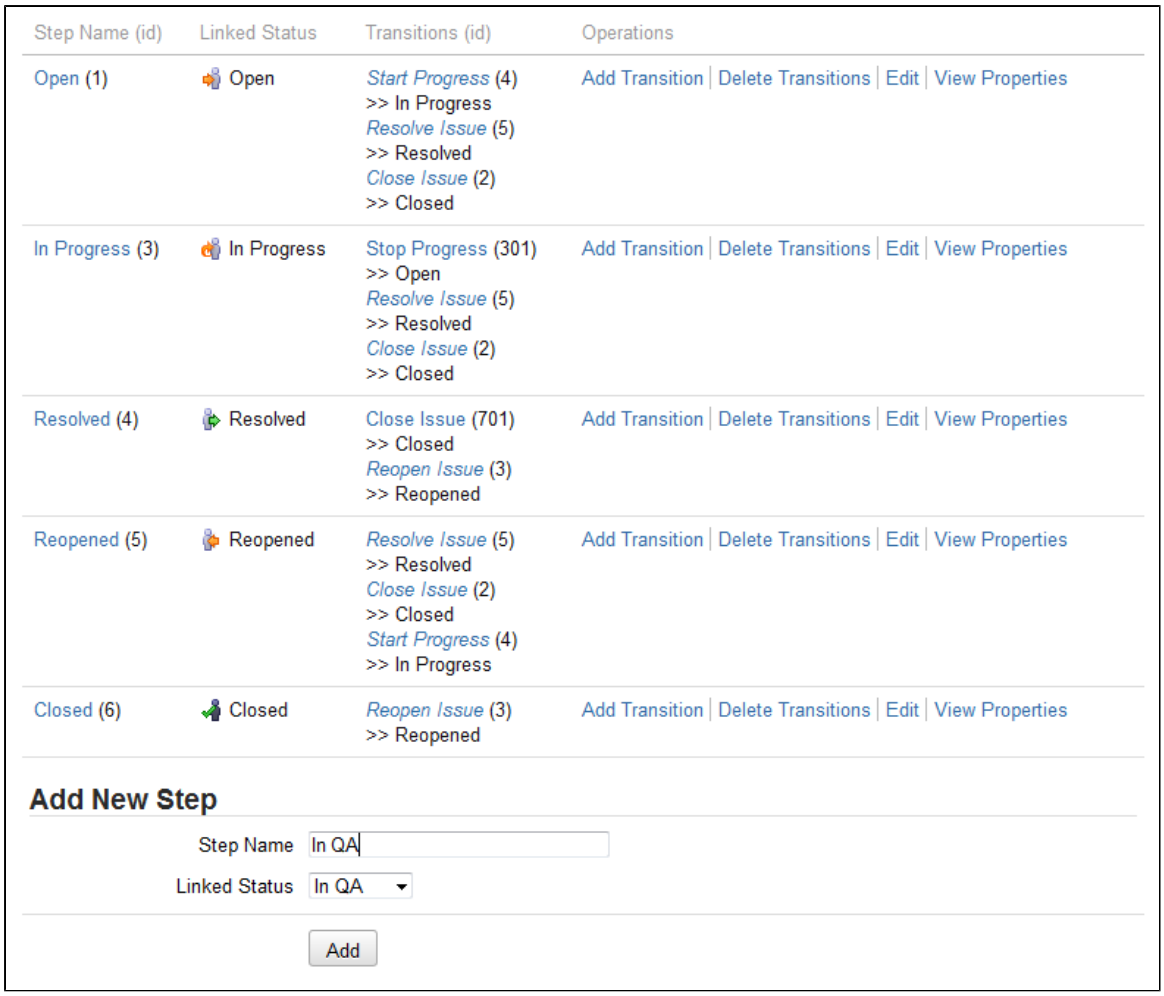
1. On the **View Workflows** page, click the **Design** link next to the workflow you wish to add a transition to. The **Workflow Designer** page is displayed, showing your workflow's existing steps as well as the lines and labels of transitions between these steps (as shown under Using the 'Workflow Designer' above).
2. Click the appropriate **Create Transition** button in the **Workflow Designer**'s main toolbar (above), according to the type of transition line you wish to create.
3. In the workflow design area, click the step that will be the **Originating Step** of the transition.
4. Click the step that will be the **Destination Step** of the transition. The **Add Transition** dialog box opens.



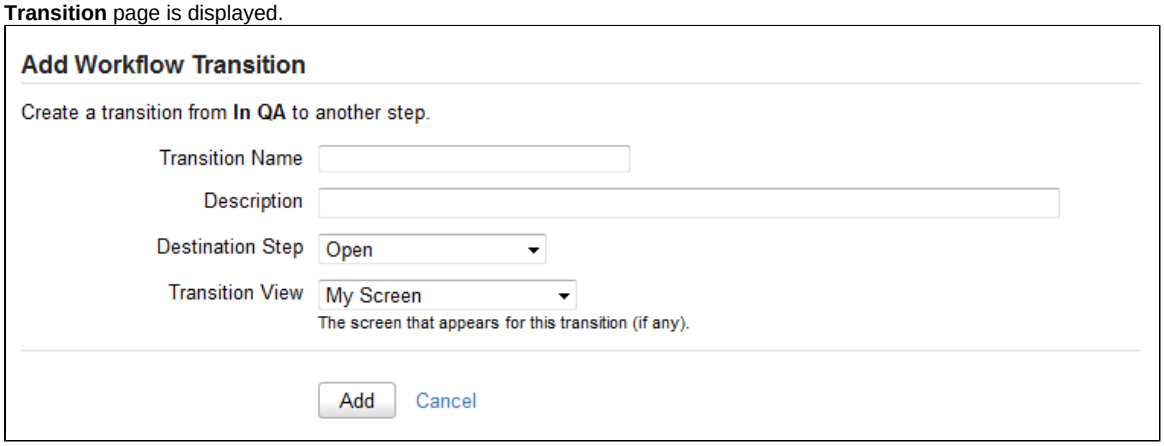
5. Choose between creating a new transition (**New Transition**), copying an existing transition (**Clone Transition**) or using a 'common transition' (**Use Common Transition**).
    - If you choose either **New Transition** or **Clone Transition**:
        a. In the **Transition Name** field, type a short name for the transition.
        This name will be shown to users on the relevant transition button, within the 'operations bar' of the 'view issue' page.
        b. *(Optional)* In the **Description** field, type a short description of the purpose of the transition.
        c. If you chose:
            - **New Transition**, then in the **Transition View** field, select either:
                - **No view for transition** — choose this if you do not need to prompt the user for input before the transition is executed (i.e. the transition will occur instantly when the user clicks the transition).
                - The name of a screen that will be shown to users, asking for input before the transition is executed. You can choose one of JIRA's default screens or any other screen you have created. If no existing screen is suitable, you may want to create a new screen.
                Many of JIRA's default screens are used in the default workflow (above) and named after the transitions they are used in (e.g. **Start Progress** and **Resolve Issue**).
            - **Clone Transition**, then in the **Transition To Clone** field, select the transition you wish to copy.
            Copying an existing transition creates a new independent copy of a transition, which does not result in a 'common transition'.
    - If you choose **Use Common Transition**, select an existing transition (currently used in your workflow which also leads to your **Destination Step**) from the **Transition To Reuse** dropdown list. See Using 'common transitions' (below) for details.
    Please Note:
        - Existing transitions in your workflow which do not lead to your **Destination Step** will not be available from the **Transition To Reuse** dropdown list.
        - You can select either an existing 'common transition' or an ordinary transition (i.e. a transition that has only been used once on the workflow). If you select an ordinary transition, it will be converted to a 'common transition'.
6. Click the **OK** button to complete the addition of your transition.

Using the 'View Workflow Steps' page to add the transition

1. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow you wish to add a transition to. The **View Workflow Steps** page is displayed, showing the existing steps that make up the workflow and each step's **Linked Status** and **Outgoing Transitions** (under **Transitions (id)**).



2. Identify the step from which your new transition will originate and click the **Add Transition** link next to the step. The **Add Workflow Transition** page is displayed.



3. In the **Transition Name** field, type a short name for the transition.
   This name will be shown to users on the relevant transition button, within the 'operations bar' of the 'view issue' page.
4. *(Optional)* In the **Description** field, type a short description of the purpose of the transition.
5. In the **Destination Step** field, choose the step to which issues will move when this transition is executed.
6. In the **Transition View** field, select either:
   - **No view for transition** — choose this if you do not need to prompt the user for input before the transition is executed (i.e. the transition will occur instantly when the user clicks the transition).
   - The name of a screen that will be shown to users, asking for input before the transition is executed. You can choose one of JIRA's default screens or any other screen you have created. If no existing screen is suitable, you may want to create a new screen.
     Many of JIRA's default screens are used in the default workflow (above) and named after the transitions they are used in (e.g. **Start Progress** and **Resolve Issue**).

Editing or deleting a transition

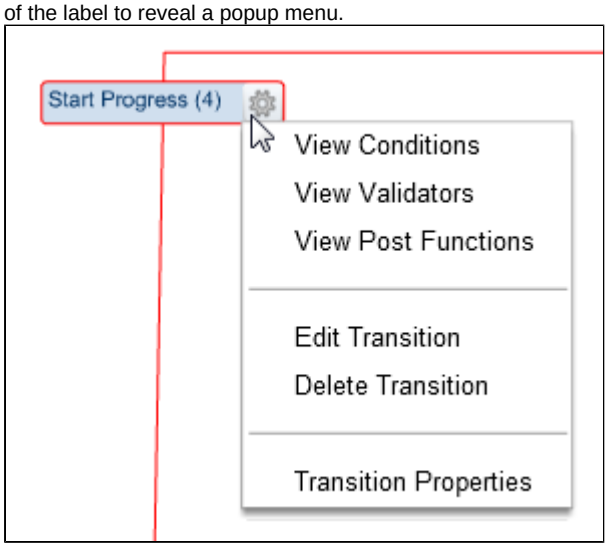
To edit or delete an existing transition of a workflow:

1. Log in as a user with the **JIRA Administrators** global permission.

2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to edit or delete the transition of the workflow.

Using the 'Workflow Designer' page to edit or delete the transition

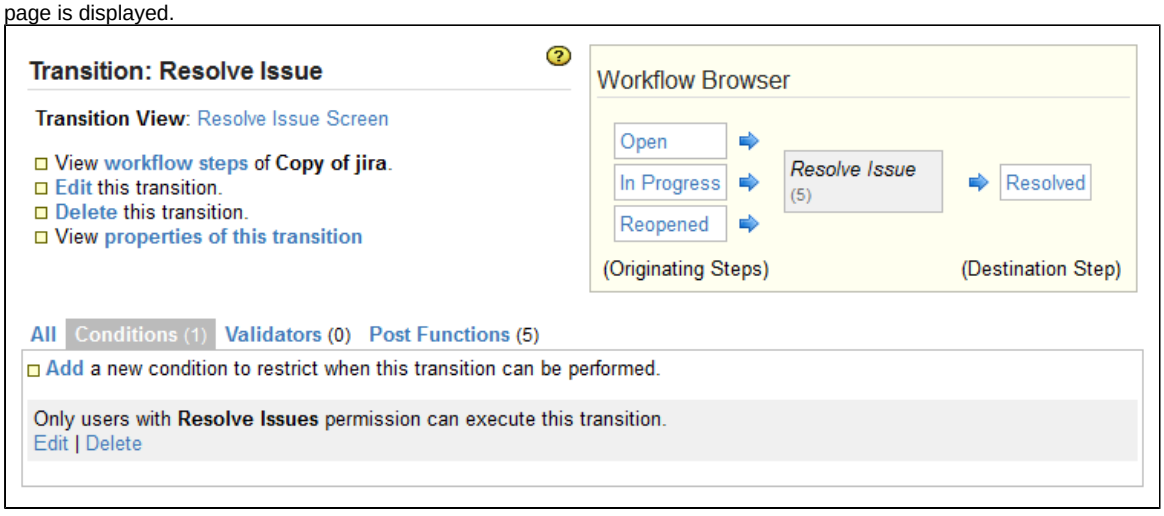
1. On the **View Workflows** page, click the **Design** link next to the workflow whose transition you wish to edit or delete. The **Workflow Designer** page is displayed, showing your workflow's existing steps as well as the lines and labels of transitions between these steps (as shown under Using the 'Workflow Designer' above).
2. In the workflow design area, move the mouse pointer over the relevant transition label and click the cog icon that appears to the right of the label to reveal a popup menu.



3. From the popup menu, you can:
   - Select **Edit Transition** to edit the **Transition Name**, **Description**, **Destination Step** and **Transition View** of the transition. See the **Add Workflow Transition** page (above) for details.
   - Select **Delete Transition** to do just that.
   - Select **Transition Properties** to edit the transition's **Properties**. See Working with transition properties (below) for details.

Using the 'View Workflow Steps' page to edit or delete the transition

1. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose transition you wish to edit or delete. The **View Workflow Steps** page is displayed, showing the existing steps that make up the workflow and each step's **Linked Status** and **Outgoing Transitions** in the **Transitions (id)** column (as shown under Adding a transition above).
2. In the **Transitions (id)** column, click the link of the **Outgoing Transition** of the step you wish to edit. The 'View Workflow Transition' page is displayed.



On the 'View Workflow Transition' page, the following information is shown about the transition:
   - On the left of the 'View Workflow Transition' page:
     - **Transition View** — the screen (usable globally) that your specific workflow transition uses. If your workflow transition does not require a view (i.e. no screen has been specified), then **None - it will happen instantly** is shown.
   - In the 'Workflow Browser' section:
     - **Originating Steps** — that is, steps whose **Outgoing Transition** is this transition.
       If the transition has more than one originating step, then it is either a global or common transition.
     - **Destination Step** — that is, the step whose **Incoming Transition** is this transition.
3. On the 'View Workflow Transition' page, you can:
   - Click '**Edit** this transition' to edit the **Transition Name**, **Description**, **Destination Step** and **Transition View** of the transition. See the **Add Workflow Transition** page (above) for details.
   - Click '**Delete** this transition' to do just that. This link is only available if the step has at least one outgoing transition

indicated in the **Workflow Browser** section.
- Click 'View **properties of this transition**' and edit the transition's **Properties**. See Working with transition properties (below) for details.

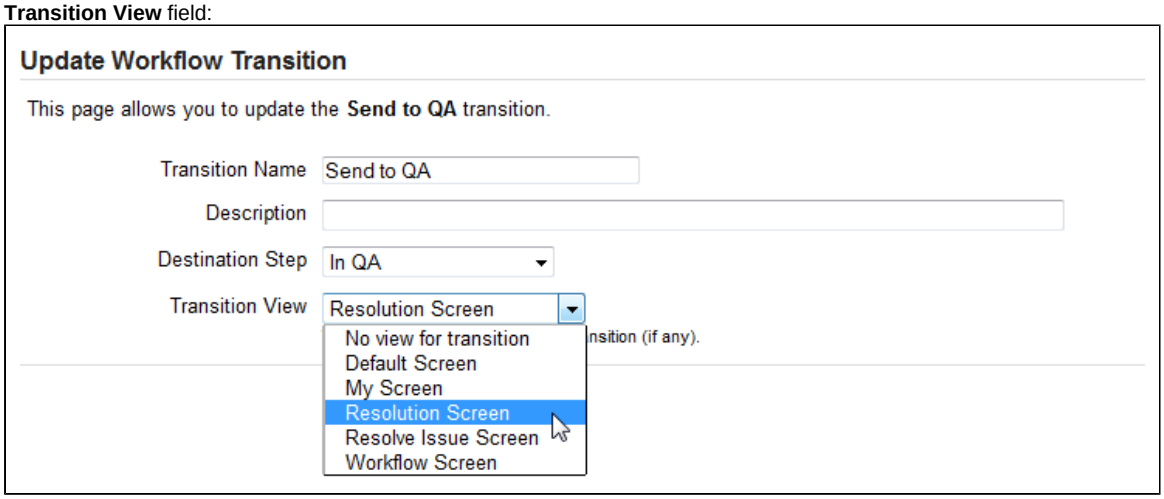
Using a screen with a transition

When a user clicks a particular transition, a screen can be used to gather input from the user before the transition is executed.

Example: using a screen to set the 'Resolution' field

For a particular step in a workflow, you might need to create a transition that will move the issue to a 'closed' status (e.g. **Closed**, **Resolved**, etc) - see 'open' and 'closed' issues. As part of this transition, you might need the user to set the **Resolution** field. To do this:

1. *(Optional)* Create a screen (e.g. named 'Resolution Screen'), that contains the **Resolution** field (and any other fields you want to show). See Defining a screen for details.
   ✅ JIRA already includes a screen named **Resolve Issue Screen**, which contains the **Resolution**, **Fix Version/s**, **Assignee** and **Log Work** fields.
2. Create or edit your transition and then choose the screen that contains the **Resolution** field (e.g. **Resolution Screen**) in the **Transition View** field:



Applying conditions to transitions

Conditions control who can perform a transition and the circumstances under which they can perform the transition (such as their project permissions, the status of any sub-tasks, or the state of source code associated with issues).

If any part of a transition's condition fails, the user will not see the transition link on the 'view issue' page.

JIRA's built-in conditions

JIRA includes the following individual conditions, which can be added immediately to any transition:

Condition	Description	Additional Parameters Required
Only Assignee Condition	Only allow the issue's current assignee to execute the transition.	None
Only Reporter Condition	Only allow the issue's reporter to execute the transition.	None
Permission Condition	Only allow users with a given permission to execute the transition.	A project-wide permission
Sub-Task Blocking Condition	Only allow a parent issue's transition to execute if all its sub-tasks have at least one of a specified set of statuses.	One or more statuses
User Is In Group	Only allow users in a given group to execute the transition.	A group

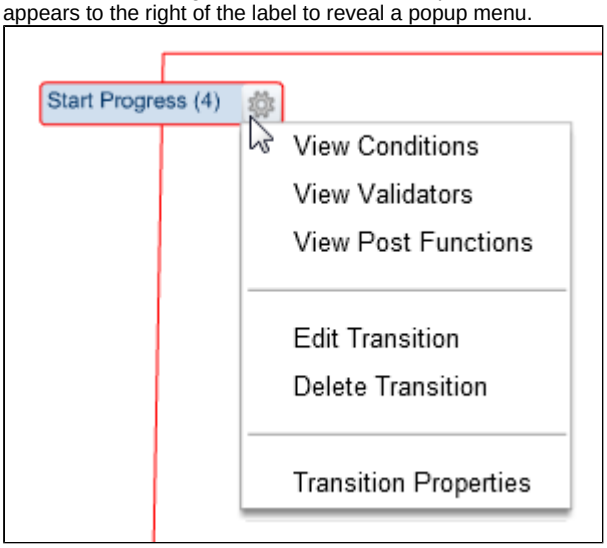
User Is In Group Custom Field	Only allow users in a given group-type custom field to execute a transition.	A custom field of type "Group"
User Is In Project Role	Only allow users in a given project role to execute a transition.	A project role
Code Committed Condition	Only allow the transition to execute if code either has or has not been committed against the issue.	Specify whether or not code must be committed
No Open Reviews Condition	Only allow the transition to execute if no related Crucible reviews are still open against the issue.	None
Unreviewed Code Condition	Only allow the transition to execute if no unreviewed changesets related to the issue exist.	None

You can also create your own conditions via the plugin system. See the Workflow Plugin Modules for details.

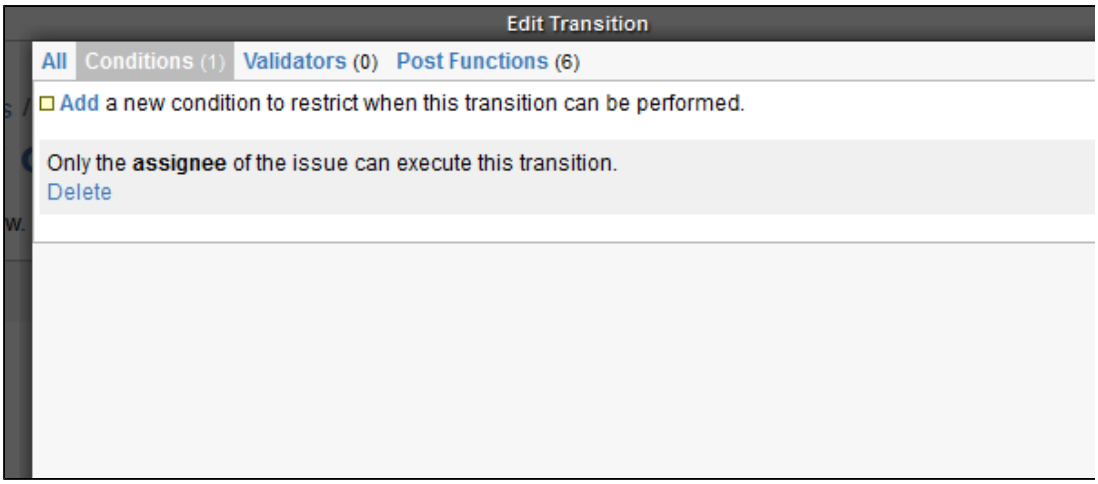
Adding a condition

To add a condition to a transition:

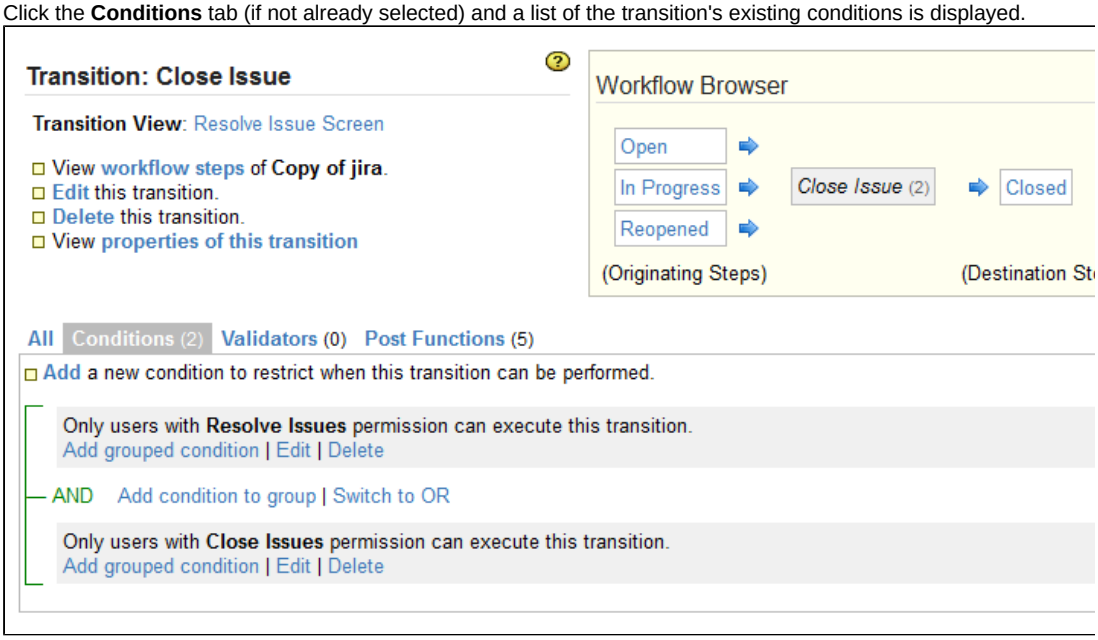
1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the 'View Workflows' page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to add the condition to the transition.
   - **Using the 'Workflow Designer' page to add the condition to the transition:**
     a. On the **View Workflows** page, click the **Design** link next to the workflow whose transition you wish to add a condition to. The **Workflow Designer** page is displayed.
     b. In the workflow design area, move the mouse pointer over the relevant transition label and click the cog icon that appears to the right of the label to reveal a popup menu.



     c. From the popup menu, select **View Conditions** to view a list of the transition's existing conditions. The **Edit Transition** dialog box's **Conditions** tab and a list of the transition's existing conditions is displayed.



- **Using the 'View Workflow Steps' page to add the condition to the transition:**
  a. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose transition you wish to add a condition to. The **View Workflow Steps** page is displayed.
  b. In the **Transitions (id)** column, click the name of the relevant transition. The 'View Workflow Transition' page is displayed.
  c. Click the **Conditions** tab (if not already selected) and a list of the transition's existing conditions is displayed.



4. Click the **Add** link (in the **Conditions** tab). A list of all available conditions is displayed. See above for a list of JIRA's built-in conditions.
5. Select a condition from the list and click the **Add** button.
   If the condition requires additional parameters (e.g. the name of a group or project role), the **Add Parameters To Condition** dialog box/page will be presented.
   - Specify your criteria on the **Add Parameters To Condition** dialog box/page and click the **Add** button. (Some criteria may require more than one step.)
6. The **Conditions** tab is displayed again, showing your new condition at the bottom of the list of conditions.
   From here, you can:
   - Click the **Edit** link next to the condition's name to edit its additional parameters (if applicable to the condition).
   - Click the **Delete** link next to the condition's name to remove the condition.
   - Combine your conditions into **AND**/**OR** groups. See Combining conditions into groups (below).

Combining conditions into groups

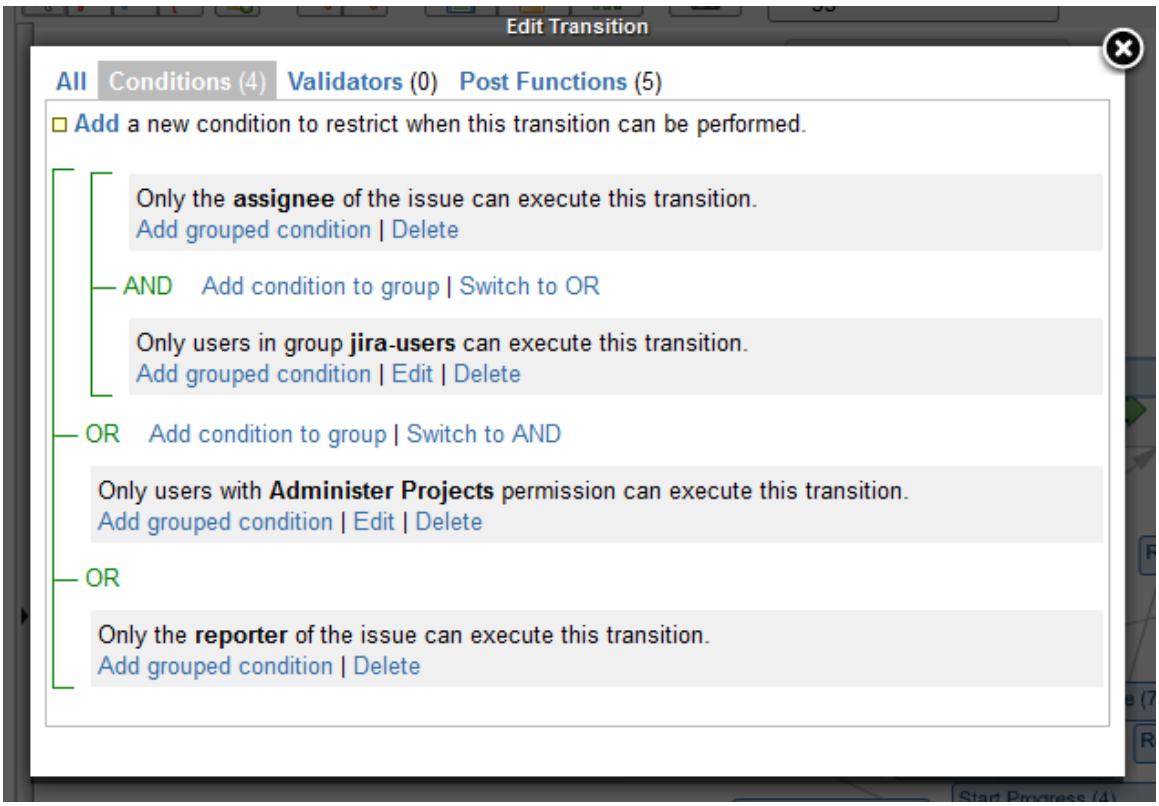
You can construct complex conditions by combining two or more 'individual conditions' (added using the procedure above) using **AND** or **OR** boolean logic to form a 'grouped condition'. For example, the following simple grouped condition could be constructed:

- Only the assignee of this issue can execute this transition
  AND
- Only users in group **jira-users** can execute this transition

This grouped condition will only be true (and hence, allow execution of the transition) if the user is the assignee of the issue **AND** the same user is in the **jira-users** group.

An overall 'complex condition' can be constructed for your transition by combining multiple grouped conditions using **AND** or **OR** boolean logic or even nesting grouped conditions. A transition with a complex condition (as such), can only be executed if all individual and grouped

conditions that constitute the complex condition, have been satisfied, e.g:



Converting an individual condition into a grouped condition

To convert an individual condition into a grouped condition:

1. Follow the Adding a condition procedure (above) to step 3.
2. Instead of clicking the **Add** link at step 4, click the **Add grouped condition** link of the individual condition you wish to convert into a grouped condition.
3. Continue on from step 5.

Please Note:

- You can use this procedure to create grouped conditions out of individual conditions which are already part of another grouped condition. This allows you to nest grouped conditions.
- If you only have a single individual condition on the 'Conditions' tab, the 'Add grouped condition' link will not be available on that condition. Hence, just click the **Add** button instead to create your grouped condition.

Please Also Note: It is not possible to add an individual condition that is separate from the outermost grouped condition. Clicking the **Add** button only adds an individual condition to the outermost grouped condition. Bear this in mind when creating a complex condition and refer to JRA-25179 for more information.

Adding individual conditions to a grouped condition

To add an individual condition to a grouped condition:

1. Follow the Adding a condition procedure (above) to step 3.
2. Instead of clicking **Add** link at step 4, click the **Add condition to group** link associated with the grouped condition you want to add your individual condition to.
3. Continue on from step 5.

Switching a grouped condition's logic

The logic of all individual conditions within a grouped condition can be switched between **AND** and **OR**. To do this, simply click the **Switch to OR** or **Switch to AND** link associated with the grouped condition whose logic you wish to switch.

If you need to switch the logic of only some (not all) individual conditions within a grouped condition, create a nested grouped condition as described above.

Applying validators to transitions

Validators check that any input available to the transition (such as user-supplied input) is valid before the transition is performed. For example, a validator can be used to ensure that the comment entered by a user on a transition's screen meets a certain project permission criterion.

If a transition's validator 'fails', the transition's post functions will not be executed and the issue will not progress to the destination step of the transition.

JIRA ships with a couple of default validators that validate whether or not the user who performed the transition had a particular project permission, which can be added immediately to any transition.

You can also create your own validators via the plugin system. See the Workflow Plugin Modules for details.

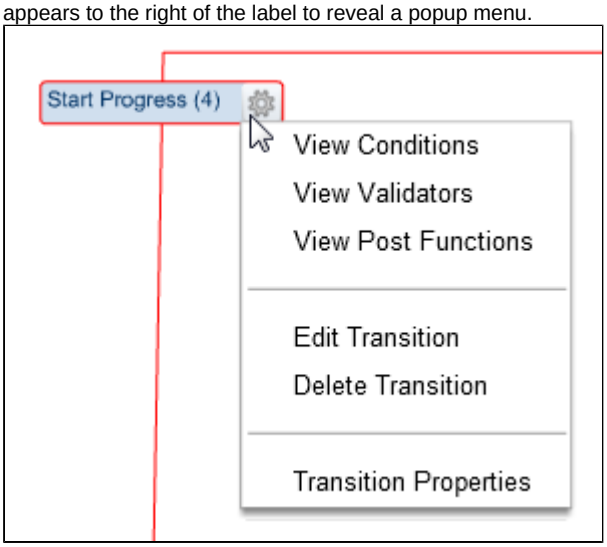
How do validators differ from conditions?

- Conditions are used to determine whether an issue's transition can be executed by a particular user, including a range of other circumstances such as the user's project permissions and the current state of the issue. However, conditions cannot validate input parameters provided by the user on the transition's screen, since if a condition fails, the user is prevented from executing the transition and cannot access the transition's screen.
- Validators have access to any input available to the transition (such as input gathered from the user on a transition's screen) and thus, can validate this input.

Adding a validator

To add a validator to a transition:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to add the validator to the transition.
   - **Using the 'Workflow Designer' page to add the validator to the transition:**
     a. On the **View Workflows** page, click the **Design** link next to the workflow whose transition you wish to add a validator to. The **Workflow Designer** page is displayed.
     b. In the workflow design area, move the mouse pointer over the relevant transition label and click the cog icon that appears to the right of the label to reveal a popup menu.



     c. From the popup menu, select **View Validators** to view a list of the transition's existing validators.
   - **Using the 'View Workflow Steps' page to add the validator to the transition:**
     a. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose transition you wish to add a validator to. The **View Workflow Steps** page is displayed.
     b. In the **Transitions (id)** column, click the name of the relevant transition. The 'View Workflow Transition' page is displayed.
     c. Click the **Validators** tab and a list of the transition's existing validators is displayed.
4. Click the **Add** link (in the **Validators** tab). A list of all available validators is displayed.
5. Select a validator from the list and click the **Add** button.
   If the validator requires additional parameters (e.g. the name of a group or project role), the **Add Parameters To Validator** dialog box/page will be presented.
   - Specify your criteria on the **Add Parameters To Validator** dialog box/page and click the **Add** button.
6. The **Validators** tab is displayed again, showing your new validator at the bottom of the list of validators.
   From here, you can:
   - Click the **Edit** link next to the validator's name to edit its additional parameters (if applicable to the validator).
   - Click the **Delete** link next to the validator's name to remove the validator.

Please Note:

- The logical relationship between multiple validators applied to a transition is **AND** only.
- Unlike Conditions, it is not possible to create 'grouped validators' or to change the logical relationship between multiple validators.

Applying post functions to transitions

Post functions carry out any additional processing required immediately after a transition is executed, such as updating an issue's fields, generating change history for an issue, adding a comment to an issue, or generating an event such as an email notification. There are two categories of post functions in JIRA — essential and optional.

Essential post functions

JIRA includes the following 'essential' post functions, which are automatically added to every newly-created transition and performed in this order:

Essential post function
Set issue status to the linked status of the destination workflow step.
Add a comment to an issue if one is entered during a transition.
Update change history for an issue and store the issue in the database.
Re-index an issue to keep indexes in sync with the database.
Fire an event that can be processed by the listeners.

These 'essential' post functions cannot be deleted from a transition or reordered relative to each other, as this could compromise other issue functionality within JIRA. However, you can insert other (optional) post functions around or between them.

Optional post functions

JIRA includes the following optional post functions which can be added to transitions:

Optional post function	Description
Assign to Current User	Assigns the issue to the user who is executing the transition. This post function will be ignored unless the user has the **Assignable User** permission. You may want to use a condition to ensure that the logged-in user has this permission before executing the transition.
Assign to Lead Developer	Assigns the issue to the component lead (if one exists) or project lead.
Assign to Reporter	Assigns the issue to the user who created the issue.
Create Perforce Job Function	Creates a Perforce Job (if required) after completing the workflow transition.
Update Issue Field	Updates one of the issue's fields to a given value. Fields which can be updated include: • Assignee • Description • Environment • Priority • Resolution • Summary • Original Estimate • Remaining Estimate Note that this post function cannot update custom fields.

You can also create your own post functions via the plugin system. See the Workflow Plugin Modules for details.

Note that the four optional post functions must be positioned *before* the **Update change history for an issue and store the issue in the database** post function (above), with the exception of the **Create** transition.

The initial transition ('Create' or 'Create Issue')

When creating an issue, it is sometimes useful to be able to perform specific processing tasks (such as setting a particular field's value). You can perform such tasks by adding post functions to the workflow's 'initial transition'.

The 'initial transition' is executed whenever a user creates an issue, which in turn, places the newly-created issue into the workflow's 'initial step'. Every workflow has only one 'initial step', which is the first step in the issue's workflow and is the 'initial step's' first incoming transition. By default:

- The 'initial transition' is called **Create** (if you created a 'blank' workflow) or **Create Issue** (if you copied the default workflow).
- The 'initial step' is called **Open** after having created a workflow above.

JIRA includes the following 'essential' post functions which are specific to a workflow's 'initial transition'. These post functions are automatically added to this transition of every newly-created workflow (and performed in this order):

Essential post function ('initial transition' only)
Creates the issue originally.
Fire an event that can be processed by the listeners.

The optional post functions (above) can also be added to a workflow's initial transition, as well as the following optional post functions (which, with the exception of **Store Issue**, are essential to all other newly-created transitions):

Optional post function ('initial transition' only)	Description
Create Comment	Adds a comment to an issue if one is entered during a transition.
Re-index Issue	Re-indexes an issue to keep JIRA's indexes in sync with the database.
Update Issue Status	Sets the issue's status to the linked status of the destination workflow step.
Store Issue	Stores updates to an issue (no change history is created).

Please be aware that a bug in JIRA (JRA-25070) currently prevents these optional post functions from being deleted once they have been added.

Optional post functions added to the workflow's **Create** transition (for example, an **Update Issue Field** post function to set the **Assignee** field to a particular user when an issue is created), must be placed *before* the **Creates the issue originally** post function.

Special case:

If you need to set the **Resolution** field when creating an issue, add the **Update Issue Field** post function *after* the **Creates the issue originally** post function and *after that*, use the **Store Issue** post function. The **Store Issue** post function is useful for setting the **Resolution** field during issue creation.

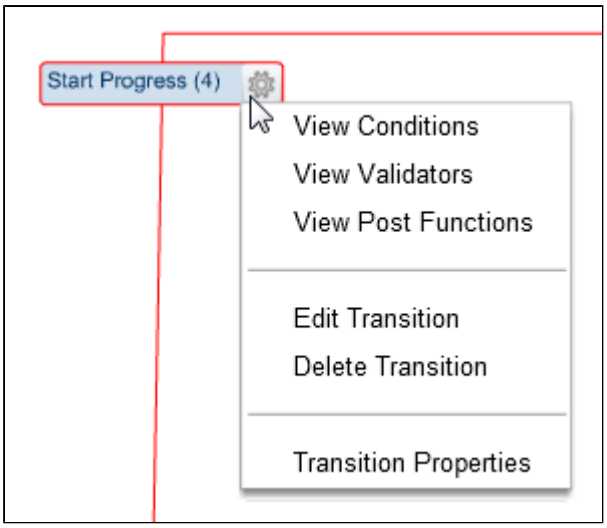
However, keep use of the **Store Issue** post function to a minimum, since this post function:

- Does not generate change history.
- Is incapable of persisting fields that have a one-to-many relationship with the issue (e.g. **Version** or **Component**).

Adding a post function

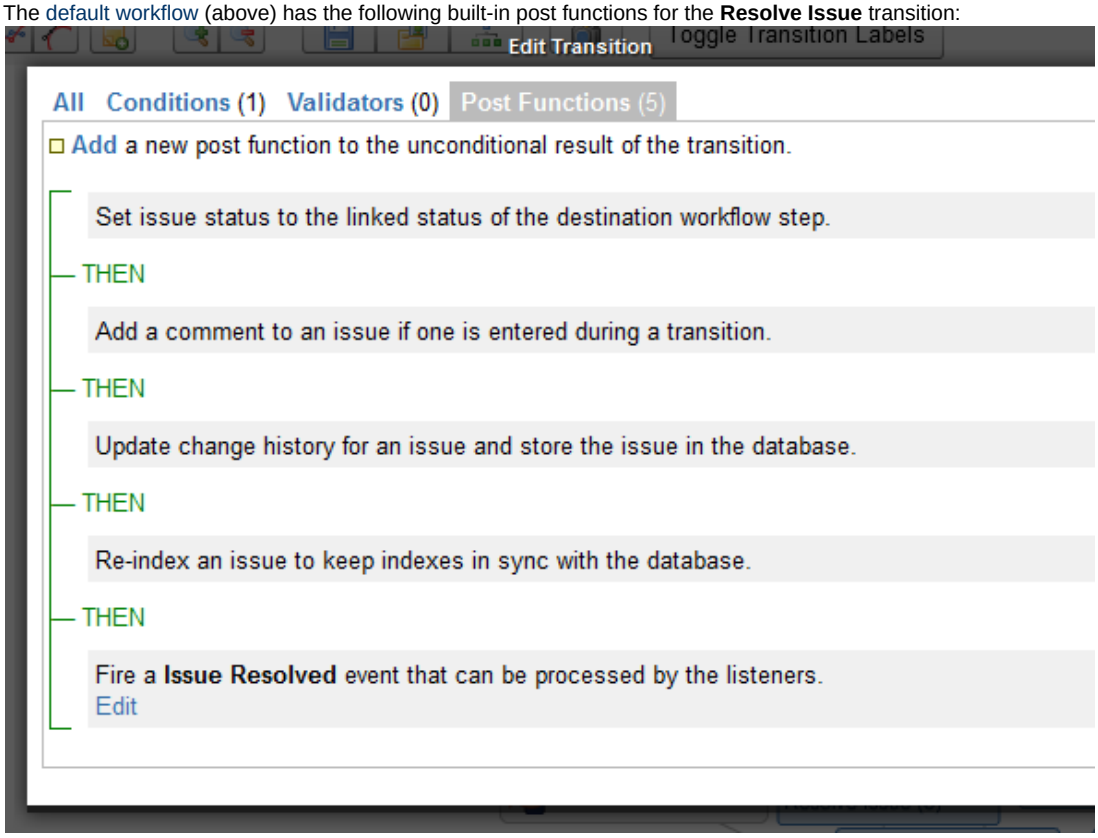
To add a post function to a transition:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to add the post function to the transition.
   - **Using the 'Workflow Designer' page to add the post function to the transition:**
     a. On the **View Workflows** page, click the **Design** link next to the workflow whose transition you wish to add a post function to. The **Workflow Designer** page is displayed.
     b. In the workflow design area, move the mouse pointer over the relevant transition label and click the cog icon that appears to the right of the label to reveal a popup menu.



c. From the popup menu, select **View Post Functions** to view a list of the transition's existing post functions.

- **Using the 'View Workflow Steps' page to add the post function to the transition:**
  a. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose transition you wish to add a post function to. The **View Workflow Steps** page is displayed.
  b. In the **Transitions (id)** column, click the name of the relevant transition. The **View Workflow Transition** page is displayed.
  c. Click the **Post Functions** tab and a list of the transition's existing post functions is displayed.
  The default workflow (above) has the following built-in post functions for the **Resolve Issue** transition:



4. Click the **Add** link (in the **Post Functions** tab). A list of all available post functions is displayed.
5. Select a post function from the list and click the **Add** button.
6. If the post function requires one or more configuration parameters (e.g. the name of an event), the **Add Parameters To Function** dialog box/page will be presented. Enter the appropriate information and click the **Add** button.
7. The **Post Functions** tab is displayed again, showing your new post function at the bottom of the list of post functions.

From here, you can:

- Click the **Edit** link next to the post function's name to edit its configuration parameters (if there are any).
- Click the **Move Up** link to move the post function higher up in the list (i.e. it will be executed earlier).
- Click the **Move Down** link to move the post function lower down in the list (i.e. it will be executed later).
- Click the **Delete** link next to the post function's name to remove the post function.

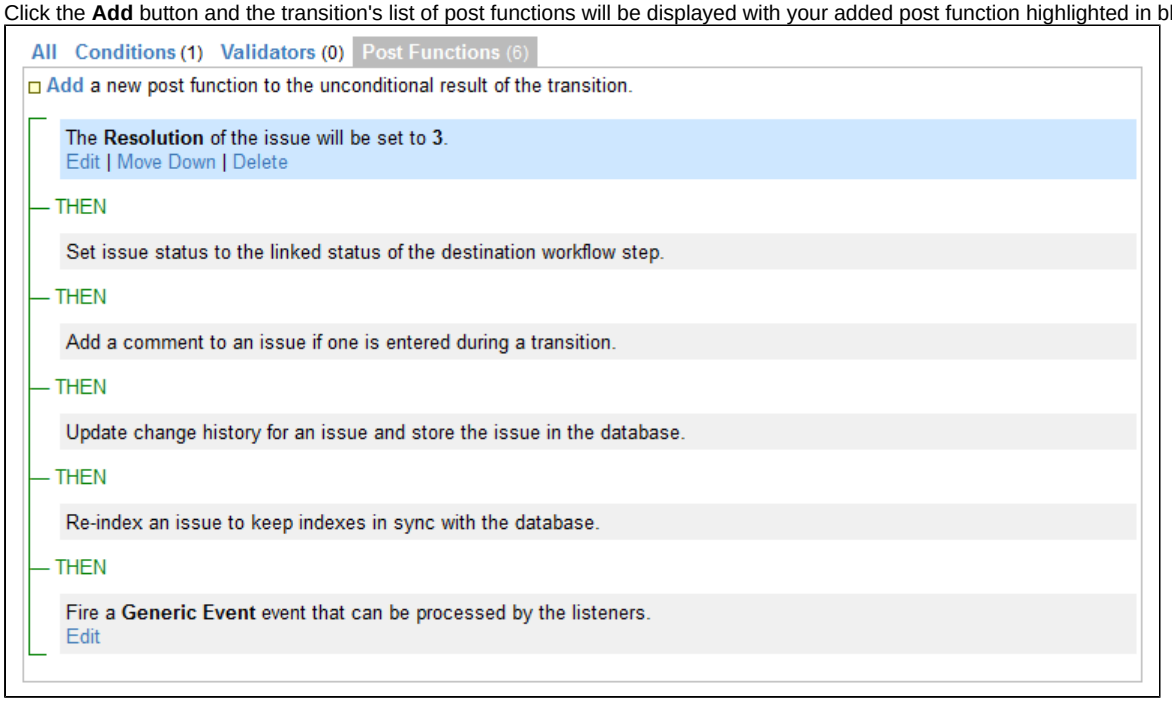
Using a post function to set a field

You can use a post function of type **Update Issue Field** to set the value of an issue's field(s) after a particular transition is executed.

Example: Using a post function to set the 'Resolution' field

For a particular step in a workflow, you might need to create a transition that will move the issue to a 'closed' status (e.g. **Close**, **Resolved**, etc) - see 'open' and 'closed' issues. As part of this transition, you might want to automatically set the **'Resolution'** field. To do this:

1. Create or edit your transition (above).
    - In the **Transition View** field on the **Add Transition** dialog box or the **Add/Update Workflow Transition** page (above), select either **No View For Transition** or the screen that does not contain the **Resolution** field (such as the **Add Comment And Assign** screen).
2. Add a new post function of type **Update Issue Field** (above).
3. Do the following on the **Add Parameters to Function** page:
    a. Select **Resolution** from the **Issue Field** select list.
    b. Select a suitable resolution from the **Field Value** select list.
4. Click the **Add** button and the transition's list of post functions will be displayed with your added post function highlighted in blue:



To create a transition that clears the **Resolution** field, follow the same steps above for adding an **Update Issue Field** post function to your transition. However, on the **Add Parameters To Function** page, select **None** from the **Field Value** select list.
The list of post functions for this transition will include the following statement:

- The **Resolution** of the issue will be **cleared**.

Each time one of these transitions is executed, the **Resolution** of the issue is automatically set or cleared as specified in these post functions.

Using a post function to send a notification

You can use the **Fire an event that can be processed by the listeners** post function to fire the **Generic Event**, which is a built-in JIRA event that can be used to trigger the sending of email notifications after a particular transition is executed.

Alternatively, you could fire a custom event that you have created specifically for this transition.

When a transition is performed, JIRA will:

- Look up the notification scheme associated with the issue's project and identify the users associated with the fired event;
- Send an email notification to each user.
  The fired event is also propagated to all registered listeners.

Example: Using a post function to fire the 'Generic Event'

You can use the **Generic Event** to send email notifications. To do this:

1. Create or edit your transition (above).
2. Click the transition's **Post Functions** tab and edit the **Fire an event that can be processed by the listeners** post function.
3. On the **Update parameters of the Fire Event Function for this transition** page, select **Generic Event** from the list of events.

Working with transition properties

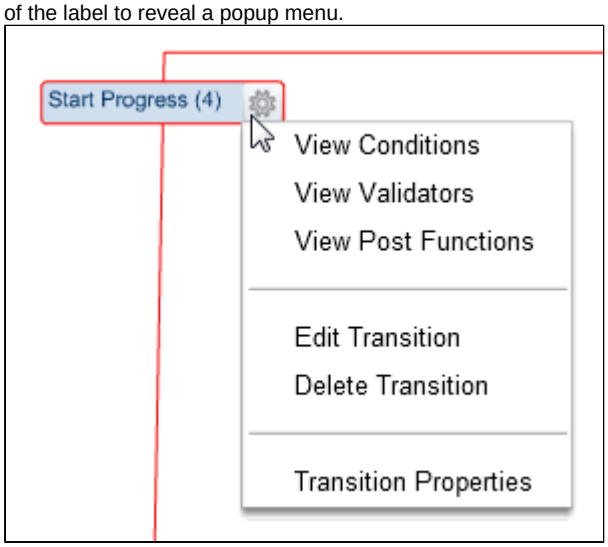
Properties are key-value pairs that are can be used to further customise transitions. For example, transition properties help to extend the default workflow (above) to allow language translations.

To view and edit the properties of a transition:

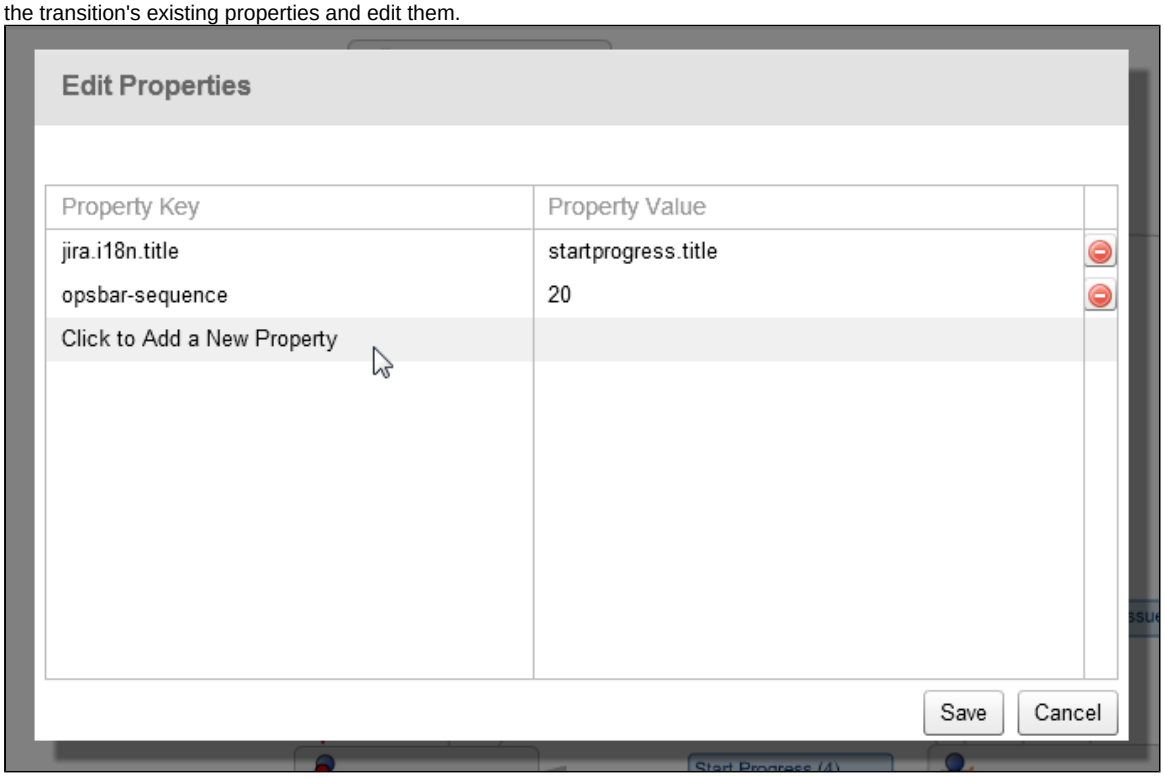
1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the 'View Workflows' page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Use either the **Workflow Designer** or the **View Workflow Steps** pages to view and edit the properties of a transition.

Using the 'Workflow Designer' page to view and edit the properties of a transition

1. On the **View Workflows** page, click the **Design** link next to the workflow whose transition properties you wish to view or edit. The **Workflow Designer** page is displayed.
2. In the workflow design area, move the mouse pointer over the relevant transition label and click the cog icon that appears to the right of the label to reveal a popup menu.



3. From the popup menu, select **Transition Properties**, which opens the **Edit Properties** dialog box that allows you to view a list of the transition's existing properties and edit them.



On the **Edit Properties** dialog box, you can:

- Add a new property to the transition.
- Edit a property's key or value, by simply clicking the property's key or value to begin editing it.
- Delete a property, by clicking the icon to the right of the property.

Using the 'View Workflow Steps' page to view and edit the properties of a transition

1. On the **View Workflows** page, click the hyperlinked number (under the **Steps** column) next to the workflow whose transition properties you wish to view or edit. The **View Workflow Steps** page is displayed.
2. In the **Transitions (id)** column, click the name of the relevant transition. The 'View Workflow Transition' page is displayed.
3. Click the 'View **properties of this transition**' link. The **View Workflow Transition Properties** page is displayed listing the properties currently set up for the transition. You can this page to add and delete properties for this transition.

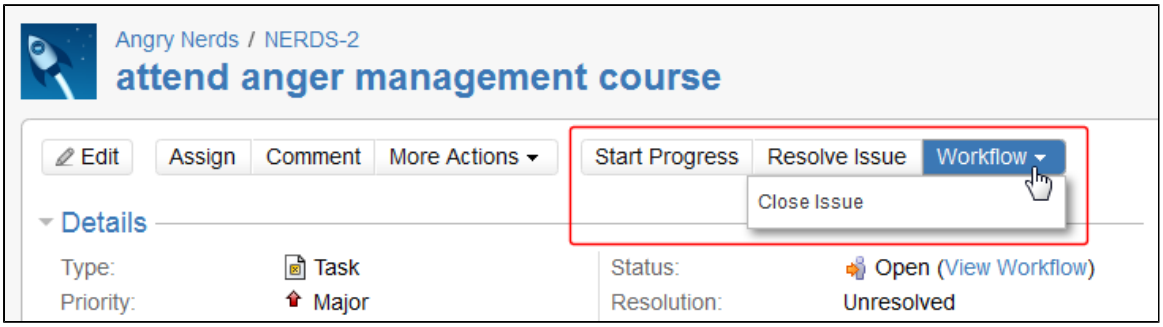
It is not possible to edit a transition's properties on this page. To change any property's key or value (or both), you must first delete the property you wish to change and add the new updated property.

For more information on specific transition properties, refer to the Workflow Properties page.

Customising workflow transitions on the 'view issue' page

When viewing an issue, most of the operations and workflow transitions accessible to a user are available from a row of buttons towards the top of the issue, known as the 'Operations Bar'. As shown in the following screenshot, workflow transitions appear in the right-most set of buttons of the operations bar.

Screenshot: Workflow transitions on the 'view issue' page



By default, the first two transitions appear as separate buttons in the set of transition buttons. Any additional transitions 'spill over' into the **Workflow** button dropdown menu. The order in which these buttons appear on the view issue page is based on the order of the default workflow (above), or for custom workflows, the order in which a JIRA administrator has added transitions to the custom workflow.

Hence, in the example above, the workflow transition order is: **Start Progress** -> **Resolve Issue** -> **Close Issue**.

JIRA provides the ability to customise the appearance and order of these transitions on the View Issue page.

Changing the number of transition buttons

To change the number of transition buttons from the default of two (with any remaining transitions spilling over into the 'Workflow' button dropdown menu):

1. Shutdown JIRA.
2. Edit the `jira-config.properties` file in your JIRA Home Directory.
   See Making changes to the `jira-config.properties` file for more information.
3. Change the value of 'X' in the `ops.bar.group.size.opsbar-transitions = X` property of this file to *the number of transition buttons required before the **Workflow** button*.
   If this property does not exist in your `jira-config.properties` file, add it. Otherwise, a default value of 2 is assumed.
4. Save the updated `jira-config.properties` file.
5. Restart JIRA.

Changing the order of transition buttons and 'workflow' menu items

To change the order of transition buttons, including additional transitions in the 'workflow' dropdown menu on the 'view issue' page, you need to add the property key `opsbar-sequence` to each workflow transition that you wish to re-order. Each `opsbar-sequence` property key requires a property value that defines the order of the transition action on issue views.

To add an `opsbar-sequence` property key and value to a workflow transition:

1. Access the workflow transition's properties, as described in Working with transition properties (above).
2. In the **Add New Property** section of the workflow transition's **View Workflow Transition Properties** page, type `opsbar-sequence` into the **Property Key** field.
3. In the **Property Value** field, type a positive integer value (starting at '0') that defines the order of the transition action on issue views.
   For each workflow transition, you may wish to use gaps in your `opsbar-sequence` property values (for example, 10, 20, 30, etc.) rather than consecutive values. This will allow you to 'insert' new workflow transitions more easily at a later point in time.
4. Click the **Add** button.

Please Note: Be aware that adding the `opsbar-sequence` property to a workflow transition does not change the order of these transitions on the **View Workflow Steps** page (above). The addition of this property only affects the order of transitions on the 'view issue' page.

Using 'common transitions'

A 'common transition' is a transition that is defined only once in a given workflow but is used more than once in the same workflow. A common transition has more than one originating step leading to a single destination step, but provides the added advantage of only

requiring you to edit or update the transition in one place.

Please Note: Global transitions (above) are similar to common transitions and provide the same editing advantages. However, they differ from each other in the following respects:

- For common transitions, the destination step of the transition has only a subset of the remaining steps in the workflow as incoming steps.
- For global transitions, the destination step of the transition has all remaining steps in the workflow as incoming steps. Be aware that global transitions can only be deleted through the **Workflow Designer** (above).

You can create common transitions using the 'Workflow Designer' by following these steps:

1. Create a transition to a destination step as described in Adding a transition (above).
2. Create another transition from another step to the same destination step, but in the **Add Transition** dialog box, choose the **Use Common Transition** option.
3. In the **Transition To Reuse** dropdown menu, select the transition you created in step 1 above.

You can edit existing common transitions via the **View Workflow Steps** page, but as described in Adding a transition (above), they cannot be created.

If you are only able to use the **View Workflow Steps** page to create or edit workflows, you can do either of the following to edit common transitions in a JIRA workflow:

- **Copy the default workflow** — the default workflow (above) contains common transitions (e.g. **Start Progress**, **Resolve Issue**, **Close Issue**). Although you cannot edit the default workflow, you can copy it and then edit its steps and transitions (via XML) to suit your requirements.
- **Create your workflow in XML** — see Using XML to create a workflow (below).

Using XML to create a workflow

JIRA's workflow editor generates OSWorkflow XML definition files that are stored in JIRA's database. If you need to take advantage of an OSWorkflow-based feature that is not available in JIRA's workflow editor (such as 'common transitions' above), you can define the workflow in XML and then import it into JIRA as described below.

Once the XML workflow has been imported, JIRA's workflow editor should be able to display most OSWorkflow definitions even if it does not support creating or editing them.

For example, conditional results of workflow transitions are displayed in the **Other** tab on the 'View Workflow Transition' page. The **Other** tab is only visible if a transition has elements that the editor does not directly support.

Importing an XML workflow into JIRA

To import an XML workflow into JIRA:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **Administration > Issues > Workflows** to open the **View Workflows** page, which shows a list of all existing workflows in your system as shown under Creating a workflow (above).
   Keyboard shortcut: **g + g +** start typing **workflows**
3. Click the **Import from XML** button to open the **Import Workflow** dialog box.
4. In the **Name** field, type a name (usually 2-3 words) to identify your new workflow.
5. *(Optional)* In the **Description** field, type a detailed description of your new workflow.
6. For the **Workflow Definition** option, you can do either of the following:
   - **Upload an XML workflow definition file** — to do this, choose the **Provide a full path to an XML file...** option and in the **File Path** field, type the full path to your XML workflow definition file.
     This path must be local one, so your XML workflow definition file must be located on your JIRA server.
   - **Paste the contents of an XML workflow definition file into JIRA** — to do this, choose the **Paste the workflow XML definition** option, copy the contents of your XML workflow definition file and in the **Workflow Definition (XML)** field, paste this copied content.
7. Click the **Import** button.

Copying a workflow between systems

Sometimes it is useful to create a workflow in a test system and then copy it into a production system. To do this:

1. In the test system, export the workflow to XML by clicking the **XML** link next to the workflow in the list shown on the **View Workflows** page (above) and save the output into a file.
2. In the production system, import the file via the 'import a workflow from XML' link as described in Using XML to create a workflow (above).

When importing an XML workflow into JIRA:

- JIRA's XML workflow definitions contain references to JIRA meta attributes. For example, the id of the linked JIRA status of each workflow step is stored as a 'jira.status.id' meta attribute in the step's definition. Therefore, when manually creating workflows in XML, please ensure that all referenced external entities exist before you import the workflow into JIRA.

When copying a workflow between systems:

- Please note that conditions, validators and post functions can have parameters that might be valid in one system and not in another. For example, different systems might contain different sets of values for the 'Resolution' field (since it is possible to define your own values). This would be a problem if the 'Update Issue Field' post function is used to set the 'Resolution' field to a value that exists in one system but not the other.

Activating Workflow

How is workflow 'activated'?

Once you have created a new workflow or modified an inactive workflow, you will need to 'activate' it. To activate a workflow, you need to:

1. Create a workflow scheme that references your workflow and optionally associate it with the relevant issue type(s).
2. Associate the workflow scheme with the relevant project(s).

Some terminology:

- '**Active**' workflows are those that are currently being used.
- '**Inactive**' workflows are those that are not associated with any workflow schemes, or are associated with workflow schemes that are not associated with any projects.

If you edit an active workflow, it does not need to be re-activated after your changes. Read more about editing active workflows.

What is a 'workflow scheme'?

A 'workflow scheme' defines a set of associations (or 'mappings') between any workflow and any issue type. A workflow scheme is associated with a project, which thereby defines what issue types available in the project use what workflows. Workflow schemes make it possible to use a different workflow for every combination of project and issue type.

On this page:

- How is workflow 'activated'?
  - What is a 'workflow scheme'?
- Creating a workflow scheme
- Associating a workflow scheme with a project
- Editing a Workflow Scheme
- Disassociating a Workflow Scheme from a Project

Creating a workflow scheme

To create a workflow scheme:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Workflows > Workflow Schemes** (tab) to open the **Workflow Schemes** page, which shows a list of all existing workflow schemes in your system.
   Keyboard shortcut: **g + g +** start typing **workflow schemes**
3. Click the **Add Workflow Scheme** link. The **Add Workflow Scheme** page is displayed.
4. Type a **Name** and (optionally) a short **Description** for the new workflow scheme.
5. Click the **Add** button. The **Edit Workflows for *<name of your workflow scheme>*** page is displayed, showing your newly-created scheme.
6. Click the **Assign a workflow** link. The **Add Workflow To Scheme** page is displayed.
   a. In the **Issue Type** dropdown list, select an issue type that is relevant to your workflow.
      You can also select **All Unassigned Issue Types** to associate your workflow with all issue types that do not have a specific association in this workflow scheme.
   b. In the **Workflow** dropdown list, select the name of your new workflow.
   c. Click the **Add** button.
7. Repeat the previous step until your new workflow has been associated with all the relevant issue types.
   You can choose different workflows for some issue types if you wish.

Once a workflow scheme is fully defined, you need to associate it with one or more projects (see below) so that the scheme's workflows are actually used by your JIRA system.

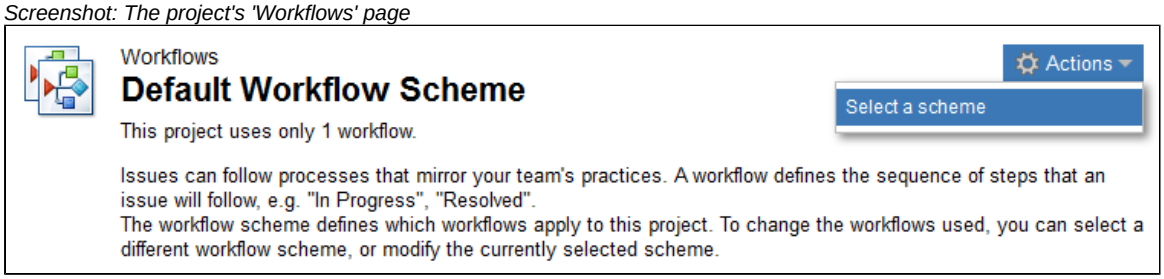
Associating a workflow scheme with a project

To associate a workflow scheme with a project:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Projects** and from the dropdown menu, select the name of the project of interest. The **Project Summary** page is displayed.

Keyboard shortcut: **g + g +** start typing **projects**

3. On the left of the **Project Summary** page, click the **Workflows** tab (or in the **Workflows** section, click either the name of the current workflow scheme or the **More** link). The **Workflows** page is displayed, indicating the current workflow scheme used by the project.
Screenshot: The project's 'Workflows' page



4. Click the **Actions** dropdown menu and choose **Select a scheme** (or **Use a different scheme**). The **Associate Workflow Scheme to Project** page is displayed.
5. Select the relevant workflow scheme from the **Scheme** list and click the **Associate** button.
6. Follow the wizard, which will guide you through migrating all the project's issues to the new scheme's workflows.

Please Note:

- You can associate a single workflow scheme with more than one project.
- Be aware that the issue type scheme associated with a project defines what issue types are available to that project. Hence, if an issue type (defined in the same project's workflow scheme) is not defined in the project's issue type scheme, then that issue type's workflow will not be used. Of course, this situation will change if that issue type is subsequently added to the project's issue type scheme.

Editing a Workflow Scheme

To edit a workflow scheme, i.e. to change which workflows are associated with which issue types:

1. Select **Administration > Issues > Workflows > Workflow Schemes** (tab) to open the **Workflow Schemes** page, which shows a list of all existing workflow schemes in your system.
Keyboard shortcut: **g + g +** start typing **workflow schemes**
2. Either click the name of the workflow scheme of interest or under the **Operations** column, click the **Workflows** link associated with the workflow scheme of interest. The **Edit Workflows** page is displayed.
3. On the **Edit Workflows** page, you can:
    - Associate a workflow with an issue type by clicking the '**Assign a workflow** to an issue type for this scheme' link.
        - Upon doing so, the **Add Workflow to Scheme** page is displayed. From this page, select the **Issue Type** and **Workflow** that you want to associate together and click the **Add** button.
    - Dissociate a workflow from an issue type by clicking the **Delete** link associated with the issue type and its workflow.

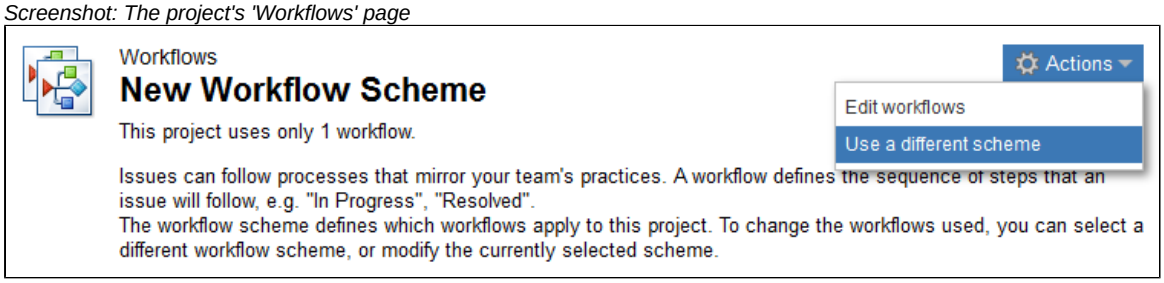
It is not possible to edit an active workflow scheme. An active workflow scheme is one that is currently associated with one or more projects. Instead:

- Copy the active workflow.
- Edit this (inactive) copy of the workflow.
- Associate all the relevant projects with the new copy.

Disassociating a Workflow Scheme from a Project

To disassociate a workflow scheme from a project, i.e. associate the project with JIRA's default workflow scheme:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Projects** and from the dropdown menu, select the name of the project of interest. The **Project Summary** page is displayed.
Keyboard shortcut: **g + g +** start typing **projects**
3. On the left of the **Project Summary** page, click the **Workflows** tab (or in the **Workflows** section, click either the name of the current workflow scheme or the **More** link). The **Workflows** page is displayed, indicating the current workflow scheme used by the project.
Screenshot: The project's 'Workflows' page



4. Click the **Actions** dropdown menu and choose **Use a different scheme**. The **Associate Workflow Scheme to Project** page is displayed.
5. Select the **Default** workflow scheme from the **Scheme** list and click the **Associate** button.
6. Follow the wizard, which will guide you through migrating all of the project's issues to the default workflow.

All projects that do not have an associated workflow scheme use JIRA's default workflow.

Adding a Custom Event

Overview of JIRA Events

JIRA uses an event-listener mechanism to alert the system that something has happened, and to peform appropriate action (e.g. send an email notification) based on the event that has occurred. Every *issue operation* within JIRA is associated with a particular *event* - e.g. the `Issue Created` event is fired when an issue has been created.

A Listener can execute a specified action once it has been notified that a particular event has been fired. For example, the MailListener can send an `Issue Created` email to a list of recipients defined in the appropriate Notification Scheme, whenever an issue is created.

Some events are fired by JIRA internally — e.g. an `Issue Updated` or `Issue Moved` event. Other events are fired from within workflow transition post-functions — e.g. an `Issue Resolved` event, or a Custom Event (see below).

Event Types

There are two types of events within JIRA:

- **System** — System events are used throughout JIRA internally, and cannot be added or deleted. You can, however, make them Inactive (see below).
- **Custom** — Custom events are used to generate an email notification (or invoke a listener) from a particular workflow transition's post-function. You can add/delete as many custom events as you need. Note that only *inactive* custom events can be deleted.

An event can be in either of the following states:

- **Active** — the event is associated with at least one notification scheme or workflow transition post-function.
- **Inactive** — the event is not associated with any notification schemes or workflow transition post-functions.
  Note that the event state does not indicate whether the event is able to be fired. A **custom event** will only be fired if it is associated with a transition post-function for an active workflow (see Activating Workflow).

On this page:

- Overview of JIRA Events
  - Event Types
    - System Events
    - Custom Events
- Configuring Notifications for a Custom Event
  - Step 1. Add a Custom Event
  - Step 2. Configure Notification Scheme to send mail on Custom Event
  - Step 3. Configure Workflow Transition Post-Function to Fire Custom Event

System Events

JIRA's built-in system events are:

Issue Created:	An issue has been entered into the system.
Issue Updated:	An issue has had its details changed.
Issue Assigned:	An issue has been assigned to a new user.
Issue Resolved:	An issue has been resolved (usually after being worked on and fixed).
Issue Closed:	An issue has been closed. (Note that an issue may be closed without being resolved; see Statuses).
Issue Commented:	An issue has had a comment added to it.
Issue Comment Edited:	An issue's comment has been modified.

Issue Reopened:	An issue has been re-opened.
Issue Deleted:	An issue has been deleted.
Issue Moved:	An issue has been moved into this project.
Work Logged On Issue:	An issue has had hours logged against it (i.e. a worklog has been added).
Work Started On Issue:	The Assignee has started working on an issue.
Work Stopped On Issue:	The Assignee has stopped working on an issue.
Issue Worklog Updated:	An entry in an issue's worklog has been modified.
Issue Worklog Deleted:	An entry in an issue's worklog has been deleted.
Generic Event:	The exact nature of this event depends on the workflow transition post-function(s) which invoke it. As with Custom Events, you can use the Generic Event to generate an email notification (or invoke a listener) from a particular workflow transition's post-function (see Workflow and Notifications).

Custom Events

You can fire a **custom event** from a custom transition post-function in a custom workflow. The appropriate listeners will be alerted of the custom transition by the firing of this event. For example, the associated notification scheme can be configured to notify users of the workflow transition based on the firing of this custom event.

Configuring Notifications for a Custom Event

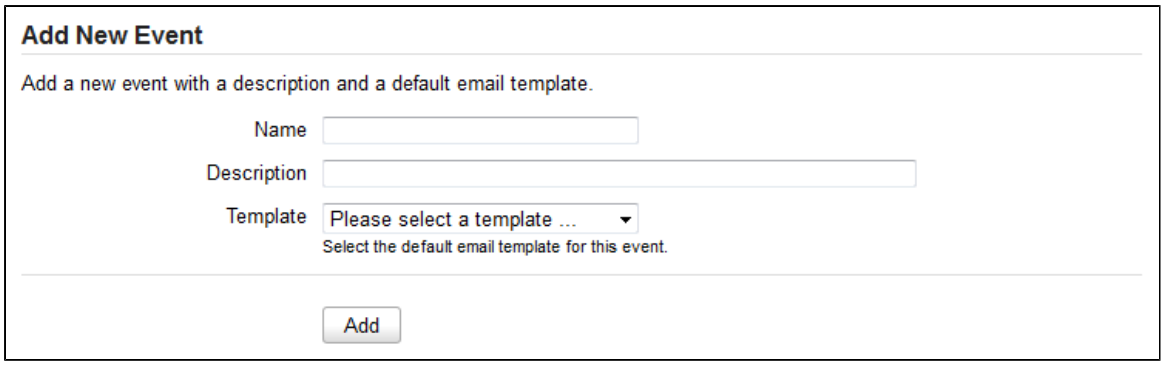
Custom events are most commonly used to generate notifications for custom workflow transitions. For example, your organisation might need you to modify the default workflow by adding a workflow step called 'QA_Inspection' (e.g. between **Resolve Issue** and **Close Issue**). You would typically also need to generate an email notification to the QA team whenever an issue progresses to the 'QA_Inspection' step of the workflow.

There are three overall steps to achieve this:

1. Add a custom event to the system (e.g. 'Issue Awaiting QA').
2. Configure the notification scheme to send an email when the custom event is fired.
3. Configure the workflow transition post-function to fire the custom event.

Step 1. Add a Custom Event

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Advanced > Events** to open the **View Events** page.
   Keyboard shortcut: **g + g +** start typing **events**
3. In the **Add New Event** form at the bottom of the page, add a **Name** and **Description** for the custom event by specifying them in these fields.



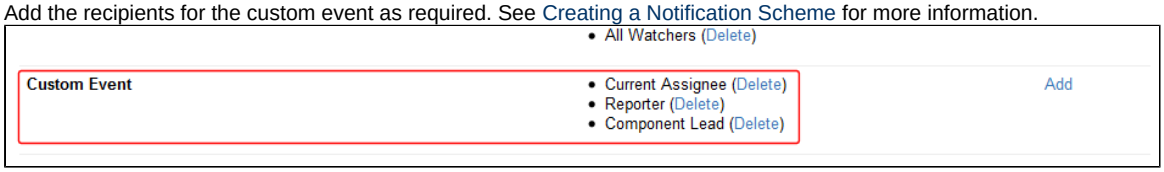
4. In the **Template** field, select the default email template to be associated with the event.
5. Click the **Add** button.

The custom event must be associated with a default email notification template. A notification scheme configured to notify users of this event will use this email template when sending the notification.

The custom event will appear in the list of events defined within the system. Initially, the event will be marked as **inactive** as it is not associated with a notification scheme or workflow post-function.

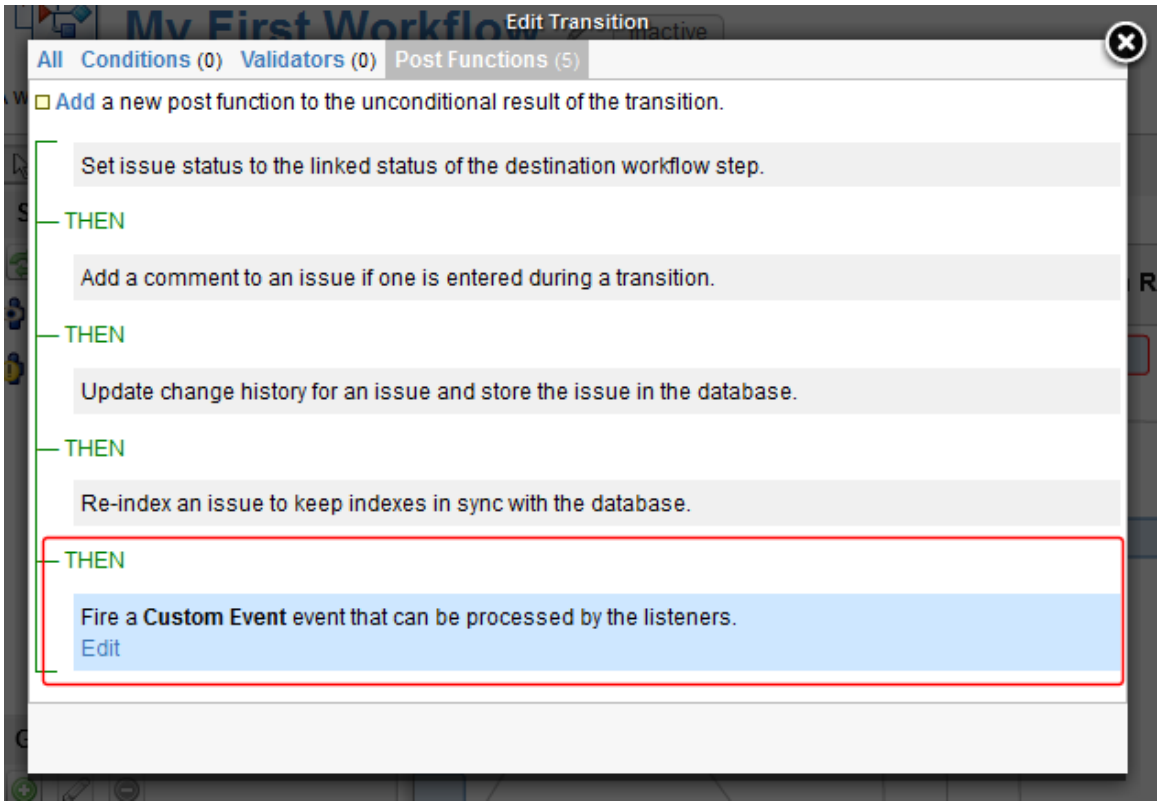
Step 2. Configure Notification Scheme to send mail on Custom Event

1. In 'Administration' mode, select **Issues > Notifications Schemes** to open the **Notification Schemes** page. You still need to be logged in as a user with the **JIRA Administrators** global permission to do this.
   Keyboard shortcut: **g + g +** start typing **notification schemes**
2. Select the notification scheme to edit, by clicking the notification scheme's name or its **Notifications** link (under **Operations**).
3. Add the recipients for the custom event as required. See Creating a Notification Scheme for more information.



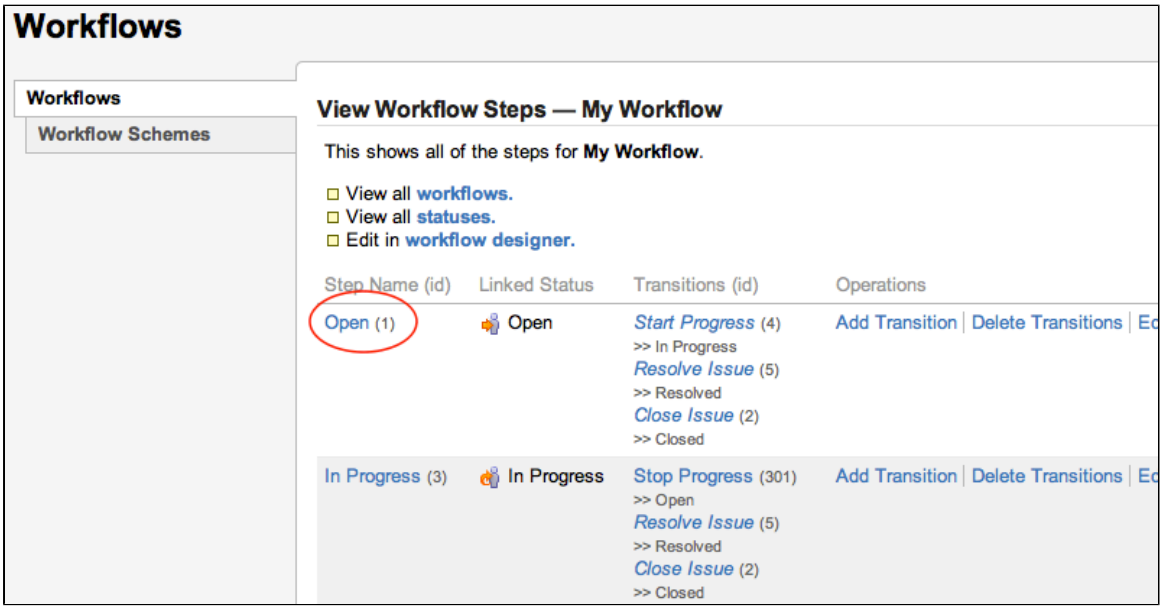
Step 3. Configure Workflow Transition Post-Function to Fire Custom Event

1. In 'Administration' mode, select **Issues > Workflows** to open the **Workflows** page. You still need to be logged in as a user with the **JIRA Administrators** global permission to do this.
   Keyboard shortcut: **g + g +** start typing **workflows**
2. Navigate to workflow transition post-function screen to be edited. See Configuring Workflow and Applying Post Functions to Transitions for more information.
3. Update the post-function to fire the custom event.
4. Activate or associate the workflow (and scheme) with the appropriate project. See Activating Workflow for more information.

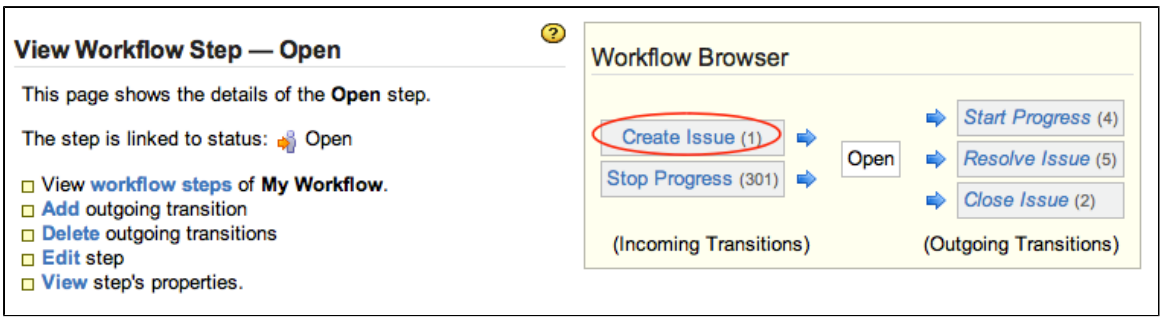


Configuring the Initial Status

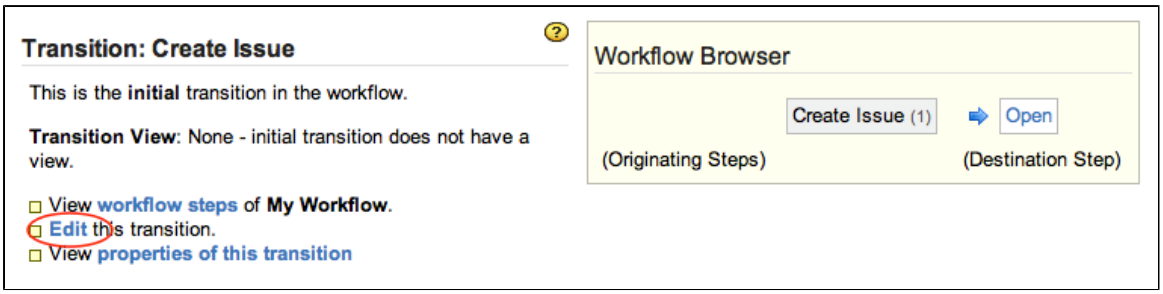
1. Usually the *Open* step will be the first status upon creating an issue. Click on it to edit it's properties:



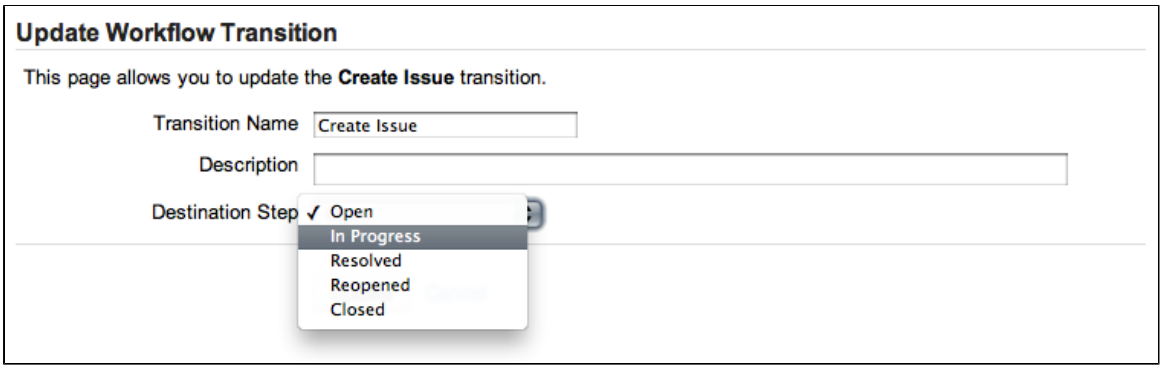
2. Click on the *Create Issue* incoming transition:



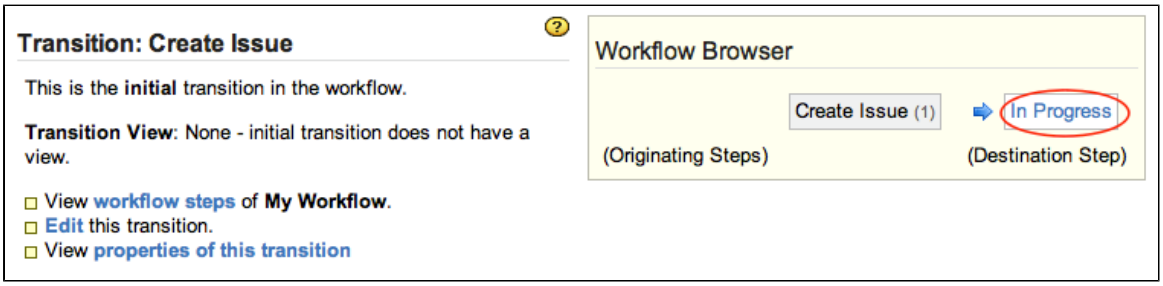
3. Click *Edit* to set the new destination step:



4.. Select you new *Destination Step*, and then click *Update* to save it:



5. This is what the result should look like, and now when a new issue is created it will go straight into the *In Progress* stap.



Workflow Properties

You can use workflow properties to implement restrictions on certain steps or transitions of a workflow.

For details on how to implement workflow properties in your workflow, please refer to Configuring Workflow.

Please Note: Not everything on this page is recommended!

- We do not recommend using all of these types of workflow properties as we cannot guarantee that some data and operations (e.g. bulk operations) will not be broken. **Hence, use these types of workflow properties your own risk!**

Available JIRA Workflow Properties

There are a few workflow properties which you can use in a transition or step of a workflow. Here are some helpful links:

- JIRA Forum - Field Properties
- JIRA API Documentation - JiraWorkflow constant values

Name	Values	Related Issues	References	Notes
`jira.field.resolution.exclude`	Resolution `id`		Resolutions per workflow step	Transition

`jira.field.resolution.include`	Resolution `id`	JRA-16443	Resolutions per workflow step	Transition
`jira.i18n.submit`	i18n property key	JRA-6798		Transition (usage: action submit button name)
`jira.i18n.title`	i18n property key	JRA-6798		Transition (usage: action name, etc.)
`jira.issue.editable`	`true`, `false`		Configuring Workflow	Step
`jira.permission.*`	user1, user2 / group1, group2 / ...?	JRA-6381	* WorkflowBasedPermissionManager class description (API documentation) * Forum discussion	Step
`opsbar-sequence`	Integer value greater than or equal to 0		Configuring Workflow (Customising Transitions)	Transitions on the 'View Issue' page

Configuring Email

- Configuring Email Notifications
- Creating Issues and Comments from Email
- Using Gmail as a JIRA Mail Server

Configuring Email Notifications

JIRA can send email notifications to users when significant events occur (e.g. creation of an issue; completion of an issue).

Enabling Email Notifications

To enable email notifications in JIRA:

1. Configure JIRA's SMTP mail server to send notifications.
2. Configure a notification scheme and associate it with the appropriate projects.

It is possible to customise your email content. The email address from which notifications are sent can also be configured for each project.

Disabling Email Notifications

To disable email notifications for a project, you can remove the notification scheme from the project by editing the project and selecting 'None' as the project's notification scheme.

Alternatively, you can edit the notification scheme so that no emails are sent.

On this page:

- Enabling Email Notifications
- Disabling Email Notifications
- Configuring a Project's Email Address
- Email Recipients
- Email HTML Formatting

In this chapter:

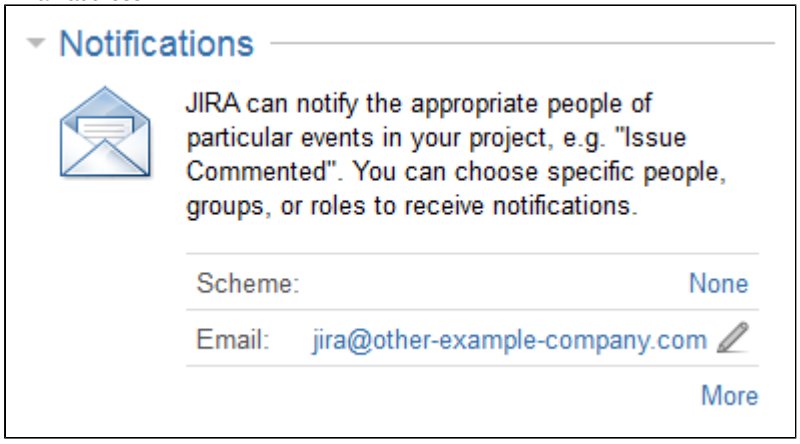
- Configuring JIRA's SMTP Mail Server to Send Notifications
- Creating a Notification Scheme
- Customising Email Content

Configuring a Project's Email Address

It is possible to configure a project's email address, which is the email address that notifications are sent from – i.e. the 'sender address'. This will also serve as the reply address for responses, which can work in conjunction with Creating Issues and Comments from Email.

By setting the **Sender Address** for a project, all notifications will be sent from this address. This setting is specific to the project selected and will not affect the configuration of the other projects. The **From address** specified in the **SMTP Mail Server** configuration is used as the default **Sender Address** for all projects.

The 'Sender Address' for a project can be configured as follows:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Click the **Administration** link at the top of the page.
3. Click **Projects** and select the project of interest from the dropdown menu. The 'Project Summary' page (see Defining a Project) for your selected project is shown.
   Keyboard shortcut: **g + g +** start typing the name of your project
4. At the lower-right section of the 'Project Summary' page, locate the **Notifications** section and click the 'pen' icon to the right of the **Email** address.
   

5. In the resulting **Project Email Address** dialog box, enter a valid email address in the **Sender Address** field and click **Update** to complete the process. This email address will now be used as the 'sender' address in all email notifications sent by this project.
   You can reinstate the default email address (as specified in the **SMTP Mail Server** configuration) by re-editing the **Sender Address** field (in the **Project Email Address** dialog box) but leaving it blank.

You cannot specify a project's email address until an **SMTP Mail Server** has been previously configured. See Configuring JIRA's SMTP Mail Server to Send Notifications for more information.

Email Recipients

For each event notification, JIRA will only send the first encountered email intended for a recipient. Hence, in the case where a user is included in two or more recipient lists (e.g. the **Project Lead** and current reporter) for one event notification, the user will only receive the first encountered email notification. JIRA will log the fact that this user was on multiple recipient lists.

Email HTML Formatting

Each JIRA user can specify in their own profile preferences whether to send outgoing emails in either text or HTML format. **JIRA Administrators** can specify a default email format under **Administration > Users > User Preferences**.

The HTML email format can accommodate internationalised words in the 'Issue Details' section. However, due to Internet Security Settings, which prevent images from being automatically downloaded, the HTML email messages may not be correctly formatted. For example, the summary column on the left may appear too wide. It is possible to correct the formatting by accepting to download these images. On some email clients, it is possible to do this in two different ways:

1. **Per email message:**
   - **Mozilla Thunderbird** — by clicking on the 'Show Remote Content' button above the email.
   - **Microsoft Outlook 2003** — by clicking on the 'Click here to download pictures. To help protect your privacy, Outlook prevented automatic download of some pictures in this message.' message above the email.
   - **Microsoft Outlook 2000** — does not have this option, it always downloads images.
   - **Microsoft Outlook Express 6** — by clicking on the 'Some pictures have been blocked to help prevent the sender from identifying your computer. Click here to download pictures.' message above the email.
2. **Configuring the email client:**
   - **Mozilla Thunderbird 1.5** — Navigate to **Tools > Options > Privacy > General** tab and ensure that *"Allow remote images if the sender is in my:"* option is checked and note which address book is selected. Then return to the e-mail sent from JIRA, right-click on the sender's e-mail address and choose *"Add to address book..."* option, adding this contact to the same address book as was selected in the Privacy options
   - **Microsoft Outlook 2003 and Outlook Express 6** — Navigate to **Control Panel > Internet Options.** On the Security tab, add JIRA's base URL to the trusted sites

Configuring JIRA's SMTP Mail Server to Send Notifications

To enable JIRA to send notifications about various events, you need to first configure an SMTP mail server in JIRA .

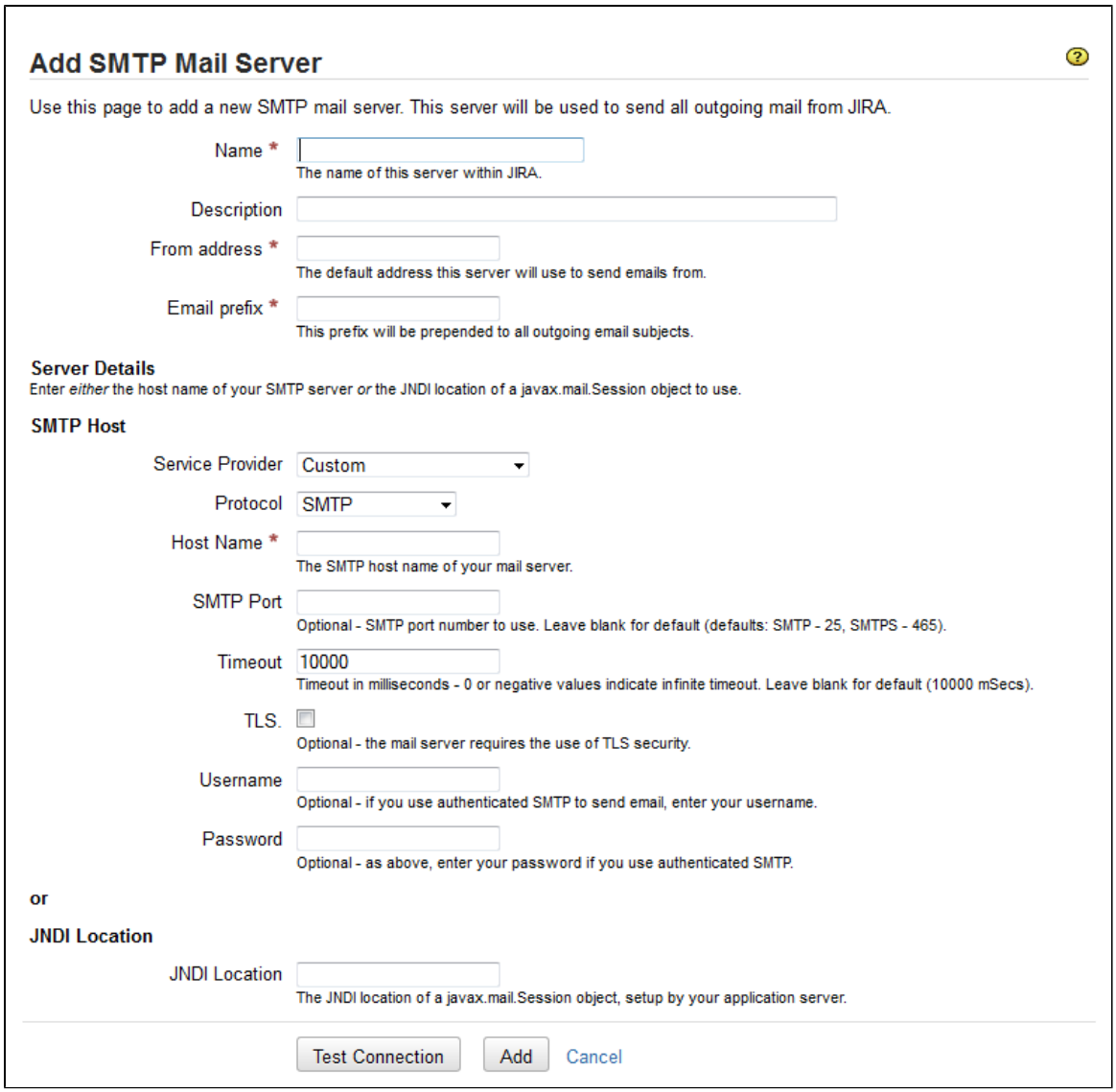
1. Define or edit the SMTP mail server

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **Administration > System > Mail > Outgoing Mail** to open the **SMTP Mail Server** page.
   Keyboard shortcut: **g + g +** start typing **outgoing mail**
   If no SMTP mail server has been defined, then a **Configure new SMTP mail server** button will be shown on the page. If one has already been defined, then the SMTP mail server's details will be shown on the page, along with a set of operation links at the right.
3. Click either the **Configure new SMTP mail server** button to define a new SMTP mail server, or the **Edit** link at the right to edit the existing SMTP mail server, which will open the **Add/Update SMTP Mail Server** page.

4. Complete the top section of this page as follows:

Name	Specify an arbitrary name to identify this SMTP mail server configuration.
Description	(Optional) Specify an arbitrary description that describes the SMTP mail server. This description appears below the **Name** of the SMTP mail server on the **SMTP Mail Server** configuration page.
From address	Specify the email address used in the 'sender address' (or 'from') field of notification messages sent by JIRA, unless overridden in a project configuration. Only specify an email address for this field (e.g. `jira@example-company.com`). JIRA will use this value to construct the full 'from' header based on the current user (`"Joe Bloggs (JIRA) <jira@example-company.com>"`). To change the 'from' header, go to **Administration > System > General Configuration** and under **Settings**) and edit the **Email from** field.
Email prefix	Specify the subject of emails sent from this server will use this string as a prefix. This is useful for your users so that they can filter email notifications from JIRA based on this prefix.

Screenshot: Add (or Update) SMTP Mail Server



On this page:

- 1. Define or edit the SMTP mail server
- 2. Specify a host name or JNDI location for your SMTP mail server
- Configuring a JNDI location
- Troubleshooting

*The information on this page does not apply to **JIRA OnDemand**.*

2. Specify a host name or JNDI location for your SMTP mail server

The second part of the **Add/Update SMTP Mail Server** page specifies the **Server Details** of the SMTP mail server to which JIRA will send mail. There are two ways you can do this. Either:

- specify the **SMTP host** details of your SMTP mail server;
  or:
- specify the **JNDI location** of a `javax.mail.Session` object — that is, use JNDI to look up an SMTP mail server that you have preconfigured in your application server. This has the following advantages:
  - **Better security**: the mail details are not available to JIRA administrators through the JIRA administration interface and are not stored in JIRA backup files.
  - **More SMTP options**: for instance, you could switch to RSET instead of NOOP for testing connections by setting the `mail.smtp.userset` property.
  - **Centralised management**: mail details are configured in the same place as database details and may be configured through your application server administration tools.

Specify the SMTP host details

Most people configure JIRA's SMTP mail server by specifying the SMTP host details of this mail server directly in JIRA.

1. In the **SMTP host** section of the **Add/Update SMTP Mail Server** page (above), complete the following form fields:

Service Provider (not available when updating an existing SMTP mail server)	Choose between using your own SMTP mail server (i.e. **Custom**), or either Gmail (i.e. **Google Apps Mail / Gmail**) or Yahoo! (i.e. **Yahoo! MailPlus**) as the service provider for your SMTP mail server. If you choose either Gmail or Yahoo! options and then switch back to **Custom**, some of the key fields in this section will automatically be populated with the relevant SMTP mail server settings for these service providers.
Protocol	Choose between whether your SMTP mail server is a standard (i.e. **SMTP**) or a secure (i.e. **SECURE_SMTP**) one.
Host Name	Specify the hostname or IP address of your SMTP mail server. Eg. `smtp.yourcompany.com`
SMTP Port	*(Optional)* The SMTP port number, usually 25 for SMTP or 465 for SMTPS, either of which are assumed if this field is left blank.
Timeout	*(Optional)* Specify the timeout period in milliseconds, which is treated as 10000 if this field is left blank. Specifying 0 or a negative value here will result in JIRA waiting indefinitely for the SMTP server to respond.
TLS	*(Optional)* Select this check box if your SMTP host uses the Transport Layer Security (TLS) protocol.
Username	*(Optional)* If your SMTP host requires authentication, specify the username of these authentication credentials here. (Most company servers require authentication to relay mail to non-local users.)
Password	*(Optional)* Again, if your SMTP host requires authentication, spcify the password associated with the username you specified above. When editing an existing SMTP mail server, select the **Change Password** check box to access and change this field.

Please Note:

- If your server's startup script uses the `-Dmail` system properties (e.g. `mail.smtp.host` or `mail.smtp.port`), they will override the settings that you specify in the above form. Additionally, if necessary you can manually specify the host name that JIRA reports itself as to the SMTP server by setting `-Dmail.smtp.localhost`

2. *(Optional)* Click the **Test Connection** button to check that JIRA can communicate with the SMTP mail server you just configured.
3. Click the **Add** (or **Update**) button to save JIRA's SMTP mail server configuration.

Specify a 'JNDI Location'

As an alternative to specifying SMTP host details directly in JIRA, you can configure them in your application server, and then look up a preconfigured mail session via JNDI.

In the **JNDI Location** section of the **Add/Update SMTP Mail Server** page (above), specify the location of a `javax.mail.Session` object to use when sending email, in the **JNDI Location** field. This will begin with the prefix `java:comp/env/`

Configuring a JNDI location

The **JNDI Location** that you specify in JIRA will depend on JIRA's application server and configuration. JNDI locations are typically configured in the application server that runs JIRA. Hence, JIRA will need to be restarted after configuring a JNDI location for that configuration to be available in JIRA.

For example, in Tomcat 6 (the application server bundled with 'recommended' distributions of JIRA), your **JNDI Location** would be `java:comp/env/mail/JiraMailServer` and you would add the following section to the `conf/server.xml` of your JIRA Installation Directory, inside the `<Context/>` node:

```
<Context path="" docBase="${catalina.home}/atlassian-jira" reloadable="false">
   ...
     <Resource name="mail/JiraMailServer"
       auth="Container"
       type="javax.mail.Session"
       mail.smtp.host="mail.yourcompany.com"
       mail.smtp.port="25"
       mail.transport.protocol="smtp"
       mail.smtp.auth="true"
       mail.smtp.user="jirauser"
       password="mypassword"
     />
   ...
</Context>
```

Or if you do not require authentication (e.g. if you are sending via localhost, or only internally within the company):

```
<Context path="" docBase="${catalina.home}/atlassian-jira" reloadable="false">
   ...
     <Resource name="mail/JiraMailServer"
       auth="Container"
       type="javax.mail.Session"
       mail.smtp.host="localhost"
       mail.smtp.port="25"
       mail.transport.protocol="smtp"
     />
   ...
</Context>
```

If you happen to be running JIRA on an application server other than Apache Tomcat (which is not a supported JIRA configuration), a similar methodology for configuring a JNDI location to your SMTP mail server should apply to that application server. For details, please see the Transaction Factory documentation.

If you have problems connecting, add a `mail.debug="true"` parameter to the `<Resource/>` element (above), which will let you see SMTP-level 'debugging' details when testing the connection.

Move the JavaMail Classes

You will also need to ensure that the JavaMail classes (typically in JAR library files) are present in your application server's classpath and that these do not conflict with JIRA's JAR library files. This is necessary because the application server itself (not JIRA) is establishing the SMTP connection and as such, the application server can not see the JAR library files in JIRA's classloader.

Some operating systems may bundle the JavaMail classes with application servers (eg. **Tomcat in Red Hat Enterprise Linux**). This may conflict with JIRA's copy of the JavaMail classes, resulting in errors like:

```
java.lang.NoClassDefFoundError: javax/mail/Authenticator
```

or:

```
java.lang.IllegalArgumentException: Mail server at location [java:comp/env/mail/JiraMailServer]
is not
         of required type javax.mail.Session.
```

Lighter application servers such as Apache Tomcat (including the one incorporated into the 'recommended' distributions of JIRA), do not always come with JavaMail.

To prevent any conflicts, check your application server's `lib/` directory:

- If the application server already contains `mail-1.4.1.jar` and `activation-1.1.1.jar`, then just **remove** `mail-1.4.1.jar` and `activation-1.1.1.jar` from the `<jira-application-dir>/WEB-INF/lib/` subdirectory of the JIRA Installation Directory.
- If the application server does not contain `mail-1.4.1.jar` and `activation-1.1.1.jar`, then **move** the `mail-1.4.1.jar` and `activation-1.1.1.jar` from the `<jira-application-dir>/WEB-INF/lib/` subdirectory of the JIRA Installation Directory into the the `lib/` subdirectory of the JIRA Installation Directory (for 'recommended' distributions of JIRA) or the `lib/` subdirectory of the application server running JIRA.

SMTP over SSL

You can encrypt email communications between JIRA and your mail server via SSL, provided your mail server supports SSL.

Firstly, you will need to **import the SMTP server certificate** into a Java keystore. The process is described on the Connecting to SSL Services page.

Important Note: Without importing the certificate, JIRA will not be able to communicate with your mail server.

Secondly, edit your mail server connection properties and specify `starttls` and `SSLSocketFactory`. From `{$JIRA_INSTALL}/conf/server.xml` (this example uses Gmail's server):

```
<Resource name="mail/GmailSmtpServer"
   auth="Container"
   type="javax.mail.Session"
   mail.smtp.host="smtp.gmail.com"
   mail.smtp.port="465"
   mail.smtp.auth="true"
   mail.smtp.user="myusername@gmail.com"
   password="mypassword"
   mail.smtp.starttls.enable="true"
   mail.smtp.socketFactory.class="javax.net.ssl.SSLSocketFactory"
/>
```

Troubleshooting

A useful tip for debugging mail-related problems in JIRA is to set the `-Dmail.debug=true` property on startup. This will cause protocol-level details of JIRA's email interactions to be logged. Additionally, turning up JIRA's log level will show when the service is running and how mails are processed.

Common Problems

- If JIRA does not appear to be creating or sending emails or creating issues and comments from email, your JIRA installation could be experiencing **OutOfMemory errors**. Please check your log files for OutOfMemory errors. If there are OutOfMemory errors, please restart JIRA and investigate the errors.
- If you find some incoming emails simply disappear, check that you have not accidentally **started a second copy of JIRA** (eg. in a staging environment) which is downloading and deleting email messages. See the Restoring Data page for flags you should set to prevent mail being processed.
- If you receive 'Mail Relay' errors, make sure you have specified the **Username** and **Password** in the **SMTP Host** section of JIRA's **SMTP Mail Server** configuration page.

Getting Help

If you cannot resolve a problem yourself, please create a support case in the 'JIRA' project and we will assist.

Creating a Notification Scheme

About Email Notifications

JIRA can generate email notifications for various *events* that happen throughout the lifecycle of an issue. Notifications are defined within a *notification scheme* (see below), which associates particular events with particular email recipients. The notification scheme is then assigned to a particular project.

You can use the same notification scheme for more than one project.

JIRA is pre-packaged with a notification scheme called **Default Notification Scheme** which, by default, is associated with all newly-created projects. You can modify this scheme or if you prefer, create other notifications schemes for particular projects.

On this page:

- About Email Notifications
- Creating a notification scheme
- Adding an event recipient to a notification scheme
- Associating a notification scheme with a project

Creating a notification scheme

1. Log in as a user with the **JIRA Administrators** global permission.

2. Select **Administration > Issues > Notification Schemes** to open the **Notification Schemes** page (see above), which lists all notification schemes that currently exist in your JIRA installation.
   Keyboard shortcut: **g + g +** start typing **notification schemes**
3. Start creating the new notification scheme, by doing either of the following:
   - Click the **Copy** link to copy an existing notification scheme. If you have a notification scheme whose event recipients are reasonably similar to what you require, creating a copy is the quickest way to add a new scheme.
     OR
   - Click the **Add Notification Scheme** button. On the **Add Notification Scheme** page, enter a name for the notification scheme and a short description of the scheme
4. If you added a new notification scheme or you copied an existing one but have clicked the **Edit** link to modify the automatically generated name and/or description of the copied notification scheme:
   a. Enter a name (or modify the existing one) for the notification scheme (e.g. 'Angry Nerds Notification scheme').
   b. *(Optional)* Enter a description (or modify the existing one) for the notification scheme.
   c. Click the **Add** button to create the notification scheme.
5. Add notifications/recipients as described below.
6. Associate your new notification scheme with a project as described below.

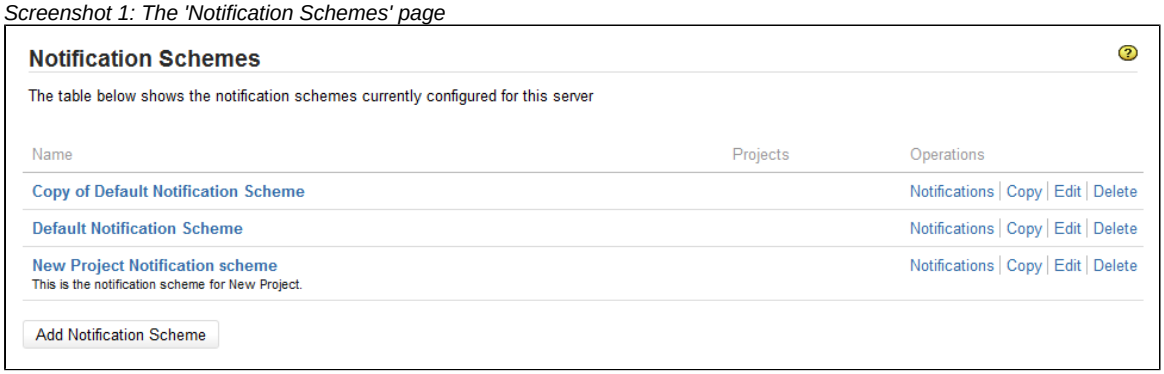
Adding an event recipient to a notification scheme

To add a new recipient for a particular event to a notification scheme, you need to:

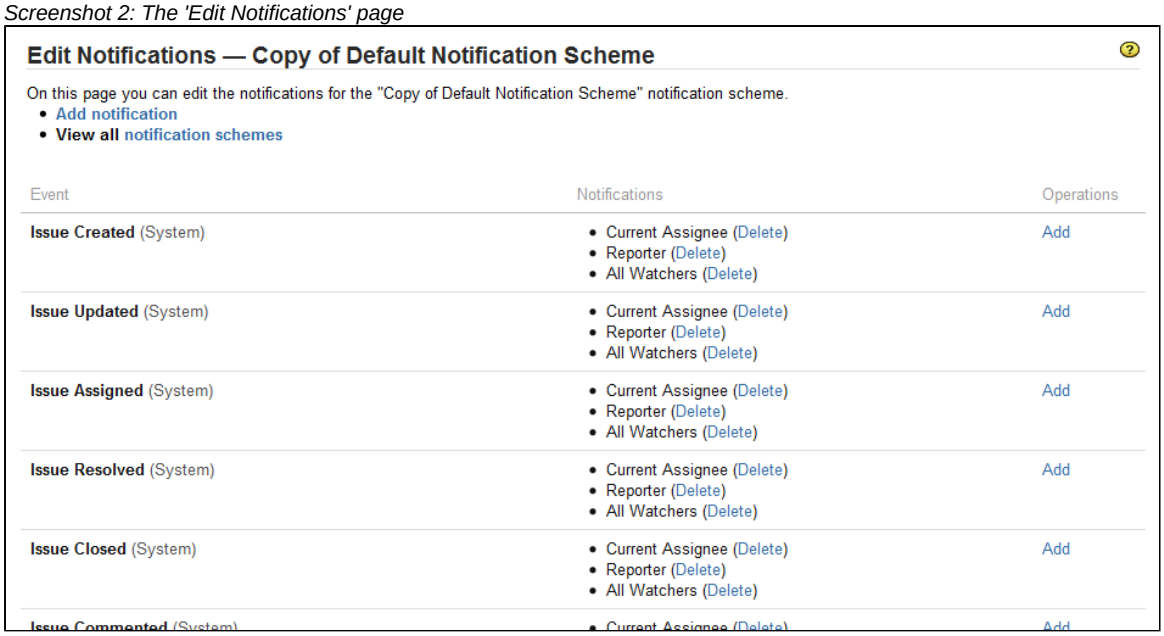
1. Identify the notification scheme used by the relevant project.
2. Add that recipient to the appropriate event in this notification scheme.

To add a new recipient for a particular event:

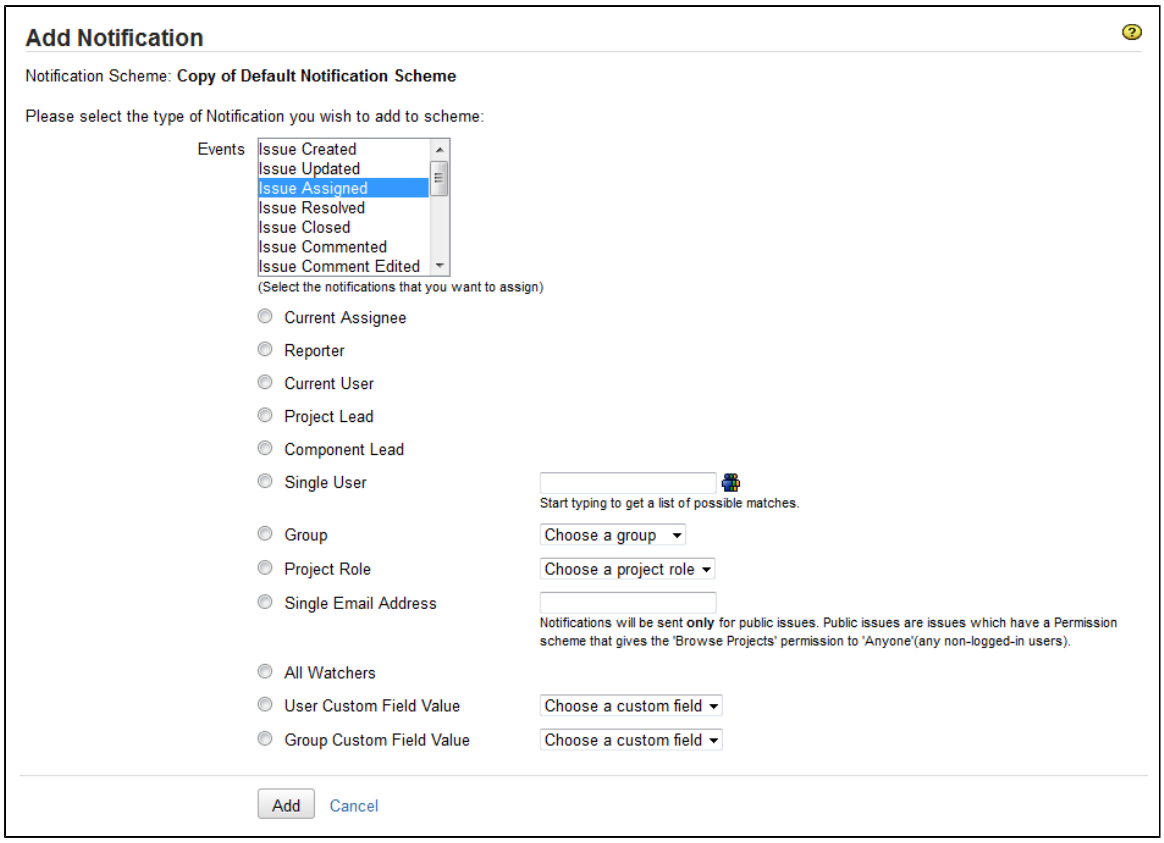
1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Issues > Notification Schemes** to open the **Notification Schemes** page.
   Keyboard shortcut: **g + g +** start typing **notification schemes**
   Screenshot 1: The 'Notification Schemes' page



3. Locate the notification scheme of interest and click its linked name to open the **Edit Notifications** page for that notification scheme. The **Edit Notifications** page lists all of the events (mentioned below), along with the recipients who will receive notifications when each event occurs:
   Screenshot 2: The 'Edit Notifications' page



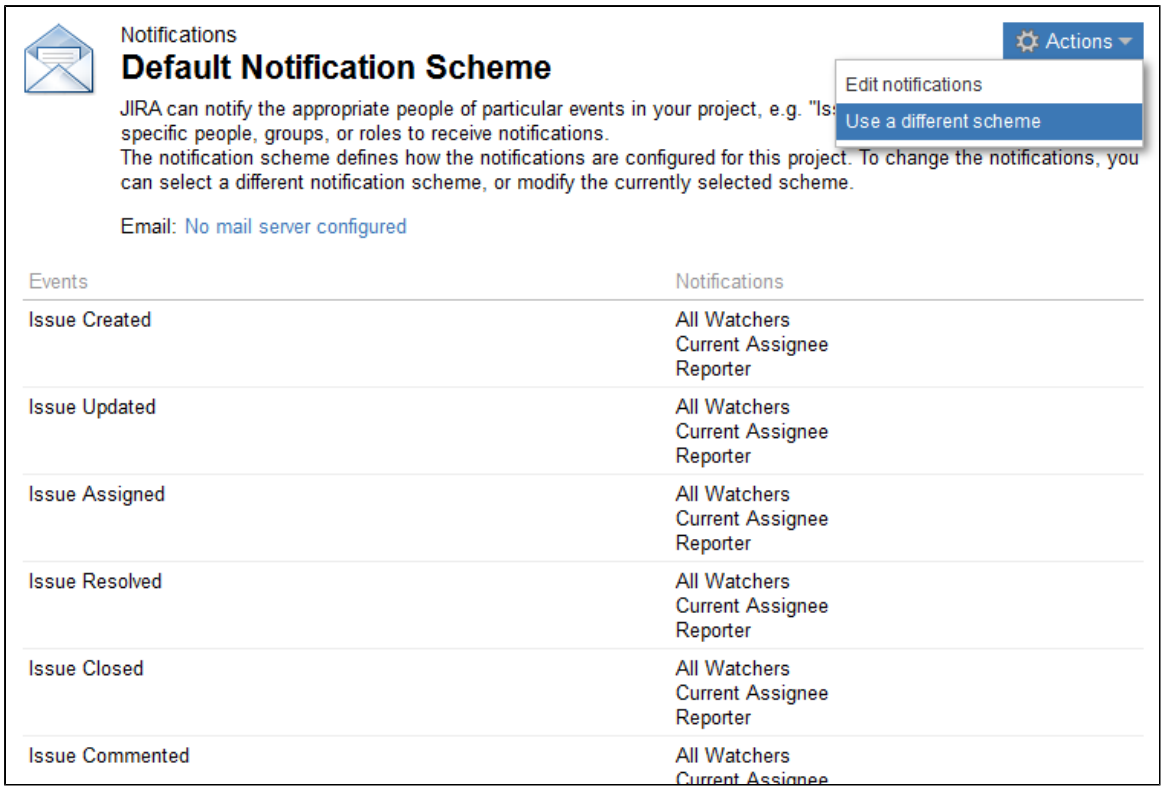
4. Click the **Add** link in the appropriate event row (see the list of events below), which opens the **Add Notification** page, where you can choose who to notify (about the event) from the list of available recipients (see below).
   Screenshot 3: The 'Add Notification' page



5. Select the appropriate recipient (filling in any required information for your particular choice of recipient).
6. Click the **Add** button. You are taken back to the **Edit Notifications** page (see above), with the notification you just specified now listed against the appropriate issue event.
7. If you make a mistake, or you would like to remove who is being notified, simply click the **Delete** link beside the person/group/role.

Associating a notification scheme with a project

1. Log in to JIRA as a project administrator. (A project administrator is someone who has the **Administer Project** project permission, but not necessarily the **JIRA Administrators** global permission.)
2. Click the **Administration** link at the top of the screen.
3. Click **Projects** and select the project of interest from the dropdown menu. The 'Project Summary' page (see Defining a Project) for your selected project is shown.
   Keyboard shortcut: **g + g +** start typing the name of your project
4. At the lower-right of the 'Project Summary' page, locate the **Notifications** section, click the name of the current scheme (e.g. **Default Notification Scheme**) or **None** (if the project is not yet associated with a scheme) to display details of the project's current notification scheme.
5. Click the **Actions** dropdown menu and choose **Use a different scheme** (or **Select a scheme**).
   Screenshot 4: The Project Notifications page



6. On the subsequent **Associate Notification Scheme to Project** page, which lists all available notification schemes, select the notification scheme you want to associate with the project and click the **Associate** button.

See also Minimising the number of Permission Schemes and Notification Schemes.

Events

JIRA supports the following events, which can generate email notifications (as defined in a notification scheme).

Event	Description
Issue Created:	An issue has been entered into the system.
Issue Updated:	An issue has had its details changed. This includes the deletion of an issue comment.
Issue Assigned:	An issue has been assigned to a new user.
Issue Resolved:	An issue has been resolved (usually after being worked on and fixed).
Issue Closed:	An issue has been closed. (Note that an issue may be closed without being resolved; see Workflow).
Issue Commented:	An issue has had a comment added to it.
Issue Comment Edited:	An issue's comment has been modified.
Issue Reopened:	An issue has been re-opened.
Issue Deleted:	An issue has been deleted.
Issue Moved:	An issue has been moved into or out of this project.
Work Logged On Issue:	An issue has had hours logged against it (i.e. a worklog has been added).

Work Started On Issue:	The Assignee has started working on an issue.
Work Stopped On Issue:	The Assignee has stopped working on an issue.
Issue Worklog Updated:	An entry in an issue's worklog has been modified.
Issue Worklog Deleted:	An entry in an issue's worklog has been deleted.
Generic Event:	The exact nature of this event depends on the workflow transition(s) from it was fired.
Custom Event(s):	The exact nature of these events depends on the workflow transition(s) from which they were fired.

JIRA does not have a specific notification event for the deletion of issue comments. When an issue's comment is deleted, JIRA sends out an email notification as an 'Issue Updated' event.

Recipients

The following types of recipients can receive email notifications.

Recipient	Description
Current Assignee	The user to whom the issue is currently assigned.
Reporter	The user who originally created the issue.
Current User	The user who performed the action that has triggered this event.
Project Lead	The user who is managing the project to which the issue belongs.
Component Lead	The user who is managing the component to which the issue belongs.
Single User	A particular user in your JIRA system.
Group	A particular group in your JIRA system.
Project Role	The members of a particular project role for this project. Note that it is recommended to use project roles (rather than groups) in your notifications as this can help minimise the number of notification schemes in your system.
Single Email Address	Any email address that you wish to alert. A Single Email Address notification will only be sent if the issue is publicly viewable (as the email address of a non-JIRA user could be specified, in which case a security check is not possible). Publicly viewable issues are issues which have a Permission scheme that gives the 'Browse Projects' permission to 'Anyone' (any non-logged-in users).
All Watchers	All users who are watching the issue.
User Custom Field Value	The value of a custom field of type *User Picker* or *Multi User Picker* that may have been associated with issues. An example of where this can be useful: if you have a custom User field called Tester, you can have the tester notified when an issue is resolved.
Group Custom Field Value	The value of a custom field of type *Group Picker* or *Multi Group Picker* that may have been associated with issues..

Please Note:

- Email notifications will only be sent to people who have permission to view the relevant issue — that is, people who:
  - have the **Browse Projects** project permission for the project to which the issue belongs; and
  - are members of any Issue security levels that have been applied to the issue.
- JIRA can only send email notifications if SMTP email has been enabled (see Email Overview).

Please also note:

JIRA will send notification emails to both the **previous assignee and the current assignee**, whenever the assignee field changes.

However, earlier versions of JIRA only sent a notification email to the previous assignee *if* the operation that changed the event was the **Assign Issue** operation. It did not send a notification if the issue was edited in some other way.
The **`jira.assignee.change.is.sent.to.both.parties`** advanced JIRA option allows this legacy behaviour to be re-instated, for those customers who prefer this behaviour behaviour.

See JRA-6344 for more details.

Customising Email Content

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

To change the columns in your filter subscriptions, you don't need to customise the mail templates. See Customising your Issue Navigator.

There's a feature request to improve this at JRA-7266. Please vote.

JIRA generates emails in reaction to events using a templating engine. The templating engine is Apache Jakarta's Velocity. This is a relatively easy to use templating language that can pull apart java objects in useful ways. The mails are generated inside JIRA by invoking Velocity with a set of objects of relevance to the event.

Email Template Locations

To customise email content, please follow this procedure.

1. Open up your JIRA distribution, and navigate to the following paths:
   - The `WEB-INF/classes/templates/email/` of the `<jira-application-dir>` in your JIRA Installation Directory.
   - The `jira/src/etc/java/templates/email/` in your extracted JIRA source directory.
2. Under this directory there are three directories: `html`, `text` and `subject`. The `html` subdirectory contains the templates used to create emails in html, while the `text` directory the plain text mail outs. The `subject` directory contains the templates used to generate the subject of the emails. The templates are named after the event that will trigger the email.
3. Bring the template up in your favourite text editor. Referring to the JIRA template documentation (particularly Velocity Context for Email Templates) and Velocity Users Guide, make the customisations you want.
4. Restart JIRA.

For new email templates:

1. Create your new `mytemplate.vm` files in the `html`, `text` and `subject` directories, based on the existing files in those directories
2. Add the templates to `atlassian-jira/WEB-INF/classes/email-template-id-mappings.xml` to make them valid choices for when you are adding a new event.

Note that since JIRA 4.1 each new template has to have a corresponding file in the `subject` directory.

Advanced Customisation

The Issue object is passed into the vm templates. Notice some of it's implementation in `/includes/summary-topleft.vm`. As an example, calling $issue.getProject() would allow you to determine the project an issue comes from, and even create logic to show different information for emails from different projects.

See also Adding Custom Fields to Email.

Creating Issues and Comments from Email

JIRA can be configured to automatically create issues or comments on existing issues based on incoming messages received by a mail server or external mail service.

This is especially useful in a helpdesk or support scenario, where users send support queries via email that you wish to track with JIRA. Subsequent email messages about the issue (for example, responses to Email Notifications) can be automatically recorded as comments. Additionally, any attachments in the emails can automatically be attached to the issue (with appropriate configuration).

Configuring issue or comment creation from email

Issues and comments in JIRA can be generated either from:

- email messages sent to an account on a POP or IMAP mail server, or
- messages written to the file system generated by an external mail service.

Step one: Configure a mail server/service

POP or IMAP email messages

To set up issue and comment creation from email, you will need to create a mail account for a POP or IMAP mail server that JIRA can access – typically, one mail account for each JIRA project. For example, for the 'ABC' project, you might establish an account `abc-issues@example-company.com`

JIRA will periodically scan for new email messages received by your mail account (via a service) and appropriately create issues or comments for any emails it finds (via a mail handler).

JIRA's mail handlers can also *optionally* create new user accounts for senders not previously seen. Note that this is not possible if you are using External User Management.

Once you have created a mail account on a POP or IMAP mail server, configure JIRA to receive email from that mail server account.

Tip: You can configure JIRA's mail servers so that recipients of email notifications can simply reply to these messages and have the body of their replies added as comments to the relevant issue. To do this, simply set the **From address** in JIRA's SMTP mail server to match that of the POP or IMAP mail server's account being monitored. (In most cases, this means having JIRA's SMTP and POP or IMAP mail servers use the same mail account.) Details on how to configure JIRA to handle these emailed replies is mentioned below.

File system messages

To set up issue and comment creation from messages written to the file system by an external mail service, your external mail service must be able to write these messages within the `import/mail` subdirectory of the JIRA Home Directory.

External mail services are very much like the POP or IMAP services above, except that instead of email messages being read from a mail account, they are read from a directory on the disk. External mail services are useful because they overcome the potential security risks associated with anonymous mail accounts. Instead you can simply configure your external mail service to dump incoming email messages within the JIRA Home Directory's `import/mail` subdirectory, which is scanned periodically.

Please also be aware that JIRA expects only one message per file, so your external mail service should be configured to generate such output.

Please Note — how JIRA handles messages on a mail server/service:

- For mail accounts, JIRA scans email messages received by your mail account's 'Inbox' folder. However, for IMAP mail servers, you can specify a different folder within your mail account.
- If JIRA successfully processes a message, JIRA deletes the message from your mail account (on a POP or IMAP mail server) or file system (i.e. for file system messages).
- If JIRA does not successfully process a message, the message will remain either in your mail account or on the file system.

On this page:

- Configuring issue or comment creation from email
  - Step one: Configure a mail server/service
  - Step two: Configure a mail handler
- Mail handlers
- Issue/comment creation
- Handy tips with mail handlers
- Best practices (pre-processing JIRA email messages)
- Troubleshooting

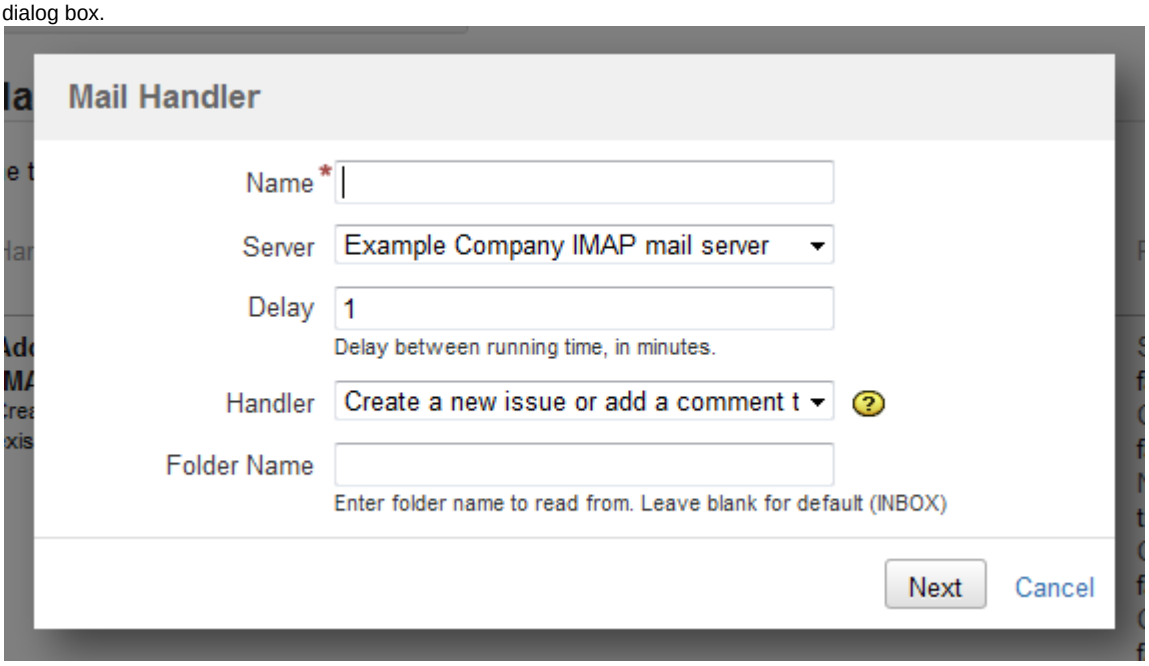
*Some functionality described on this page is restricted in **JIRA OnDemand**.*

Step two: Configure a mail handler

Once you have configured JIRA to receive messages from a mail server/service, you configure JIRA to handle these messages through a 'mail handler'.

To configure a JIRA mail handler:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Mail > Incoming Mail** to access the **Mail Handlers** section of the **Incoming Mail** page (below the **POP / IMAP Mail Servers** section).
   Keyboard shortcut: **g + g +** start typing **incoming mail**
3. Click the **Configure new incoming mail handler** button (or the **Edit** link next to an existing mail handler) to open the **Mail Handler** dialog box.



4. Specify a **Name** that describes what your mail handler will do — for example, 'Create issues or comments from Example Company's IMAP mail server'.
5. Select the mail **Server** that you configured in step one (above). This is either a POP or IMAP mail server or the **Local Files** option for an external mail service that writes messages to the file system.
6. Specify the **Delay** (in minutes) between the mail handler's running time. This effectively defines the frequency with which JIRA scans the **Server** that you specified in the previous step.
7. Choose the type of mail **Handler** from dropdown list. For more information, refer to the Mail Handlers section below.
8. If you chose either an IMAP mail server or the **Local Files** option in the **Server** field, then a **Folder Name** field appears below the **Handler** dropdown list:
   - For an IMAP mail server, if you want mail handler to scan for new messages from a folder other than the 'Inbox' in your mail account, specify the name of that folder here.
   - For the **Local Files** option, if your file messages are being written to a subdirectory within the `import/mail` subdirectory of the JIRA Home Directory, specify the subdirectory structure (within `import/mail`) here.
9. Click **Next** to continue with specifying the remaining options specific to mail **Handler** you selected above. For more information, refer to the Mail Handlers section below.
10. *(Optional)* Click the **Test** button to test your mail handler.
11. Click the **Add / Save** button to save your mail handler.

Please Note — the relationship between JIRA mail handlers and services:

- A JIRA mail handler is part of a JIRA service. Hence, when you create a mail handler, its service will appear as an entry on the **Services** page.
- Be aware that editing mail handlers can only be performed through the **Mail Handlers** page (described above).

- On the **Mail Handlers** page, clicking the **Delete** link associated with a mail handler removes that handler. Since a mail handler is part of a service, then if you delete a mail handler's service on the **Services** page, its associated handler will also be removed from the **Mail Handlers** page.

Mail handlers

JIRA provides the following default mail handlers:

- Create a new issue or add a comment to an existing issue
- Add a comment from the non quoted email body
- Add a comment with the entire email body
- Create a new issue from each email message
- Add a comment before a specified marker or separator in the email body
- Custom mail handlers

For more information about how these mail handlers create issues and comments in JIRA, refer to Issue/comment creation (below).

Also refer to the Handy tips with mail handlers (below) for tips on tweaking mail handlers to allow JIRA to handle the following types of email messages:

- Email sent from people without a JIRA user account.

Create a new issue or add a comment to an existing issue

This message handler creates a new issue, or adds a comment to an existing issue. If the subject contains an issue key, the message is added as a comment to that issue. If no issue key is found, a new issue is created in the default project.

To configure a 'Create a new issue or add a comment to an existing issue' mail handler:

1. If you have not already done so, begin configuring your mail handler (above).
2. On the **Create a new issue or add a comment to an existing issue** dialog box, complete the following fields/options:

Project	Specify the project key of the default project to which new issues are created by this handler — for example, `JRA`. **Please Note:** • This field is only relevant for issue creation, not for issue commenting. • If an email message contains an issue key in its subject line and that issue key exists in your JIRA installation, the handler will add the email message content as a comment on the issue, regardless of which project the issue is in.
Issue Type	Choose the default issue type for new issues.
Strip Quotes	Select this check box to remove quoted text from from an email message's body (e.g. from previous email replies) before the body's content is added to the JIRA issue's comment.
Catch Email Address	If specified, only email messages whose **To:**, **Cc:**, **Bcc:** lines contain the recipient specified in this field will be processed — for example, `issues@mycompany.com` Upon specifying an address here, all email messages whose **To:**, **Cc:**, **Bcc:** lines contain addresses other than the **Catch Email Address** are ignored. This is useful if you have multiple aliases for the same mail account (e.g. `foo-support@example-co.com` and `bar-support@example-co.com` aliases for `support@example-co.com`) for multiple mail services (e.g. each one to create issues in separate JIRA projects). **Please note:** in practice, **this option is rarely useful** and should not be confused with the more common **Default Reporter**. You can only specify one catch email address and one issue type per mail handler.
Bulk	This option only affects 'bulk' email messages whose header has either its **Precedence:** field set to **bulk** or its **Auto-Submitted** field set to **no**. Such messages would typically be sent by an automated service. When such an email message is received, the following action will be performed, based on the option you choose: a. Ignore the email and do nothing. b. Forward the email (i.e. to the address set in the **Forward Email** text field). c. Delete the email permanently. It is generally a good idea to set **bulk=forward** and set a Forward Email address, to prevent mail loops between JIRA and another automated service (eg. another JIRA installation).
Forward Email	If specified, then if this mail service is unable to handle an email message it receives, an email message indicating this problem will be forwarded to the email address specified in this field. **Please Note:** An SMTP mail server must be configured for this option to function correctly.

Create Users	Select this check box if you want JIRA to create new user accounts from any received email messages whose **From:** field contains an address that does not match one associated with an existing JIRA user account. This allows the creator of the email message to be notified of subsequent updates to the issue, which can be achieved by configuring the relevant project's notification scheme to notify the **Reporter** of updates. The username and email address of these newly created JIRA user accounts will be the email addresses specified in the **From:** fields of these received messages. The password for these new JIRA users is randomly generated and an email message is sent their addresses informing them about their new JIRA user account. **Note:** this option is not compatible with **Default Reporter** field option below and as such, choosing the **Create Users** option will hide the **Default Reporter** option.
Default Reporter	Specify the username of a default reporter, which will be used if the email address in the **From:** field of any received messages does not match the address associated with that of an existing JIRA user — for example, a JIRA username such as `emailed-reporter` **Please Note:** • This option is not available if the **Create Users** check box is selected. • Please ensure that the user specified in this field has the **Create Issues** project permission for the relevant **Project** (specified above) as well as the **Create Comments** project permission for the other relevant projects to which this mail handler should add comments. • When an issue is created and this option is specified, the email message's **From:** field address is appended in a brief message at the end of the issue's **Description** field, so that the sender can be identified.
Notify Users	Clear this check box if you do not want JIRA to send out an email message notifying users whose accounts have been created by the **Create Users** option above. **Note:** this option only functions if the **Create Users** check box has been selected.

3. Test and save your mail handler (above).

Add a comment from the non quoted email body

This message handler creates a comment, but only uses the 'non quoted' lines of the body of the email message. A quoted line is any line that starts with a '>' or '|' symbol and such lines of text will not be added to the comment. The issue to which the comment is added is chosen from the first issue key found in the email subject. The author of the comment is taken from the address of the email message's **From:** field.

To configure an 'Add a comment from the non quoted email body' mail handler:

1. If you have not already done so, begin configuring your mail handler (above).
2. On the **Create a new issue or add a comment to an existing issue** dialog box, complete the following fields/options:

Catch Email Address	If specified, only email messages whose **To:**, **Cc:**, **Bcc:** lines contain the recipient specified in this field will be processed — for example, `issues@mycompany.com` Upon specifying an address here, all email messages whose **To:**, **Cc:**, **Bcc:** lines contain addresses other than the **Catch Email Address** are ignored. This is useful if you have multiple aliases for the same mail account (e.g. `foo-support@example-co.com` and `bar-support@example-co.com` aliases for `support@example-co.com`) for multiple mail services (e.g. each one to create issues in separate JIRA projects). **Please note:** in practice, **this option is rarely useful** and should not be confused with the more common **Default Reporter**. You can only specify one catch email address and one issue type per mail handler.
Bulk	This option only affects 'bulk' email messages whose header has either its **Precedence:** field set to **bulk** or its **Auto-Submitted** field set to **no**. Such messages would typically be sent by an automated service. When such an email message is received, the following action will be performed, based on the option you choose: a. Ignore the email and do nothing. b. Forward the email (i.e. to the address set in the **Forward Email** text field). c. Delete the email permanently.
Forward Email	If specified, then if this mail service is unable to handle an email message it receives, an email message indicating this problem will be forwarded to the email address specified in this field. **Please Note:** An SMTP mail server must be configured for this option to function correctly.

Create Users	Select this check box if you want JIRA to create new user accounts from any received email messages whose **From:** field contains an address that does not match one associated with an existing JIRA user account. This allows the creator of the email message to be notified of subsequent updates to the issue, which can be achieved by configuring the relevant project's notification scheme to notify the **Reporter** of updates. The username and email address of these newly created JIRA user accounts will be the email address specified in the **From:** field of the message. The password for the new user is randomly generated, and an email is sent to the new user informing them about their new account in JIRA. **Note:** this option is not compatible with **Default Reporter** field option below and as such, choosing the **Create Users** option will hide the **Default Reporter** option.
Default Reporter	Specify the username of a default reporter, which will be used if the email address in the **From:** field of any received messages does not match the address associated with that of an existing JIRA user — for example, a JIRA username such as `emailed-reporter` **Please Note:** • This option is not available if the **Create Users** check box is selected. • Please ensure that the user specified in this field has the **Create Issues** project permission for the relevant **Project** (specified above) as well as the **Create Comments** project permission for the other relevant projects to which this mail handler should add comments.
Notify Users	Clear this check box if you do not want JIRA to send out an email message notifying users whose accounts have been created by the **Create Users** option above. **Note:** this option only functions if the **Create Users** check box has been selected.

3. Test and save your mail handler (above).

Add a comment with the entire email body

This message handler creates a comment based on the entire body of the email message received. The issue to which the comment is added is chosen from the first issue key found in the email subject. The author of the comment is taken from the address of the email message's **From:** field.

To configure an 'Add a comment with the email body' mail handler:

1. If you have not already done so, begin configuring your mail handler (above).
2. On the **Create a new issue or add a comment to an existing issue** dialog box, complete the following fields/options:

Catch Email Address	If specified, only email messages whose **To:**, **Cc:**, **Bcc:** lines contain the recipient specified in this field will be processed — for example, `issues@mycompany.com` Upon specifying an address here, all email messages whose **To:**, **Cc:**, **Bcc:** lines contain addresses other than the **Catch Email Address** are ignored. This is useful if you have multiple aliases for the same mail account (e.g. `foo-support@example-co.com` and `bar-support@example-co.com` aliases for `support@example-co.com`) for multiple mail services (e.g. each one to create issues in separate JIRA projects). **Please note:** in practice, **this option is rarely useful** and should not be confused with the more common **Default Reporter**. You can only specify one catch email address and one issue type per mail handler.
Bulk	This option only affects 'bulk' email messages whose header has either its **Precedence:** field set to **bulk** or its **Auto-Submitted** field set to **no**. Such messages would typically be sent by an automated service. When such an email message is received, the following action will be performed, based on the option you choose: a. Ignore the email and do nothing. b. Forward the email (i.e. to the address set in the **Forward Email** text field). c. Delete the email permanently.
Forward Email	If specified, then if this mail service is unable to handle an email message it receives, an email message indicating this problem will be forwarded to the email address specified in this field. **Please Note:** An SMTP mail server must be configured for this option to function correctly.

Create Users	Select this check box if you want JIRA to create new user accounts from any received email messages whose **From:** field contains an address that does not match one associated with an existing JIRA user account. This allows the creator of the email message to be notified of subsequent updates to the issue, which can be achieved by configuring the relevant project's notification scheme to notify the **Reporter** of updates. The username and email address of these newly created JIRA user accounts will be the email address specified in the **From:** field of the message. The password for the new user is randomly generated, and an email is sent to the new user informing them about their new account in JIRA. **Note:** this option is not compatible with **Default Reporter** field option below and as such, choosing the **Create Users** option will hide the **Default Reporter** option.
Default Reporter	Specify the username of a default reporter, which will be used if the email address in the **From:** field of any received messages does not match the address associated with that of an existing JIRA user — for example, a JIRA username such as `emailed-reporter` **Please Note:** • This option is not available if the **Create Users** check box is selected. • Please ensure that the user specified in this field has the **Create Issues** project permission for the relevant **Project** (specified above) as well as the **Create Comments** project permission for the other relevant projects to which this mail handler should add comments.
Notify Users	Clear this check box if you do not want JIRA to send out an email message notifying users whose accounts have been created by the **Create Users** option above. **Note:** this option only functions if the **Create Users** check box has been selected.

3. Test and save your mail handler (above).

Create a new issue from each email message

This message handler creates a new issue for each incoming message.

To configure an 'Create a new issue from each email message' mail handler:

1. If you have not already done so, begin configuring your mail handler (above).
2. On the **Create a new issue or add a comment to an existing issue** dialog box, complete the following fields/options:

Project	Specify the project key of the default project to which new issues are created by this handler — for example, `JRA`. **Please Note:** • This field is only relevant for issue creation, not for issue commenting. • If an email message contains an issue key in its subject line and that issue key exists in your JIRA installation, the handler will add the email message content as a comment on the issue, regardless of which project the issue is in.
Issue Type	Choose the default issue type for new issues.
Catch Email Address	If specified, only email messages whose **To:**, **Cc:**, **Bcc:** lines contain the recipient specified in this field will be processed — for example, `issues@mycompany.com` Upon specifying an address here, all email messages whose **To:**, **Cc:**, **Bcc:** lines contain addresses other than the **Catch Email Address** are ignored. This is useful if you have multiple aliases for the same mail account (e.g. `foo-support@example-co.com` and `bar-support@example-co.com` aliases for `support@example-co.com`) for multiple mail services (e.g. each one to create issues in separate JIRA projects). **Please note:** in practice, **this option is rarely useful** and should not be confused with the more common **Default Reporter**. You can only specify one catch email address and one issue type per mail handler.
Bulk	This option only affects 'bulk' email messages whose header has either its **Precedence:** field set to **bulk** or its **Auto-Submitted** field set to **no**. Such messages would typically be sent by an automated service. When such an email message is received, the following action will be performed, based on the option you choose: a. Ignore the email and do nothing. b. Forward the email (i.e. to the address set in the **Forward Email** text field). c. Delete the email permanently.

Forward Email	If specified, then if this mail service is unable to handle an email message it receives, an email message indicating this problem will be forwarded to the email address specified in this field. **Please Note:** An SMTP mail server must be configured for this option to function correctly.
Create Users	Select this check box if you want JIRA to create new user accounts from any received email messages whose **From:** field contains an address that does not match one associated with an existing JIRA user account. This allows the creator of the email message to be notified of subsequent updates to the issue, which can be achieved by configuring the relevant project's notification scheme to notify the **Reporter** of updates. The username and email address of these newly created JIRA user accounts will be the email address specified in the **From:** field of the message. The password for the new user is randomly generated, and an email is sent to the new user informing them about their new account in JIRA. **Note:** this option is not compatible with **Default Reporter** field option below and as such, choosing the **Create Users** option will hide the **Default Reporter** option.
Default Reporter	Specify the username of a default reporter, which will be used if the email address in the **From:** field of any received messages does not match the address associated with that of an existing JIRA user — for example, a JIRA username such as `emailed-reporter` **Please Note:** • This option is not available if the **Create Users** check box is selected. • Please ensure that the user specified in this field has the **Create Issues** project permission for the relevant **Project** (specified above) as well as the **Create Comments** project permission for the other relevant projects to which this mail handler should add comments. • When an issue is created and this option is specified, the email message's **From:** field address is appended in a brief message at the end of the issue's **Description** field, so that the sender can be identified.
Notify Users	Clear this check box if you do not want JIRA to send out an email message notifying users whose accounts have been created by the **Create Users** option above. **Note:** this option only functions if the **Create Users** check box has been selected.
CC Assignee	Select this check box if you want JIRA to automatically assign the issue created to a JIRA user: • Who's email address (registered with their JIRA account) matches the first matching address encountered in the **To:**, then **Cc:** and then **Bcc:** field of the email message received. • Who also has the **Assignable User** project permission for the relevant **Project** (specified above).
CC Watchers	Select this check box if you want JIRA to automatically add JIRA users to the issue created, where those users' email addresses (registered with their JIRA accounts) match addresses encountered in the **To:, Cc:** or **Bcc:** fields of the email message received. Please note that when an issue is created, new JIRA users created by the **Create Users** option (above) *cannot also be added* to the issue's watchers list by this **CC Watchers** option. JIRA users must *already* exist in JIRA's userbase, and must have an email address.

3. Test and save your mail handler (above).

Add a comment before a specified marker or separator in the email body

This message handler creates a comment from the body of an email message - but ignores any part of the body past a marker or separator that matches a specified regular expression (regex).

For mail systems like Lotus Notes and Outlook, the core content of an email message is separated from other (e.g. replied or forwarded) content in the body by some predictable text string like '`---- Original Message ----`' or '`Extranet\n email.address/DOM/REG/CONT/CORP@CORPMAIL`'. Hence, use this message handler, which can take any valid regex, to filter core from extraneous content from various different mail systems.

Also note that the issue to which the comment is added is chosen from the first issue key found in the email subject.

The **Add a comment before a specified marker or separator in the email body** mail handler has the following behaviour with respect to received email messages:

- If the regex pattern (specified in the mail handler) is found, the text in the email message body before the first regex pattern match is used for the comment and the remainder of the body is discarded.
- If the regex pattern (specified in the mail handler) is not found, the entire text in the email message body is used for the comment.
- If no regex pattern is specified in the mail handler, the entire text in the email message body is used for the comment.
- If the regex expression specified in the mail handler is erroneous, the entire text in the email message body is used for the comment.

To configure an 'Add a comment before a specified marker or separator in the email body' mail handler:

1. If you have not already done so, begin configuring your mail handler (above).
2. On the **Create a new issue or add a comment to an existing issue** dialog box, complete the following fields/options:

Split Regex	Specify a regular expression matching the text that separates the content of the email message mail body from other (replied or forwarded) content in the body.
	
	Please Note:
	• The regex must begin and end with a delimiter character, typically '`/`'.
	• Commas are not allowed in a regex, as they are used to separate each mail handler field/option when they are integrated into a JIRA service and there is not (as yet) an escape syntax.
	
	For example:
	`/----\s*Original Message\s*----/`
	
	or
	`/_____*/`

Catch Email Address	If specified, only email messages whose **To:**, **Cc:**, **Bcc:** lines contain the recipient specified in this field will be processed — for example, `issues@mycompany.com` Upon specifying an address here, all email messages whose **To:**, **Cc:**, **Bcc:** lines contain addresses other than the **Catch Email Address** are ignored. This is useful if you have multiple aliases for the same mail account (e.g. `foo-support@example-co.com` and `bar-support@example-co.com` aliases for `support@example-co.com`) for multiple mail services (e.g. each one to create issues in separate JIRA projects). **Please note:** in practice, **this option is rarely useful** and should not be confused with the more common **Default Reporter**. You can only specify one catch email address and one issue type per mail handler.
Bulk	This option only affects 'bulk' email messages whose header has either its **Precedence:** field set to **bulk** or its **Auto-Submitted** field set to **no**. Such messages would typically be sent by an automated service. When such an email message is received, the following action will be performed, based on the option you choose: a. Ignore the email and do nothing. b. Forward the email (i.e. to the address set in the **Forward Email** text field). c. Delete the email permanently.
Forward Email	If specified, then if this mail service is unable to handle an email message it receives, an email message indicating this problem will be forwarded to the email address specified in this field. **Please Note:** An SMTP mail server must be configured for this option to function correctly.
Create Users	Select this check box if you want JIRA to create new user accounts from any received email messages whose **From:** field contains an address that does not match one associated with an existing JIRA user account. This allows the creator of the email message to be notified of subsequent updates to the issue, which can be achieved by configuring the relevant project's notification scheme to notify the **Reporter** of updates. The username and email address of these newly created JIRA user accounts will be the email address specified in the **From:** field of the message. The password for the new user is randomly generated, and an email is sent to the new user informing them about their new account in JIRA. **Note:** this option is not compatible with **Default Reporter** field option below and as such, choosing the **Create Users** option will hide the **Default Reporter** option.
Default Reporter	Specify the username of a default reporter, which will be used if the email address in the **From:** field of any received messages does not match the address associated with that of an existing JIRA user — for example, a JIRA username such as `emailed-reporter` **Please Note:** • This option is not available if the **Create Users** check box is selected. • Please ensure that the user specified in this field has the **Create Issues** project permission for the relevant **Project** (specified above) as well as the **Create Comments** project permission for the other relevant projects to which this mail handler should add comments.
Notify Users	Clear this check box if you do not want JIRA to send out an email message notifying users whose accounts have been created by the **Create Users** option above. **Note:** this option only functions if the **Create Users** check box has been selected.

3. Test and save your mail handler (above).

Custom mail handlers

You can design your own message handlers to better integrate your own processes into JIRA. Such custom mail handlers configured using the standard procedure above.

For more information about creating custom mail handlers, see Adding your own email handling classes.

Issue/comment creation

The following points describe how JIRA processes each incoming email message and determines how its content gets added as either a comment to an existing issue or a new issue altogether.

- The **subject** of an email message is examined for an existing issue key:
  - If an issue key is found in the **subject**, the content of the email message's **body** is processed and added as a comment to the issue with that issue key.
  - If an issue key is NOT found in the **subject**, the **in-reply-to header** is examined:
    - If the email message is found to be a reply to another email message from which an issue was previously created, the **body** is processed and added as a comment to that issue.
    - If the email message is NOT found to be a reply, a new issue is created.

For example, an email message to a mail account `foo@example-co.com` on a POP or IMAP mail server configured against a JIRA server will be processed as follows:

- Issue Creation:
  - The **subject** of the email message will become the issue summary.
    Since all issues require a summary, each email message intended for issue creation should include a **subject**.
  - The **body** of the email message will be the issue description.
  - A bug will be created for project 'JRA' with the above information. (This is essentially based on the mail handler configuration above).
  - Any attachments to the email message will become attachments to the issue (assuming attachments have been enabled in JIRA).
    To ensure compatibility with various operating systems, any of the following characters in the filename will be replaced with an underscore character: \, /, ", %, :, $, ?, *, <, |, >.
  - If the incoming email is set to a high priority, the corresponding issue will be created with a higher priority than the default priority that is set in your JIRA system.
- Comment Creation:
  - The **body** of the email will become a comment on the issue.
  - Any attachments to the email will become attachments to the issue (assuming attachments have been enabled in JIRA).

Handy tips with mail handlers

Allowing JIRA to handle email messages from people without a JIRA user account

To allow JIRA to handle email messages sent from people without a JIRA user account:

1. Create an 'anonymous'/'dummy' mail account on your mail server/service (above).
2. Create an equivalent 'anonymous'/'dummy' JIRA user account, whose **Email** field matches the mail account you created in the previous step.
3. When configuring your mail handler(s) (above) to handle messages from this mail account, set the **Default Reporter** to this 'anonymous'/'dummy' JIRA user account.

Best practices (pre-processing JIRA email messages)

For JIRA production servers, we recommend that setting up the following email message pre-processing:

- Since JIRA mail handlers remove successfully processed email messages from your mail server, ensure that your mail is sent to a backup folder so that a record of what mail JIRA processed is available.
- If your mail folder contains replies to JIRA's email notifications, set up rules that filter out auto-replies and bounces.
  If you do not do this, there is a strong possibility of mail loops between JIRA and autoresponders like 'out of office' notifications. JIRA sets a 'Precedence:bulk' header (unless you have disabled this) and an 'Auto-Submitted' header on outgoing email, but some autoresponders ignore it.
  There is no bulletproof way of detecting whether an email is a bounce or autoreply. The following rules (in procmail format) will detect most autoreplies:

```
^From:.*mailer-daemon@
^Auto-Submitted:.auto-
^Content-Type:\ multipart/report;\ report-type=delivery-status
^Subject:\ Delivery\ Status\ Notification
^Subject:\ Undeliverable
^Subject: Returned Mail:
^From:\ System\ Administrator
^Precedence:\ auto_reply
^Subject:.*autoreply
^Subject:.*Account\ signup
```

Even with these rules, you may encounter autoreplies with nothing in the headers to distinguish it from a regular mail, In these cases you will just need to manually update the filters to exclude that sender.

- Set up a filter to catch email with huge attachments. JIRA uses the standard JavaMail library to parse email, and it quickly runs out of memory on large attachments (e.g. > 50 MB given 512 MB heap). As the un-handled mail is not deleted, it will be reprocessed (causing another OutOfMemoryError) each time the mail service runs.
  In practice this problem is rarely seen, because most mail servers are configured to not accept email with huge attachments. Unless you are sure your mail server will not pass a huge attachment on to JIRA, it is best to configure a filter to prevent JIRA encountering any huge attachments.
- Set up spam filtering rules, so JIRA does not have to process (and possibly create issues from) spam.

Troubleshooting

JIRA's **Logging & Profiling** page has configuration options for Outgoing and Incoming mail.

Whenever you create a new (or edit an existing) mail handler (above), a **Test** button is available to allow you to test your mail handler's configuration to ensure it works as expected.

A useful tip for debugging mail-related problems in JIRA is to set the `-Dmail.debug=true` property on startup. This will cause protocol-level details of JIRA's email interactions to be logged in `catalina.out` (or standard output).

Common problems

- If JIRA does not appear to be creating sending emails or creating issues and comments from email, your JIRA instance could be experiencing **OutOfMemory errors**. Please check your log files for OutOfMemory errors. If there are OutOfMemory errors, please restart JIRA and investigate the errors.
- If you find some incoming emails simply disappear, check that you have not accidentally **started a second copy of JIRA** (e.g. in a staging environment) which is downloading and deleting mails. See the Disable email sending/receiving section of the Restoring Data page for flags you should set to prevent mail being processed.
- If you receive email with **non-ASCII attachment names**, particularly from Thunderbird users, you will need to configure JavaMail to support RFC 2231-encoded attachments. See JRA-12525 for details.
- If replies by email of JIRA's notifications list JIRA's SMTP server rather than the configured handler POP account (ie, in Outlooks' 'Reply-to' functionality), the project needs to be configured to add a 'reply-to' header in outgoing notifications. This can be configured in the project view for that particular project in JIRA's Administration.
- If HTML/Rich Text formatting is not being process correctly by JIRA, this is an expected behaviour. The email comment handler was designed to do plain text conversion.

Getting help

If you cannot resolve a problem yourself, please refer to the Getting Help page.

Configuring JIRA to Receive Email from a POP or IMAP Mail Server

To enable JIRA to create comments and issues from email, you need to first configure JIRA to receive email from a POP or IMAP mail serve as described below.

> *The information on this page does not apply to **JIRA OnDemand**.*

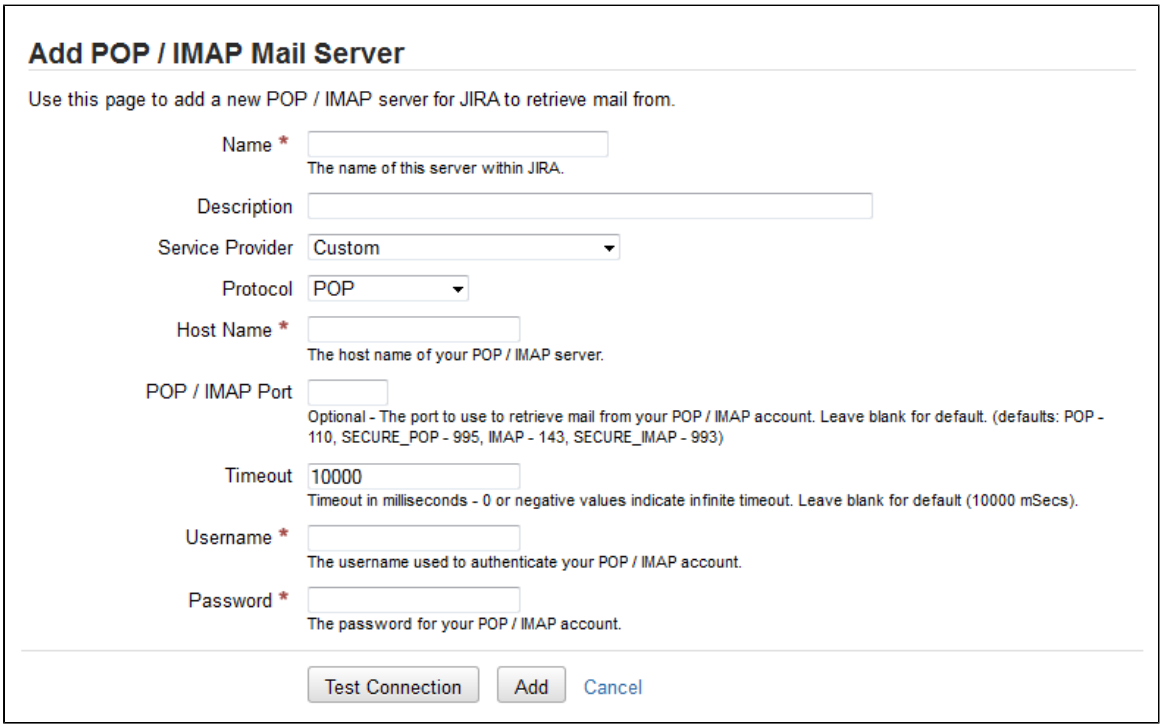
Add or edit a POP or IMAP mail server

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Mail > Incoming Mail** to open the **POP / IMAP Mail Servers** page.
   Keyboard shortcut: **g + g +** start typing **incoming mail**
3. Click either the **Configure new POP / IMAP mail server** button to define a new POP / IMAP mail server, or the **Edit** link at the right of an existing POP / IMAP mail server configuration, which will open the **Add/Update POP / IMAP Mail Server** page.
4. Complete the fields on this page as follows:

Name	Specify a short, arbitrary name to identify your POP or IMAP mail server configuration. You could possibly just specify the email address of the POP / IMAP mail server.
Description	(Optional) Specify an arbitrary description that describes the POP or IMAP mail server configuration and/or what it is used for. For example, 'Email Issue Creation/Comments for <Project>'. This description appears below the **Name** of the POP / IMAP mail server on the **POP / IMAP Mail Servers** configuration page.
Service Provider (not available when updating an existing POP / IMAP mail server)	Choose between using your own POP / IMAP mail server (i.e. **Custom**), Gmail POP / IMAP (i.e. **Google Apps Mail / Gmail [POP3 / IMAP]**) or Yahoo! POP (i.e. **Yahoo! MailPlus**) as the service provider for your POP / IMAP mail server. If you choose any of the Gmail or Yahoo! options and then switch back to **Custom**, some of the key fields in this section will automatically be populated with the relevant POP / IMAP mail server settings for these service providers.
Protocol	Choose between whether your POP / IMAP mail server is a standard (i.e. **POP or IMAP**) or a secure (i.e. **SECURE_POP or SECURE_IMAP**) one.
Host Name	Specify the hostname or IP address of your POP / IMAP mail server. Eg. `pop.yourcompany.com` or `imap.yourcompany.com`
POP / IMAP port	(Optional) The port to use to retrieve mail from your POP / IMAP account. Leave blank for default. Defaults are: POP: 110; SECURE_POP: 995; IMAP: 143; SECURE_IMAP: 993.
Timeout	(Optional) Specify the timeout period in milliseconds, which is treated as 10000 if this field is left blank. Specifying 0 or a negative value here will result in JIRA waiting indefinitely for the POP / IMAP server to respond.
Username	The username used to authenticate your POP / IMAP account.
Password	The password for your POP / IMAP account. When editing an existing POP / IMAP mail server, select the **Change Password** check box to access and change this field.

5. *(Optional)* Click the **Test Connection** button to check that JIRA can communicate with the POP / IMAP mail server you just configured.
6. Click the **Add** (or **Update**) button to save the POP / IMAP mail server configuration.

Screenshot: Add/Update POP / IMAP Mail Server



POP / IMAP over SSL

You can encrypt email communications between JIRA and your mail server via SSL, provided your mail server supports SSL.

Firstly, you will need to **import the mail server certificate** into a Java keystore. The process is described on the Connecting to SSL Services page.

Important Note: Without importing the certificate, JIRA will not be able to communicate with your mail server.

Using Gmail as a JIRA Mail Server

This page describes how to use a Gmail account as either an SMTP mail server to send notifications from JIRA or a POP3 mail server to receive email messages that create JIRA issues or comments, or both.

*The information on this page does not apply to **JIRA OnDemand**.*

Configuring JIRA to use Gmail as an SMTP mail server

1. Shut down JIRA.
2. **Move (not copy)** the 'activation' and 'mail' JAR files from the from the `<jira-application-dir>/WEB-INF/lib/` subdirectory of the JIRA Installation Directory to the `/common/lib` (Tomcat 5.5) or `/lib` (Tomcat 6) subdirectory of the JIRA Installation Directory (for 'recommended' distributions of JIRA) or the `lib/` subdirectory of the application server running JIRA.
   For example, on a *nix-based system, at a shell prompt, change directory into the JIRA Installation Directory (of a 'recommended' distribution of JIRA) and enter the following:

```
mv atlassian-jira/WEB-INF/lib/activation-1.1.1.jar lib/; mv
atlassian-jira/WEB-INF/lib/mail-1.4.1.jar lib/
```

3. Add Gmail as a JNDI resource within the within the `<Context/>` elements of the `/conf/server.xml` file. Change your username and password to those required to authenticate against your Gmail account:

```
<Resource name="mail/GmailSmtpServer"
    auth="Container"
    type="javax.mail.Session"
    mail.smtp.host="smtp.gmail.com"
    mail.smtp.port="465"
    mail.smtp.auth="true"
    mail.smtp.user="myusername@gmail.com"
    password="mypassword"
    mail.smtp.starttls.enable="true"
    mail.smtp.socketFactory.class="javax.net.ssl.SSLSocketFactory"
/>
```

4. If you are not using the built in cacerts file, you will need to add Gmail as a secure server. (Most default configurations can skip this step).

   Click here to expand...

   a. Download OpenSSL:
      - **Linux**: http://www.openssl.org/
      - **Windows**: http://gnuwin32.sourceforge.net/packages/openssl.htm

   b. Import the SSL certificate from Gmail:

      - **For Windows**: double-click the openssl file from the directory that gets installed. Run

```
s_client -connect smtp.gmail.com:465
```

      - **For Linux**: run:

```
openssl s_client -connect smtp.gmail.com:465
```

   c. From the output, you want only the alphanumeric string between the *lines* which say 'BEGIN CERTIFICATE' and 'END CERTIFICATE' (inclusive). Copy the results into a file called gmail.cert using your favorite text editor.

```
   i:/C=US/O=Equifax/OU=Equifax Secure Certificate Authority
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDWzCCAsSgAwIBAgIKFMs0nQADAAASjjANBgkqhkiG9w0BAQUFADBGMQswCQYD
VQQGEwJVUzETMBEGA1UEChMKR29vZ2xlIEluYzEiMCAGA1UEAxMZR29vZ2xlIElu
dGVybmV0IEF1dGhvcml0eTAeFw0xMDA0MjIyMDAyNDVaFw0xMTA0MjIyMDEyNDVa
MGgxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1N
b3VudGFpbiBWaWV3MRMwEQYDVQQKEwpHb29nbGUgSW5jMRcwFQYDVQQDEw5zbXRw
LmdtYWlsLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAz4B46NG2l4Bk
DdccddAuuOssQ4ZxCHoyj+ErdlEqZurVCoGQzAkhZTkzHrmIgYmM6roTbkF/zEaa
2ROe9s4VoL4OgUnoyB0u1KbwrG5PHBbsMdk0r6jMmhh+MORFVYgQrFxJnu8GnHiG
W4QhmKVytu0FclYE+F1gOOx5qIgTVCMCAwEAAaOCASswggEoMB0GA1UdDgQWBBS1
+KKmwdxtkpKkJgvwhZztqe0uszAfBgNVHSMEGDAWgBS/wDDr9UMRPme6npH7/Gra
42sSJDBbBgNVHR8EVDBSMFCgTqBMhkpodHRwOi8vd3d3LmdzdGF0aWMuY29tL0dv
b2dsZUludGVybmV0QXV0aG9yaXR5L0dvb2dsZUludGVybmV0QXV0aG9yaXR5LmNy
bDBmBggrBgEFBQcBAQRaMFgwVgYIKwYBBQUHMAKGSmh0dHA6Ly93d3cuZ3N0YXRp
Yy5jb20vR29vZ2xlSW50ZXJuZXRBdXRob3JpdHkvR29vZ2xlSW50ZXJuZXRBdXRo
b3JpdHkuY3J0MCEGCSsGAQQBgjcUAgQUHhIAVwBlAGIAUwBlAHIAdgBlAHIwDQYJ
KoZIhvcNAQEFBQADgYEAUc39Y22ucA2N4mmDdPMC8DKbrRGlGMpC294mfh3yCkJa
YRWjodFQSgCtFyKhTZ9roFpSEL0D4QxAtLSwtC0bSaSVusfAW3/CFgG/Cya724ic
9UoEosoG8CK+eGoGQuEsZ0xR86J5jOs2H+4QPHFCn594V7ZySzdeQi93YV31RBE=
-----END CERTIFICATE-----
subject=/C=US/ST=California/L=Mountain View/O=Google Inc/CN=smtp.gmail.com
issuer=/C=US/O=Google Inc/CN=Google Internet Authority
---
```

d. Exit the openssl prompt and return to your Java installation's `bin` directory. Import the cert into your keystore:

- **For Windows**:

```
keytool -import -alias smtp.gmail.com -keystore
"%JAVA_HOME%/jre/lib/security/cacerts" -file C:\path\to\gmail.cert
```

Tip:

- "A keystore is created whenever you use a -genkey, -import, or -identitydb command to add data to a keystore that doesn't yet exist. More specifically, if you specify, in the -keystore option, a keystore that doesn't yet exist, that keystore will be created. If you don't specify a -keystore option, the default keystore is a file named .keystore in your home directory. If that file does not yet exist, it will be created."
  From Sun´s Documentation on Keytool

- **For Linux:**

```
sudo keytool -import -alias smtp.gmail.com -keystore
$JAVA_HOME/jre/lib/security/cacerts -file /path/to/gmail.cert
```

Tip: The default keystore password is `changeit`

5. Restart JIRA so that JIRA will acknowledge the JNDI location you defined above.
6. Follow the instructions in Configuring JIRA's SMTP Mail Server to Send Notifications to configure JIRA's SMTP mail server and at the second part of the configuration, specify in the **JNDI Location** field (if using the default example above):

```
java:comp/env/mail/GmailSmtpServer
```

Configuring Gmail for Incoming Mail (POP)

To use Gmail, for example, as a create and comment mail handler:

If you did not import the SSL certificate from Gmail during configuration of Gmail as an SMTP mail server, refer to the instructions above.

Enable POP access in your Gmail account's settings.

Add a POP3 mail account in JIRA.

Migrating from Other Issue Trackers

When migrating from other issue trackers, you may wish to take your data with you. Depending on what issue tracker you are migrating from you are recommended to use one of the methods listed below to import data into JIRA.

1. Built-in importers

JIRA ships with the JIRA Importers Plugin pre-installed, so that you can easily import your bugs from Bugzilla, FogBugz, Mantis, Pivotal Tracker, or Trac.

- Bugzilla Version 2.0 or later of the JIRA Importers Plugin is compatible with Bugzilla 2.20 to 3.6.4. Users of older Bugzilla versions will need to first upgrade the Bugzilla database tables to a supported version, using Bugzilla's `checksetup.pl` script. The JIRA Importers Plugin requires your Bugzilla database to be MySQL or PostgreSQL.
- FogBugz Server (behind your firewall) Version 2.0 or later of the JIRA Importers Plugin is compatible with Fogbugz versions 7.3.6 to 8.2.27. The JIRA Importers Plugin requires your FogBugz database to be MySQL or MS SQL or MS SQL Express.
- FogBugz OnDemand (Hosted) Version 3.1 or later of the JIRA Importers Plugin is required.
- Mantis Version 2.0 or later of the JIRA Importers Plugin is compatible with Mantis versions 1.1.8 to 1.2.4. The JIRA Importers Plugin requires your Mantis database to be MySQL or PostgreSQL. (The JIRA Importers Plugin has also been reported to work with MS SQL, Oracle and DB2, but it has not been tested against these databases.)
- Pivotal Tracker Version 2.5 or later of the JIRA Importers Plugin is required.
- Trac Version 2.6.1 or later of the JIRA Importers Plugin is compatible with Trac version 0.12.2.

On this page:

- 1. Built-in importers
- 2. CSV Importer
- 3. Requests for Non-Supported Importers
- 4. Other Non-Supported Methods
- Other references

*The information on this page does not apply to **JIRA OnDemand**.*

2. CSV Importer

If you are migrating from a system for which JIRA does not provide a built-in importer, you may be able to import your data into JIRA via CSV format instead. Your system must be able to export your data into a CSV (comma-separated value) file. You can then import the CSV file into JIRA using JIRA's CSV importer:

- Importing Data From CSV

There is also a workaround for importing comments.

3. Requests for Non-Supported Importers

We are also tracking requests to add other systems to our built-in importers. We encourage users to vote and comment on the systems they are interested in:

- Redmine
- Rally
- Gemini
- GitHub
- BitBucket

4. Other Non-Supported Methods

There are also a few other non-supported options to get your data into JIRA:

1. Write a Jelly script that will import your data. JIRA ships with some Jelly tags that make operations like creating issues in JIRA easy.
2. Create your own scripts to move issues into JIRA,some examples are: Importing data from Trac into JIRA; Migrating Trac to JIRA; and yet another Trac 2 JIRA import.
3. JIRA ships with an RPC plugin which enables limited remote access to JIRA. It is available through REST, SOAP and XML-RPC interfaces. We recommend using the REST interface when possible as it will be our primary focus in the future. The JIRA RPC Services page provides a starting point for all your remote procedure call needs. The full source of the plugin is available and you are free to modify and the extend the source. We'd also be happy to accept code contributions to the project, as Simon Mittag has done in the past. Check out the RPC Endpoint Plugin Module for more information.

4. It is possible to use whatever tools you feel comfortable with, to import the data directly into JIRA's database. JIRA's database schema is described in XML format in the `WEB-INF/classes/entitydefs/entitymodel.xml` file under the JIRA web application. When using this approach please take care to maintain database integrity.
5. Finally as a last resort our built-in importer can be extended to support other systems, there is a very limited starting guide for those

interested in taking this avenue.

Other references

- Commercial migrations by Atlassian Experts. A number of partners (Consulting Toolsmiths and others) have provided custom migrations from Remedy, TeamTrack, ClearQuest, GNATS and Bugzilla in the past.
- Ask for help on the JIRA Development Forum.
- ClearQuest Import Forums Discussion
- Migrating Unfuddle tickets to JIRA
- Comparison of JIRA with other issue trackers

Importing Data from Bugzilla

About importing from Bugzilla

JIRA ships with the JIRA Importers Pluginpre-installed, so that you can easily import your bugs from Bugzilla.Version 2.0 or later of the JIRA Importers Plugin is compatible with Bugzilla 2.20 to 3.6.4. Users of older Bugzilla versions will need to first upgrade the Bugzilla database tables to a supported version, using Bugzilla's `checksetup.pl` script. The JIRA Importers Plugin requires your Bugzilla database to be MySQL or PostgreSQL. For the very latest version of the JIRA Importers Plugin, please visit plugins.atlassian.com.

Please Note: JIRA's character encoding is set to UTF-8 by default. If, however, your JIRA installation's character encoding is set to something other than UTF-8, you may encounter problems with importing data from Bugzilla. For more information, please refer to JIM-5. Importing Bugzilla data into a non-UTF-8 JIRA installation is not supported.

The import process consists of:

1. **Running the Import Wizard to configure the import process and save it into a configuration file.** (Note that you can edit your configuration file later by by re-running the Import Wizard and providing the name of your existing configuration file.)
2. **Configuring the connection from your JIRA server to the Bugzilla database.**
3. **Running the import using your saved configuration file.** You will typically do this several times (e.g. once or more for tests, once for production).

These steps are described in more detail below.

On this page:

- About importing from Bugzilla
- How to import from Bugzilla

*The information on this page does not apply to **JIRA OnDemand**.*

During the import process, the following data is copied from the Bugzilla database into JIRA:

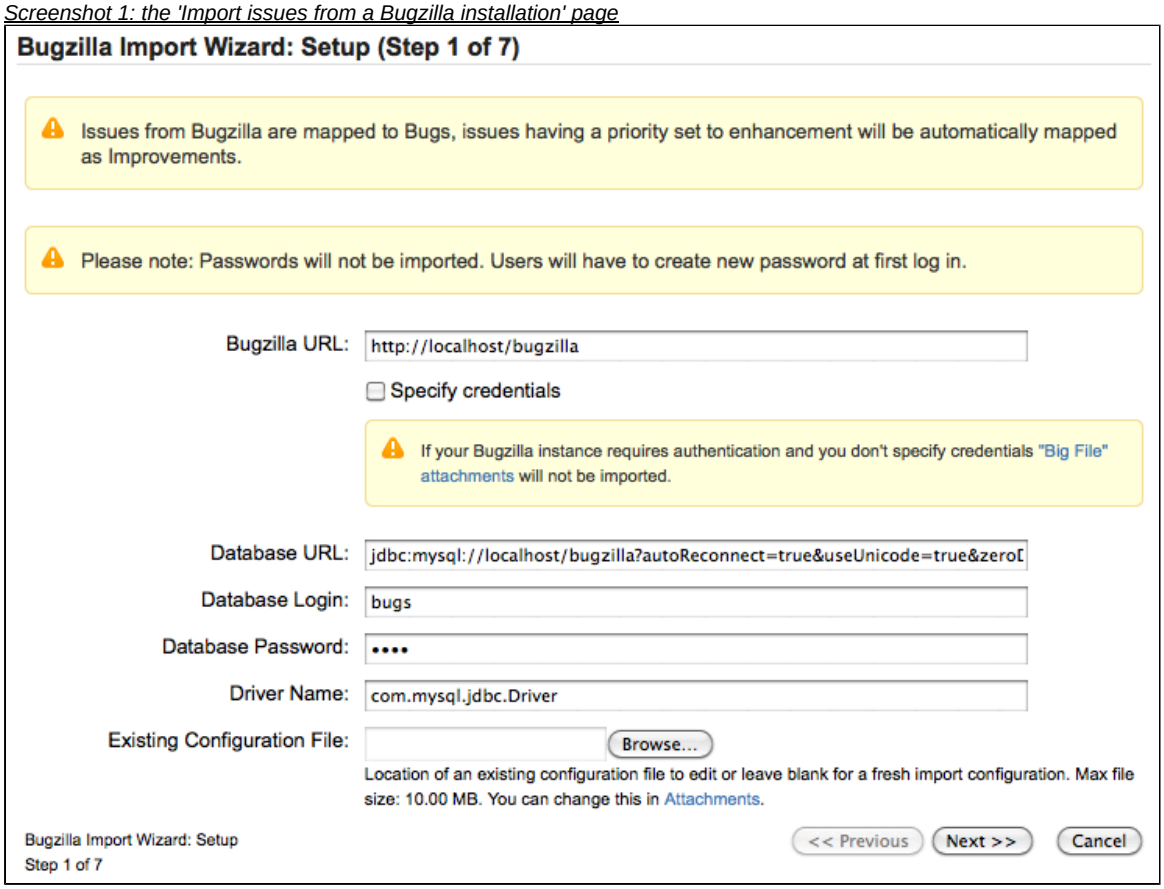
In Bugzilla	In JIRA	Import Notes
Product	Project	Bugzilla data is imported on a per-project basis. You can either specify an existing JIRA project as the target, or the importer will automatically create a project(s) for you at time of import. (For more information about JIRA projects, please see Defining a Project.)
External Project	Project Category	
Component	Component	You can choose to have the importer automatically create your Bugzilla component(s) in JIRA, or choose to have bugs imported into no component in JIRA.
Milestone	Fix Version	Versions are imported from Bugzilla (if you choose), and are set to the Un-Released and Un-Archived state.
Bug	Issue	Every Bugzilla bug becomes a JIRA issue of type 'Bug', with one exception: a Bugzilla issue with severity 'Enhancement' becomes a JIRA issue of type 'Improvement' and priority 'Major'.

ID	Bugzilla ID	Each imported issue will be given a new JIRA ID, and the old Bugzilla ID will be saved into a JIRA custom field called 'Bugzilla ID'. This custom field is searchable, so you can search for JIRA issues by their old Bugzilla ID. If you don't need this custom field, delete it or 'hide' it (as described in Specifying Field Behaviour).
Summary	Summary	
Description	Description	
Comments	Comments	
Attachments	Attachments	Attachments are extracted from the Bugzilla database and saved to disk. To specify the location on disk, see Configuring File Attachments.
Priority	Priority (or a custom field)	You can choose to map one of either the Bugzilla Priority field or the Bugzilla Severity field (see below) to the built-in JIRA Priority field, and the other to a custom field. (Alternatively, you can choose to map both the Bugzilla Priority field and the Bugzilla Severity field to JIRA custom fields.) When importing into the JIRA Priority field, you can configure mapping of specific Bugzilla values to specific JIRA values.
Severity	Priority (or a custom field)	You can choose to map one of either the Bugzilla Priority field (see above) or the Bugzilla Severity field to the built-in JIRA Priority field, and the other to a custom field. (Alternatively, you can choose to map both the Bugzilla Priority field and the Bugzilla Severity field to JIRA custom fields.) When importing into the JIRA Priority field, you can configure mapping of specific Bugzilla values to specific JIRA values.
Status	Status	You can configure mapping of specific Bugzilla values to specific JIRA values. • The JIRA '**Status**' field is integral to JIRA workflow. To learn more, please see What is Workflow and Configuring Workflow.
Resolution	Resolution	You can configure mapping of specific Bugzilla values to specific JIRA values.
Duplicates Depends on Blocks	Link	You can configure mapping of specific Bugzilla link types to JIRA link types. • In JIRA, you can configure different types of links (please see Configuring Issue Linking).
Work History	Work Log	Each Bugzilla worklog report will appear in JIRA as a separate worklog entry.
Estimated	Original Estimate	See Configuring Time Tracking.
Remaining	Remaining Estimate	See Configuring Time Tracking.
Logged	Time Spent	See Configuring Time Tracking.
Votes	Voters	If a user has voted one or more times for a Bugzilla issue, a JIRA vote is stored for that user.
CC List	Watchers	

User	User	You can choose to have the importer automatically create JIRA users for any Bugzilla users who do not already exist in JIRA. • Users who interacted with the Bugzilla system will be created as active accounts in JIRA. Other users will be imported into a special group called "bugzilla-import-unused-users" and will be deactivated. • Passwords from Bugzilla are not imported for v2.16+ of Bugzilla (as they are hashed in the database). Users from Bugzilla will need to get their passwords emailed to them the first time they log into JIRA. • Users with no real name stored in Bugzilla will get the portion of their email address (login name) before the "@" character as their Full Name in JIRA. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import (this way, votes etc can be imported correctly). • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
Status Whiteboard	Status Whiteboard	A JIRA custom field called 'Status Whiteboard' will be created.
Other fields	Custom fields	If your Bugzilla system contains any custom fields, you can choose to map them to specific JIRA custom field(s). If your custom fields don't yet exist in JIRA, the importer can automatically create them for you.

How to import from Bugzilla

1. Before you begin, please backup your JIRA data.
2. In your Bugzilla system, run the Bugzilla 'Sanity Check' to ensure your data is error-free.
3. Log in as a user with the **'JIRA Administrators'** global permission.
4. Select **'Administration' > 'System' > 'Import & Export' > 'External System Import' > 'Bugzilla'**.
5. The '**Import issues from a Bugzilla installation**' page will be displayed:
   Screenshot 1: the 'Import issues from a Bugzilla installation' page



6. If you don't yet have a configuration file (or if you want to create a new one, or update the one you have), leave the '**Existing Configuration File**' field blank and click '**Next**' to create a new Configuration File:
   - **Project Key Mappings:** For each Bugzilla project that you wish to import into JIRA, select the JIRA '**Project category**' (if applicable) and the JIRA '**Project key**'.
     - To import into an existing JIRA project, type a project key (or project name) that already exists in JIRA.
       There is no warning or error message if you select an existing key (or existing project name with a different

key). The importer will import issues to the project specified by the key (or project name).
- To create a new JIRA project and import into it, type a project key that does not yet exist in JIRA.

  The JIRA '**Project key**' will be the prefix for the IDs of all issues in the given project.

- **Custom Fields:** If your Bugzilla system contains any custom fields, you can either choose the JIRA custom field to which they will be mapped, or choose to have the importer automatically create a new custom field(s) in JIRA.

  Regardless of whether you specify mapping, the importer will automatically create a JIRA custom field for each extra Bugzilla field, unless you un-check the'**Create new custom fields**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
- **Field Value Mappings:**
  - **'Priority' field** — If you don't specify mappings, the importer will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority (e.g. "Normal" in Bugzilla to newly-created "Normal" in JIRA).
  - **Usernames** — If you don't specify mapping, the importer will automatically map Bugzilla usernames to JIRA usernames (lowercase).

    Regardless of whether you specify mapping, JIRA will automatically create usernames for missing users unless you un-check the'**Create new users**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
  - **'Status' field** — It is mandatory to map the Bugzilla '**bug_status**' field to specific values of the JIRA '**Status**' field, as the JIRA '**Status**' field is integral to JIRA workflow (to learn more, please see What is Workflow and Configuring Workflow).
  - **'Resolution' field** — If you don't specify mapping, the importer will create corresponding Resolutions in JIRA instead of using the existing ones.
- **Link Types:** To learn more about JIRA link types, please see Configuring Issue Linking.

7. Configure the connection details to your Bugzilla database as follows:
   - **Bugzilla URL** — the URL of your Bugzilla instance.
   - **Specify credentials** — tick this box to show/hide the **Bugzilla Login** and **Bugzilla Password** fields. Note that if your Bugzilla instance requires authentication and you don't specify credentials, Bugzilla "Big File" attachments will not be imported.
   - **Bugzilla Login** — the Bugzilla user to use during the import.
   - **Bugzilla Password** — the password of the Bugzilla user to use during the import.
   - **Database URL**— the location of the Bugzilla My SQL database server. Enter this in the following format:

     ```
     jdbc:mysql:host[:port]/databasename?parameters
     ```

     - **host** — the server hosting the Bugzilla MySQL database. You can specify which incoming TCP connections to allow by using the **port** parameter (defaults to 3306).
     - **databasename** — the name of the Bugzilla MySQL database (usually 'bugs').

       The database name, username and user password can usually be found in the 'localconfig' file in Bugzilla's root directory, or in /etc/bugzilla/.
   - **Database Login** — the database user that JIRA will use to access the Bugzilla database.

     Ensure that this account can access your Bugzilla database from the machine where JIRA is running.
   - **Database Password** — the password of the database user that JIRA will use to access the Bugzilla database.
   - **Driver Name**— The fully qualified name of the base class of the JDBC driver to use.:
     - "com.mysql.jdbc.Driver" for MySQL.
     - "org.postgresql.Driver" for PostgreSQL.
8. The '**Import Data**' page will then be displayed wit the following options:
   - '**Create new users**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new usernames for Bugzilla users who do not already exist in JIRA.
   - '**Create new versions**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new versions for Bugzilla '**milestones**' which do not already exist in JIRA.
   - '**Create new components**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new components for Bugzilla components which do not already exist in JIRA.
   - '**Create new custom fields**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create custom fields for Bugzilla fields which do not have a corresponding field in JIRA.
   - '**Maximum issues and failues**' — If you wish, specify a maximum number of failed issues after which the importer will stop. If you want the import to continue regardless of any failures, leave this field blank. If your Bugzilla instance has a large number of issues, it's generally a good idea to run first the importer on a limited number of issues (e.g. 100), then manually inspect the imported issues to confirm whether your configuration file was specified correctly. When the results are satisfactory, you can run the import with no limit.
9. The importer will display updates as the import progresses, then a success message when the import is complete. You can download the import log if you wish.

Congratulations, you have successfully imported your Bugzilla projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Importing Data From A FogBugz Server

JIRA ships with the JIRA Importers Pluginpre-installed, so that you can easily import your bugs from FogBugz.Version 2.0 or later of the JIRA Importers Plugin is compatible with Fogbugz versions 7.3.6 to 8.2.27. The JIRA Importers Plugin requires your FogBugz database to be MySQL or MS SQL or MS SQL Express. For the very latest version of the JIRA Importers Plugin, please visit plugins.atlassian.com.

These instructions refer to a FogBugz server behind your firewall. To import from a FogBugz OnDemand (Hosted) instance please follow the instructions for here.

The import process consists of:

1. **Running the Import Wizard to configure the import process and save it into a configuration file.** (Note that you can edit your configuration file later by by re-running the Import Wizard and providing the name of your existing configuration file.)
2. **Configuring the connection from your JIRA server to the FogBugz database.**
3. **Running the import using your saved configuration file.** You will typically do this several times (e.g. once or more for tests, once for production).

These steps are described in more detail below.

On this page:

- How to import from a FogBugz Server

*The information on this page does not apply to **JIRA OnDemand**.*

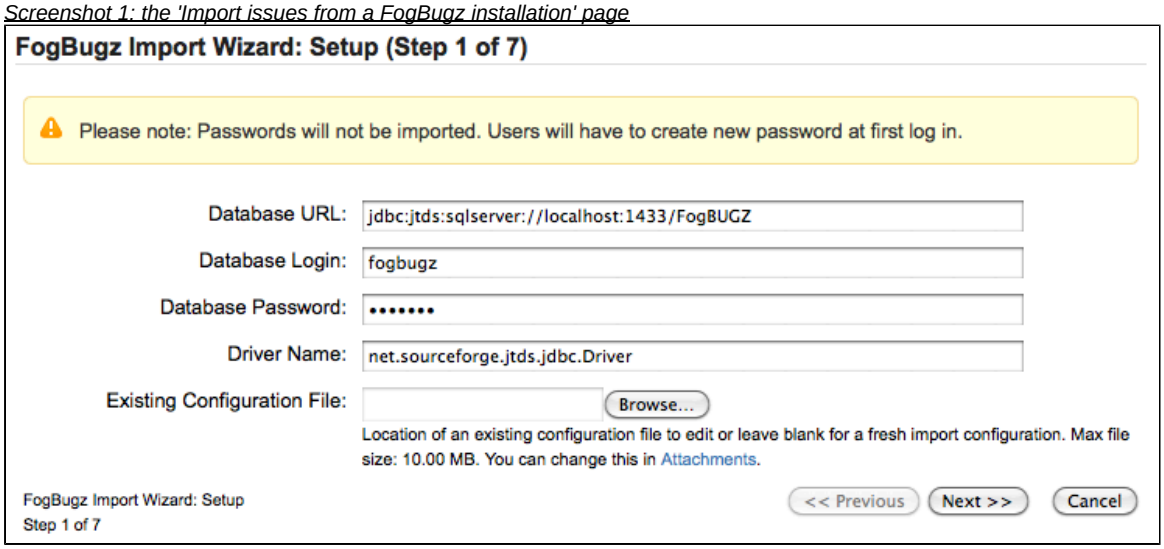
During the import process, the following data is copied from the FogBugz database into JIRA:

In FogBugz	In JIRA	Import Notes
Project	Project	FogBugz data is imported on a per-project basis. You can either specify an existing JIRA project as the target, or the importer will automatically create a project(s) for you at time of import. (For more information about JIRA projects, please see Defining a Project.)
Area	Component	You can choose to have the importer automatically create your FogBugz components in JIRA, or choose to have bugs imported into no component in JIRA.
Milestone	Fix Version	Versions are imported from FogBugz (if you choose). After importing, you can manually set appropriate versions to the Released state in JIRA if you wish.
Case	Issue	Every FogBugz case becomes a JIRA issue.
Case ID ixBug	Bug Import ID	Each imported issue ('case') will be given a new JIRA ID, and the old FogBugz ID will be saved into a JIRA custom field called 'Bug Import ID'. This custom field is searchable, so you can search for JIRA issues by their old FogBugz ID. If you don't need this custom field, delete it or 'hide' it (as described in Specifying Field Behaviour).
Summary	Summary	
Comments	Comments	FogBugz allows for links to other issues to be automatically generated by using the format "bug issueId" or "case issue id". After import, any string matching this pattern will be rewritten to their new JIRA key. For example, a comment "Please see case 100" may be rewritten to "Please see IMP-100".
Attachments	Attachments	Attachments are extracted from the FogBugz database and saved to disk. Any e-mail issues will be parsed for attachments and the e-mail text saved as a comment. The dates and user attaching the attachments will be retained.To specify the location on disk, see Configuring File Attachments.
Category	Issue Type	You can configure mapping of specific Case Categories to specific Issue Types.
Priority	Priority	You can configure mapping of specific FogBugz values to specific JIRA values.
Status	Status	You can configure mapping of specific FogBugz values to specific JIRA values, provided you create your workflows in JIRA before running the importer. • The JIRA '**Status**' field is integral to JIRA workflow. To learn more, please see What is Workflow. • To create a **JIRA workflow**, please see Configuring Workflow. • To create a **JIRA workflow scheme** (which you can then associate with appropriate projects and Issue Types), please see Activating Workflow.
Resolution	Resolution	You can configure mapping of specific FogBugz values to specific JIRA values.
Duplicates BugRelations	Links	You can configure mapping of specific FogBugz link types to JIRA link types. • In JIRA, you can configure different types of links (please see Configuring Issue Linking).

Computer	Computer	The FogBugz Computer field is imported into a JIRA Custom Field called 'Computer'.
Customer Email	Customer Email	The FogBugz Customer Email field is imported into a JIRA Custom Field called 'Customer Email'.
User	User	You can choose to have the importer automatically create JIRA users for any FogBugz users who do not already exist in JIRA. • Users who interacted with the FogBugz system will be created as active accounts in JIRA. Other users will be imported into a special group called "fogbugz-import-unused-users" and will be deactivated. • Passwords from FogBugz are not imported (as they are hashed in the database). Users from FogBugz will need to get their passwords emailed to them the first time they log into JIRA. • Users with no real name stored in FogBugz will get the portion of their email address (login name) before the "@" character as their Full Name in JIRA. • If you don't specify any particular mappings, the user name will be created from the first letter of the first name and the last name, all in lowercase. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
Other fields	Custom fields	If your FogBugz system contains any custom fields, you can choose to map them to specific JIRA custom field(s). If your custom fields don't yet exist in JIRA, the importer can automatically create them for you. Please note that the *FogBugz Custom Field plugin* is not supported.

How to import from a FogBugz Server

1. Before you begin, please backup your JIRA data.
2. Log in to JIRA as a user with the **'JIRA Administrators'** global permission.
3. Select **'Administration' > 'System' > 'Import & Export' > 'External System Import' > 'FogBugz for Your Server'**.
4. The '**Import issues from a FogBugz installation**' page will be displayed:
   Screenshot 1: the 'Import issues from a FogBugz installation' page



5. If you don't yet have a configuration file (or if you want to create a new one, or update the one you have), leave the '**Existing Configuration File**' field blank and click '**Next**' to create a new Configuration File:
   - **Project Key Mappings:** For each FogBugz project that you wish to import into JIRA, select the JIRA '**Project category**' (if applicable) and the JIRA '**Project key**'.
     - To import into an existing JIRA project, type a project key (or project name) that already exists in JIRA.
       There is no warning or error message if you select an existing key (or existing project name with a different key). The importer will import issues to the project specified by the key (or project name).
     - To create a new JIRA project and import into it, type a project key that does not yet exist in JIRA.
       The JIRA '**Project key**' will be the prefix for the IDs of all issues in the given project.
   - **Custom Fields:** If your FogBugz system contains any custom fields, you can either choose the JIRA custom field to which they will be mapped, or choose to have the importer automatically create a new custom field(s) in JIRA.
     Regardless of whether you specify mapping, the importer will automatically create a JIRA custom field for each extra FogBugz field, unless you un-check the '**Create new custom fields**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
   - **Field Value Mappings:**
     - **'Priority' field** — If you don't specify mappings, the importer will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority (e.g. "Normal" in FogBugz to newly-created

"Normal" in JIRA).
- **Usernames** — If you don't specify mapping, the importer will automatically map FogBugz usernames to JIRA usernames (lowercase).
  Regardless of whether you specify mapping, JIRA will automatically create usernames for missing users unless you un-check the '**Create new users**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
- **'Status' field** — It is mandatory to map the FogBugz '**Status**' field to specific values of the JIRA '**Status**' field, as the JIRA '**Status**' field is integral to JIRA workflow (to learn more, please see What is Workflow and Configuring Workflow).
- **'Resolution' field** — If you don't specify mapping, the importer will create corresponding Resolutions in JIRA instead of using the existing ones.

- **Link Types:** To learn more about JIRA link types, please see Configuring Issue Linking.

6. Configure the connection details to your FogBugz database as follows:
   - **FogBugz URL** — the URL of your FogBugz instance.
   - **FogBugz Login** — the FogBugz user to use during the import.
   - **FogBugz Password** — the password of the FogBugz user to use during the import.
   - **Database URL**— the location of the FogBugz database server. Enter this in the following format:

     ```
     jdbc:mysql:host[:port]/databasename?parameters
     ```

     - **host** — the server hosting the FogBugz database. You can specify which incoming TCP connections to allow by using the **port** parameter (defaults to 3306).
     - **databasename** — the name of the FogBugz database (usually 'bugtracker').
       The database name can usually be found in the registry. Documentation is at http://bugs.movabletype.org/help/topics/setup/WindowsWhatSetupDoes.html (search for Initialize Registry Settings).
   - **Database Login** — the database user that JIRA will use to access the FogBugz database.
     Ensure that this account can access your FogBugz database from the machine where JIRA is running.
   - **Database Password** — the password of the database user that JIRA will use to access the FogBugz database.
   - **Driver Name**— The fully qualified name of the base class of the JDBC driver to use:
     - "com.mysql.jdbc.Driver" for MySQL.
     - "org.postgresql.Driver" for PostgreSQL.
7. The '**Import Data**' page will then be displayed with the following options:
   - '**Create new users**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new usernames for FogBugz users who do not already exist in JIRA.
   - '**Create new versions**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new versions for FogBugz '**backlogs**' which do not already exist in JIRA.
   - '**Create new components**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new components for FogBugz components (areas) which do not already exist in JIRA.
   - '**Create new custom fields**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create custom fields for FogBugz fields which do not have a corresponding field in JIRA.
   - '**Maximum issues and failues**' — If you wish, specify a maximum number of failed issues after which the importer will stop. If you want the import to continue regardless of any failures, leave this field blank. If your FogBugz instance has a large number of issues, it's generally a good idea to run first the importer on a limited number of issues (e.g. 100), then manually inspect the imported issues to confirm whether your configuration file was specified correctly. When the results are satisfactory, you can run the import with no limit.
8. The importer will display updates as the import progresses, then a success message when the import is complete. You can download the import log if you wish.

Congratulations, you have successfully imported your FogBugz projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Importing Data From FogBugz OnDemand

About importing from FogBugz OnDemand

JIRA ships with the JIRA Importers Pluginpre-installed, so that you can easily import your bugs from FogBugz OnDemand.Version 3.1 or later of the JIRA Importers Plugin is required.If you have an earlier version you can upgrade it via the Universal Plugin Manager.
These instructions refer to FogBugz OnDemand (Hosted). To import from a FogBugz Server behind your firewall please follow the instructions for here.

On this page:

- About importing from FogBugz OnDemand
- How to import from FogBugz OnDemand

*The information on this page does not apply to **JIRA OnDemand**.*

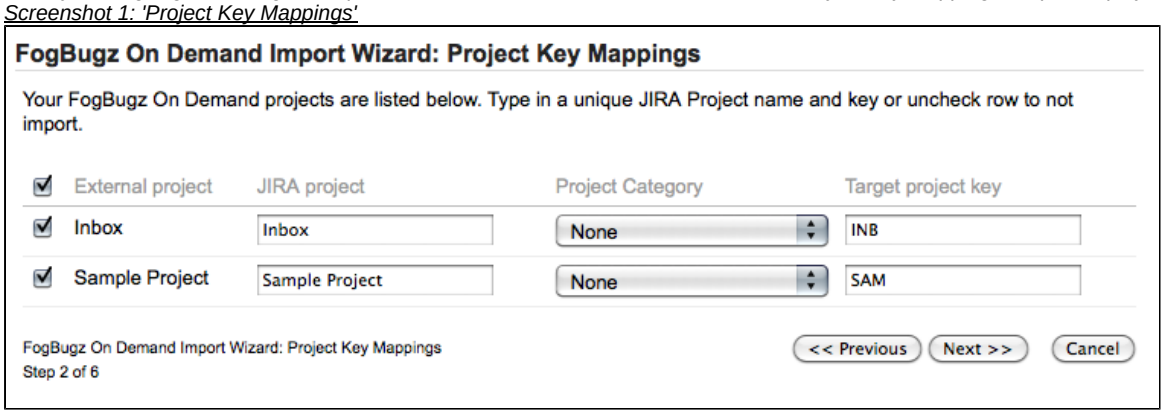
The import process converts FogBugz OnDemand data as follows:

FogBugz OnDemand	In JIRA	Import Notes
Project	Project	FogBugz data is imported on a per-project basis. You can either specify an existing JIRA project as the target, or the importer will automatically create a project(s) for you at time of import. (For more information about JIRA projects, please see Defining a Project.)
Area	Component	You can choose to have the importer automatically create your FogBugz components in JIRA, or choose to have bugs imported into no component in JIRA.
Milestone	Fix Version	Versions are imported from FogBugz (if you choose). After importing, you can manually set appropriate versions to the Released state in JIRA if you wish.
Case	Issue	Every FogBugz case becomes a JIRA issue.
Case ID ixBug	External issue ID and External issue URL	Each imported issue ('case') will be given a new JIRA ID, and the old FogBugz ID will be saved into a JIRA custom field called 'External issue ID'. This custom field is searchable, so you can search for JIRA issues by their old FogBugz ID. If you don't need this custom field, delete it or 'hide' it (as described in Specifying Field Behaviour).
Summary	Summary	
Comments	Comments	FogBugz allows for links to other issues to be automatically generated by using the format "bug issueId" or "case issue id". After import, any string matching this pattern will be rewritten to their new JIRA key. For example, a comment "Please see case 100" may be rewritten to "Please see IMP-100".
Attachments	Attachments	Attachments are extracted from the FogBugz database and saved to disk. Any e-mail issues will be parsed for attachments and the e-mail text saved as a comment. The dates and user attaching the attachments will be retained.To specify the location on disk, see Configuring File Attachments.
Category	Issue Type	You can configure mapping of specific Case Categories to specific Issue Types.
Priority	Priority	You can configure mapping of specific FogBugz values to specific JIRA values.
Status	Status	You can configure mapping of specific FogBugz values to specific JIRA values, provided you create your workflows in JIRA before running the importer. • The JIRA '**Status**' field is integral to JIRA workflow. To learn more, please see What is Workflow. • To create a **JIRA workflow**, please see Configuring Workflow. • To create a **JIRA workflow scheme** (which you can then associate with appropriate projects and Issue Types), please see Activating Workflow.
Resolution	Resolution	You can configure mapping of specific FogBugz values to specific JIRA values.
Duplicates BugRelations		They are not imported due to limitations of FogBugz Remote API
Computer	Computer	The FogBugz Computer field is imported into a JIRA Custom Field called 'Computer'.
Customer Email	Customer Email	The FogBugz Customer Email field is imported into a JIRA Custom Field called 'Customer Email'.

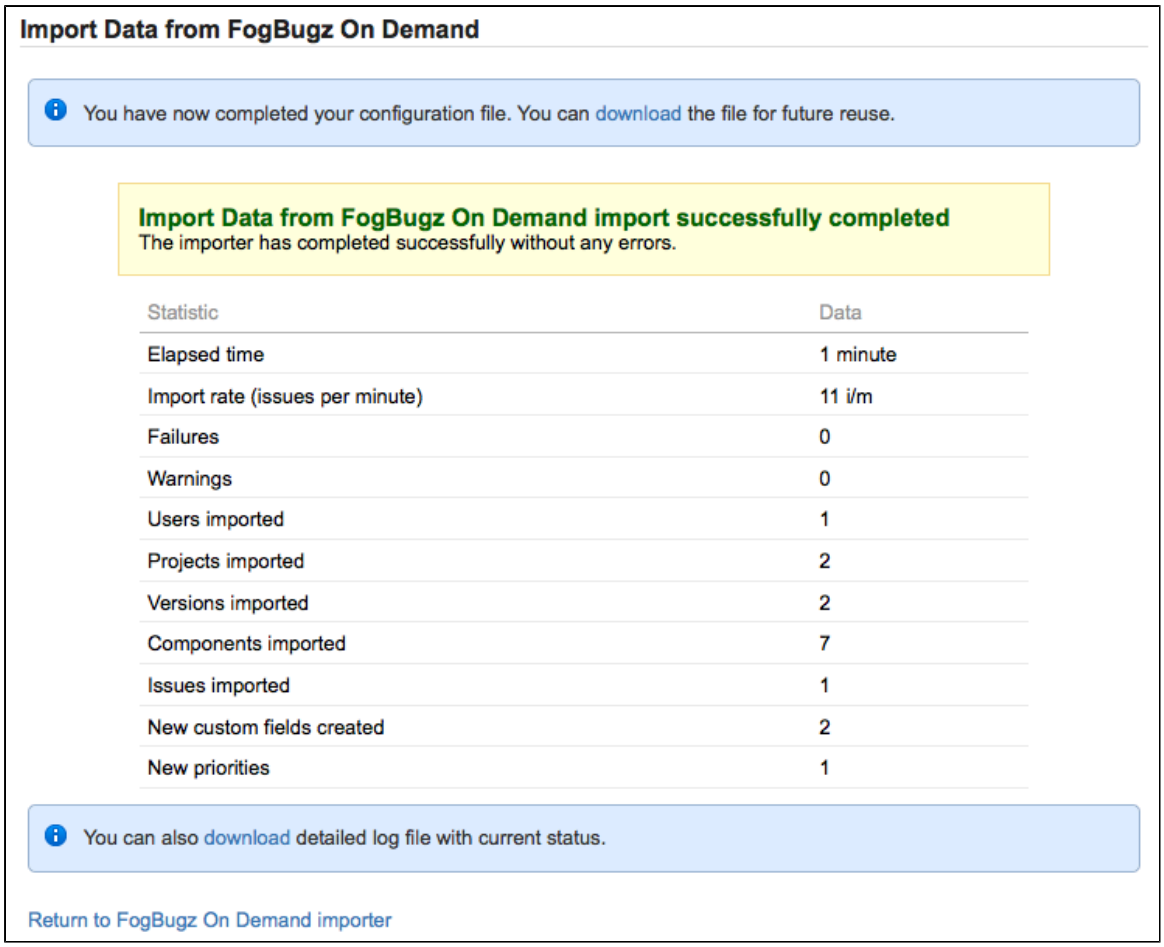
User	User	You can choose to have the importer automatically create JIRA users for any FogBugz users who do not already exist in JIRA. • Users who interacted with the FogBugz system will be created as active accounts in JIRA. Other users will be imported into a special group called "fogbugz-import-unused-users" and will be deactivated. • Passwords from FogBugz are not imported (as they are hashed in the database). Users from FogBugz will need to get their passwords emailed to them the first time they log into JIRA. • Users with no real name stored in FogBugz will get the portion of their email address (login name) before the "@" character as their Full Name in JIRA. • If you don't specify any particular mappings, the user name will be created from the first letter of the first name and the last name, all in lowercase. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
Other fields	Custom fields	If your FogBugz system contains any custom fields, you can choose to map them to specific JIRA custom field(s). If your custom fields don't yet exist in JIRA, the importer can automatically create them for you. Please note that the *FogBugz Custom Field plugin* is not supported.

How to import from FogBugz OnDemand

1. Before you begin, please back up your JIRA data. (If you have none then you can skip this step.)
2. Log in to JIRA as as a user with the **'JIRA Administrators'** global permission.
3. Select **'Administration' > 'System' > 'Import & Export' > 'External System Import' > 'FogBugz OnDemand'**.
4. Enter your FogBugz URL, login and password and then click the '**Next**' button. The 'Project Key Mappings' step is displayed:
   Screenshot 1: 'Project Key Mappings'



   - The check box of each FogBugz OnDemand project to be imported is selected. To prevent a project from being imported, clear its check box.
   - By default, a new JIRA project for each FogBugz OnDemand project you import will be created. (JIRA automatically suggests a name and key for each of these JIRA projects.)
   - If applicable, select 'Project category' for these JIRA projects.
   - To import a FogBugz OnDemand project into an existing JIRA project, type in the exact name and key of the existing JIRA project.
5. The **Field Value Mappings**page with then be displayed with the following options:
   - External fields: This will allow you to transform values of various fields during the import process.
   - Workflow Scheme: Select the JIRA workflow scheme for all projects to use or accept the default.
     - Then the importer will allow you to map values.
6. The **Issue Link Mappings** page will then be displayed. This is where you will select how to map FogBugz's Parent/Subcase links in JIRA.
7. The '**Import Data**' page will then be displayed with the following options:
   - '**Create new users**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new usernames for FogBugz users who do not already exist in JIRA.
   - '**Create new versions**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new versions for FogBugz '**backlogs**' which do not already exist in JIRA.
   - '**Create new components**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new components for FogBugz components (areas) which do not already exist in JIRA.
   - '**Create new custom fields**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create custom fields for FogBugz fields which do not have a corresponding field in JIRA.
   - '**Maximum issues and failues**' — If you wish, specify a maximum number of failed issues after which the importer will stop. If you want the import to continue regardless of any failures, leave this field blank. If your FogBugz instance has a large number of issues, it's generally a good idea to run first the importer on a limited number of issues (e.g. 100), then manually inspect the imported issues to confirm whether your configuration file was specified correctly. When the results are satisfactory, you can run the import with no limit.
8. The importer will display updates as the import progresses, then a success message when the import is complete.
   Screenshot 3: the 'Success' page



Congratulations, you have successfully imported your FogBugz OnDemand projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Importing Data From Mantis

About importing from Mantis

JIRA ships with the JIRA Importers Plugin pre-installed, so that you can import your bugs from Mantis.

Version 2.0 or later of the JIRA Importers Plugin is compatible with Mantis versions 1.1.8 to 1.2.4. The JIRA Importers Plugin requires your Mantis database to be MySQL or PostgreSQL. (The JIRA Importers Plugin has also been reported to work with MS SQL, Oracle and DB2, but it has not been tested against these databases.)

For the very latest version of the JIRA Importers Plugin, please visit plugins.atlassian.com.

The import process consists of:

1. **Running the Import Wizard to configure the import process and save it into a configuration file.** (Note that you can edit your configuration file later by by re-running the Import Wizard and providing the name of your existing configuration file.)
2. **Configuring the connection from your JIRA server to the Mantis database.**
3. **Running the import using your saved configuration file.** You will typically do this several times (e.g. once or more for tests, once for production).

These steps are described in more detail below.

On this page:

- About importing from Mantis
- How to import from Mantis

*The information on this page does not apply to **JIRA OnDemand**.*

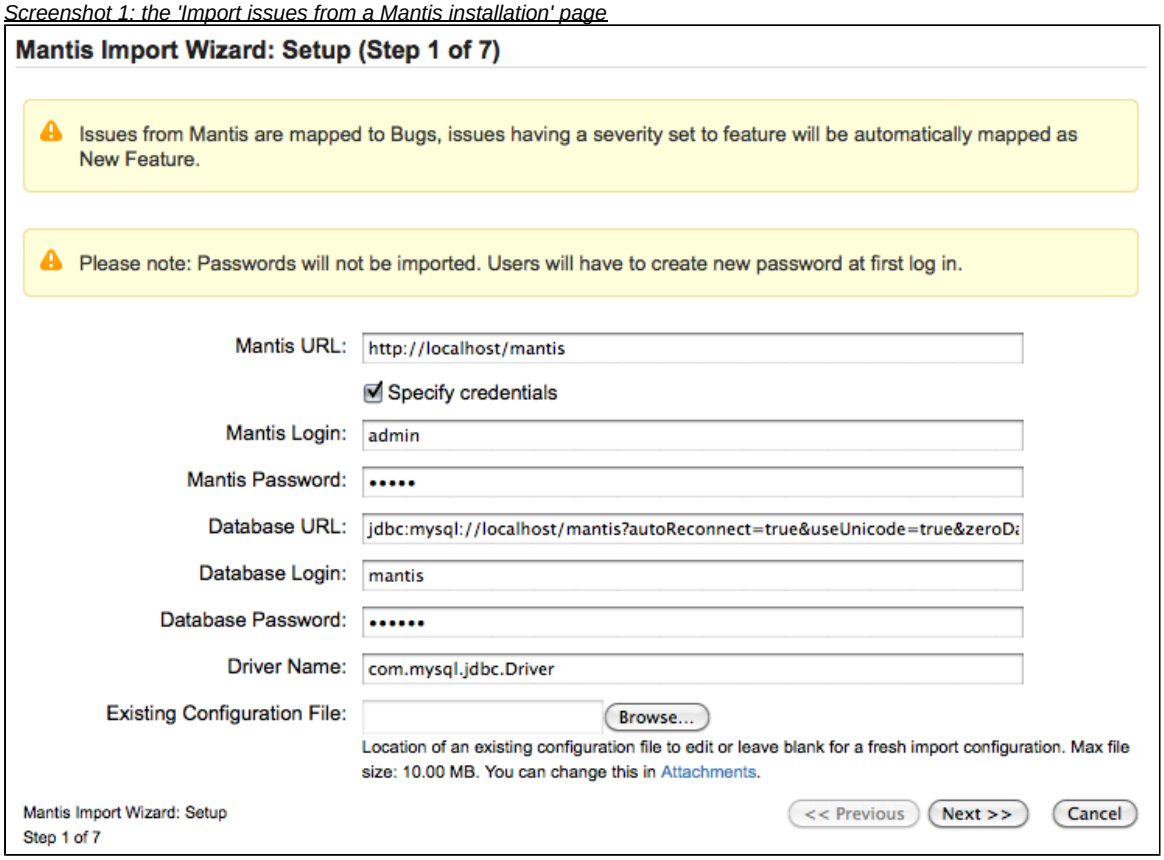
During the import process, the following data is copied from the Mantis database into JIRA:

In Mantis	In JIRA	Import Notes
Project Sub Project	Project	Mantis data is imported on a per-project basis. You can either specify an existing JIRA project as the target, or the importer will automatically create a project(s) for you at time of import. (For more information about JIRA projects, please see Defining a Project.)
Category	Component	You can choose to have the importer automatically create your Mantis components in JIRA, or choose to have bugs imported into no component in JIRA.
Version	Fix Version	Versions are imported from Mantis (if you choose). After importing, you can manually set appropriate versions to the Released state in JIRA if you wish.
Bug	Issue	Every Mantis bug becomes a JIRA issue of type 'Bug'.
ID	Bug Import ID	Each imported issue will be given a new JIRA ID, and the old Mantis ID will be saved into a JIRA custom field called 'Bug Import ID'. This custom field is searchable, so you can search for JIRA issues by their old Mantis ID. If you don't need this custom field, delete it or 'hide' it (as described in Specifying Field Behaviour).
Summary	Summary	
Description	Description	Within text, Mantis links (e.g. #1234) are converted to JIRA links (e.g. TST-123).
Comments	Comments	Within text, Mantis links (e.g. #1234) are converted to JIRA links (e.g. TST-123).
Attachments	Attachments	Attachments are extracted from the Mantis database and saved to disk. To specify the location on disk, see Configuring File Attachments.
Priority	Priority (or a custom field)	You can choose to map one of either the Mantis Priority field or the Mantis Severity field (see below) to the built-in JIRA Priority field, and the other to a custom field. (Alternatively, you can choose to map both the Mantis Priority field and the Mantis Severity field to JIRA custom fields.) When importing into the JIRA Priority field, you can configure mapping of specific Mantis values to specific JIRA values.
Severity	Priority (or a custom field)	You can choose to map one of either the Mantis Priority field (see above) or the Mantis Severity field to the built-in JIRA Priority field, and the other to a custom field. (Alternatively, you can choose to map both the Mantis Priority field and the Mantis Severity field to JIRA custom fields.) When importing into the JIRA Priority field, you can configure mapping of specific Mantis values to specific JIRA values.
Status	Status	You can configure mapping of specific Mantis values to specific JIRA values, provided you create your workflows in JIRA before running the importer. • The JIRA '**Status**' field is integral to JIRA workflow. To learn more, please see What is Workflow. • To create a **JIRA workflow**, please see Configuring Workflow. • To create a **JIRA workflow scheme** (which you can then associate with appropriate projects and Issue Types), please see Activating Workflow.
Resolution	Resolution	You can configure mapping of specific Mantis values to specific JIRA values.
Relationships	Links	You can configure mapping of specific Mantis relationship types to JIRA link types. • In JIRA, you can configure different types of links (please see Configuring Issue Linking).
CC List	Watchers	

User	User	You can choose to have the importer automatically create JIRA users for any Mantis users who do not already exist in JIRA. • Users who interacted with the Mantis system will be created as active accounts in JIRA. Other users will be imported into a special group called "mantis-import-unused-users" and will be deactivated. • Passwords from Mantis are not imported (as they are hashed in the database). Users from Mantis will need to get their passwords emailed to them the first time they log into JIRA. • Users with no real name stored in Mantis will get the portion of their email address (login name) before the "@" character as their Full Name in JIRA. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
Other fields	Custom fields	If your Mantis system contains any custom fields, you can choose to map them to specific JIRA custom field(s). If your custom fields don't yet exist in JIRA, the importer can automatically create them for you.

How to import from Mantis

1. Before you begin, please backup your JIRA data.
2. Log in to JIRA as as a user with the **'JIRA Administrators'** global permission.
3. Select **'Administration' > 'System' > 'Import & Export' > 'External System Import' > 'Mantis'**.
4. The '**Import issues from a Mantis installation**' page will be displayed:
   Screenshot 1: the 'Import issues from a Mantis installation' page



5. If you don't yet have a configuration file (or if you want to create a new one, or update the one you have), leave the '**Existing Configuration File**' field blank and click '**Next**' to create a new Configuration File:
   - **Project Key Mappings:** For each Mantis project that you wish to import into JIRA, select the JIRA '**Project category**' (if applicable) and the JIRA '**Project key**'.
     - To import into an existing JIRA project, type a project key (or project name) that already exists in JIRA.
       There is no warning or error message if you select an existing key (or existing project name with a different key). The importer will import issues to the project specified by the key (or project name).
     - To create a new JIRA project and import into it, type a project key that does not yet exist in JIRA.
       The JIRA '**Project key**' will be the prefix for the IDs of all issues in the given project.
   - **Custom Fields:** If your Mantis system contains any custom fields, you can either choose the JIRA custom field to which they will be mapped, or choose to have the importer automatically create a new custom field(s) in JIRA.
     Regardless of whether you specify mapping, the importer will automatically create a JIRA custom field for each extra

Mantis field, unless you un-check the '**Create new custom fields**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
- **Field Value Mappings:**
  - **'Priority' field** — If you don't specify mappings, the importer will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority (e.g. "Normal" in Mantis to newly-created "Normal" in JIRA).
  - **Usernames** — If you don't specify mapping, the importer will automatically map Mantis usernames to JIRA usernames (lowercase).

    Regardless of whether you specify mapping, JIRA will automatically create usernames for missing users unless you un-check the '**Create new users**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
  - **'Status' field** — It is mandatory to map the Mantis '**Status**' field to specific values of the JIRA '**Status**' field, as the JIRA '**Status**' field is integral to JIRA workflow (to learn more, please see What is Workflow and Configuring Workflow).
  - **'Resolution' field** — If you don't specify mapping, the importer will create corresponding Resolutions in JIRA instead of using the existing ones.
- **Link Types:** To learn more about JIRA link types, please see Configuring Issue Linking.

6. Configure the connection details to your Mantis database as follows:
   - **Mantis URL** — the URL of your Mantis instance.
   - **Mantis Login** — the Mantis user to use during the import.
   - **Mantis Password** — the password of the Mantis user to use during the import.
   - **Database URL**— the location of the Mantis MySQL database server. Enter this in the following format:

     ```
     jdbc:mysql:host[:port]/databasename?parameters
     ```

     - **host** — the server hosting the Mantis MySQL database. You can specify which incoming TCP connections to allow by using the **port** parameter (defaults to 3306).
     - **databasename** — the name of the Mantis MySQL database (usually 'bugtracker').

       The database name, username and user password can usually be found in the Mantis file 'config_inc.php' (the default username is "root", default password is empty). See also http://www.mantisbt.org/manual/manual.configuration.database.php
   - **Database Login** — the database user that JIRA will use to access the Mantis database.

     Ensure that this account can access your Mantis database from the machine where JIRA is running.
   - **Database Password** — the password of the database user that JIRA will use to access the Mantis database.
   - **Driver Name**— The fully qualified name of the base class of the JDBC driver to use.:
     - "com.mysql.jdbc.Driver" for MySQL.
     - "org.postgresql.Driver" for PostgreSQL.
7. The '**Import Data**' page will then be displayed with the following options:
   - '**Create new users**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new usernames for Mantis users who do not already exist in JIRA.
   - '**Create new versions**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new versions for Mantis '**milestones**' which do not already exist in JIRA.
   - '**Create new components**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new components for Mantis components (categories) which do not already exist in JIRA.
   - '**Create new custom fields**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create custom fields for Mantis fields which do not have a corresponding field in JIRA.
   - '**Maximum issues and failues**' — If you wish, specify a maximum number of failed issues after which the importer will stop. If you want the import to continue regardless of any failures, leave this field blank. If your Mantis instance has a large number of issues, it's generally a good idea to run first the importer on a limited number of issues (e.g. 100), then manually inspect the imported issues to confirm whether your configuration file was specified correctly. When the results are satisfactory, you can run the import with no limit.
8. The importer will display updates as the import progresses, then a success message when the import is complete. You can download the import log if you wish.

Congratulations, you have successfully imported your Mantis projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Importing Data From Pivotal Tracker

About importing from Pivotal Tracker

JIRA ships with the JIRA Importers Pluginpre-installed, so that you can easily import your bugs from Pivotal Tracker.Version 2.5 or later of the JIRA Importers Plugin is required.If you have an earlier version you can upgrade it via the Universal Plugin Manager.

Please make sure that you have switched on "Allow API Access" in your Pivotal Project Settings.

On this page:

- About importing from Pivotal Tracker
- How to import from Pivotal Tracker

*The information on this page does not apply to **JIRA OnDemand**.*

The import process converts Pivotal Tracker data as follows:

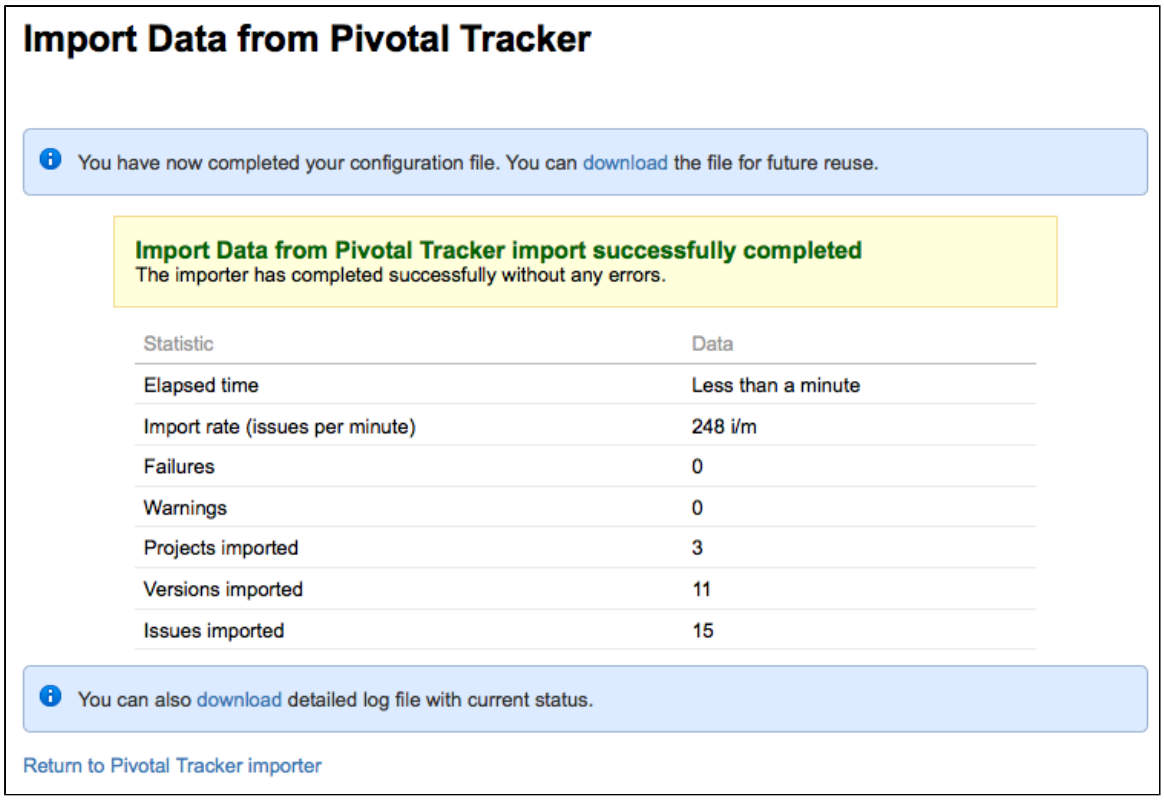
Pivotal Tracker	JIRA	Import Notes
Project	Project	Each Pivotal Tracker project is imported into a new JIRA project. You can optionally import into an existing project if you have used the importer before.
Story	Issue	Pivotal Tracker story types are recreated in JIRA.
Summary	Summary	
Comments	Comments	
Attachments	Attachments	Attachments are extracted from the Pivotal Tracker database and saved to disk. The dates and user attaching the attachments will be retained.
Status	Status	JIRA will recreate the Pivotal Tracker workflow and statuses during import.
Labels	Labels	Pivotal Tracker labels with spaces are imported with underscores (JIRA does not support spaces in labels).
Story ID	Story ID and Story URL	JIRA will create these as custom fields.
Iterations	Fix Version/s	Past iterations in Pivotal are imported as released versions in JIRA.
Story Estimates	Story Points	
Order of stories	Rank	You will need to configure this custom field in JIRA after the import. If you are using GreenHopper, you may wish to activate issue ranking. This can be done either before or after importing your Pivotal Tracker data.

Time Tracker	Time Tracking	If you use time tracking in Pivotal this data will be automatically imported into a new JIRA issue type called 'Chore' with a Summary field value of "Placeholder for imported time tracking data".
User	User	The importer will automatically create JIRA users for any Pivotal Tracker users who do not exist in JIRA. • Passwords from Pivotal Tracker are not imported (as they are hashed in the database). Users from Pivotal Tracker will need to get their passwords emailed to them. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
User Roles	Project Roles	Viewer = User ; Member = Developers ; Owner = Administrators

How to import from Pivotal Tracker

1. Before you begin, please back up your JIRA data. (If you have none then you can skip this step.)
2. Log in to JIRA as as a user with the **'JIRA Administrators'** global permission.
3. Select **'Administration' > 'System' > 'Import & Export' > 'External System Import' > 'Pivotal Tracker'**.
4. Enter your Pivotal Tracker login and password credentials and click the '**Next**' button. The 'Project Key Mappings' step is displayed:
   Screenshot 1: 'Project Key Mappings'

   - The check box of each Pivotal Tracker project to be imported is selected. To prevent a project from being imported, clear its check box.
   - By default, a new JIRA project for each Pivotal Tracker project you import will be created. (JIRA automatically suggests a name and key for each of these JIRA projects.)
   - If applicable, select 'Project category' for these JIRA projects.
   - To import a Pivotal Tracker project into an existing JIRA project, type in the exact name and key of the existing JIRA project.
     ⚠ The existing JIRA project must also use the 'PT Workflow Scheme', which is created with your first Pivotal Tracker import.
5. The importer will display updates as the import progresses, then a success message when the import is complete.
   Screenshot 3: the 'Success' page



Congratulations, you have successfully imported your Pivotal Tracker projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Importing Data From Trac

About importing from Trac

JIRA ships with the JIRA Importers Plugin pre-installed, so that you can easily import your bugs from Trac.Version 2.6.1 or later of the JIRA Importers Plugin is compatible with Trac version 0.12.2.If you have an earlier version you can upgrade it via the Universal Plugin Manager.

On this page:

- About importing from Trac
- How to import from Trac
  - First you will need to ZIP up the contents of your Trac environment
  - Import into JIRA

*The information on this page does not apply to **JIRA OnDemand**.*

The import process converts Trac data as follows:

In Trac	In JIRA	Import Notes
Project Environment	Project	Each Trac Environment is imported as a JIRA project. You can either specify an existing JIRA project as the target, or the importer will automatically create a project for you at time of import.
Ticket Type	Issue Type	You can configure mapping of Trac Ticket Types to specific JIRA Issue Types.
Ticket #	External Issue ID	The Trac Ticket number is captured in a JIRA custom field. The import is not designed to have the JIRA issue number match the Trac ticket number.
Status	Status	You can configure mapping of specific Trac values to specific JIRA values.
Summary	Summary	
Description	Description	

Versions	Versions	Versions are imported from Trac (if you choose), and are set to the Un-Released and Un-Archived state.
Component	Components	You can choose to have the importer automatically create your Trac components in JIRA, or choose to have bugs imported into no component in JIRA.
Comments	Comments	
Priority	Priority (or a custom field)	You can choose to map one of either the Trac Priority field or the Trac Severity field (see below) to the built-in JIRA Priority field, and the other to a custom field. (Alternatively, you can choose to map both the Trac Priority field and the Trac Severity field to JIRA custom fields.) When importing into the JIRA Priority field, you can configure mapping of specific Trac values to specific JIRA values.
Severity	Priority (or a custom field)	You can choose to map one of either the Trac Priority field or the Trac Severity field (see below) to the built-in JIRA Priority field, and the other to a custom field. (Alternatively, you can choose to map both the Trac Priority field and the Trac Severity field to JIRA custom fields.) When importing into the JIRA Priority field, you can configure mapping of specific Trac values to specific JIRA values.
Milestone	Milestone	JIRA will create this as a custom field.
Attachments	Attachments	Attachments are extracted from the Trac Environment and saved to disk. To specify the location on disk, see Configuring File Attachments.
Resolution	Resolution	You can configure mapping of specific Trac values to specific JIRA values.
CC	Watcher	
Keywords	Labels	
User	User	The importer will automatically create JIRA users for any Trac users who do not exist in JIRA. • Passwords from Trac are not imported. Users from Trac will need to get their passwords emailed to them. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
Other fields	Custom fields	If your Trac system contains any custom fields, you can choose to map them to specific JIRA custom fields. If your custom fields don't yet exist in JIRA, the importer can automatically create them for you.

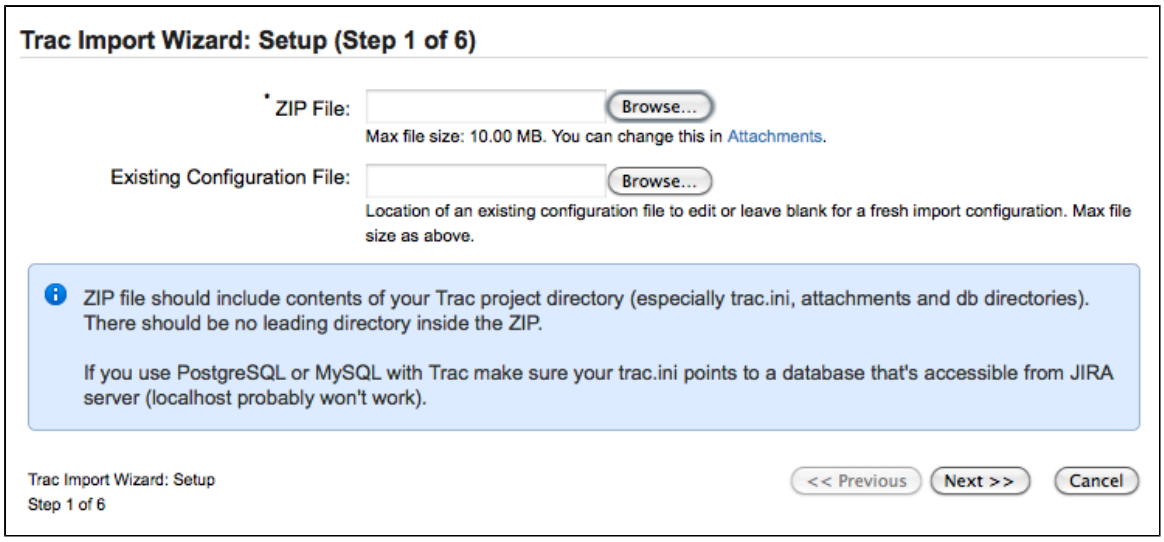
How to import from Trac

First you will need to ZIP up the contents of your Trac environment

1. Go to your Trac Environment.
2. If you use SQLite(Trac default) PostgreSQL or MySQL database make sure your database URL saved in conf/trac.ini is also reachable from JIRA server (using localhost or unix socket won't work).
3. ZIP contents of Trac Environment without any leading directories.

Import into JIRA

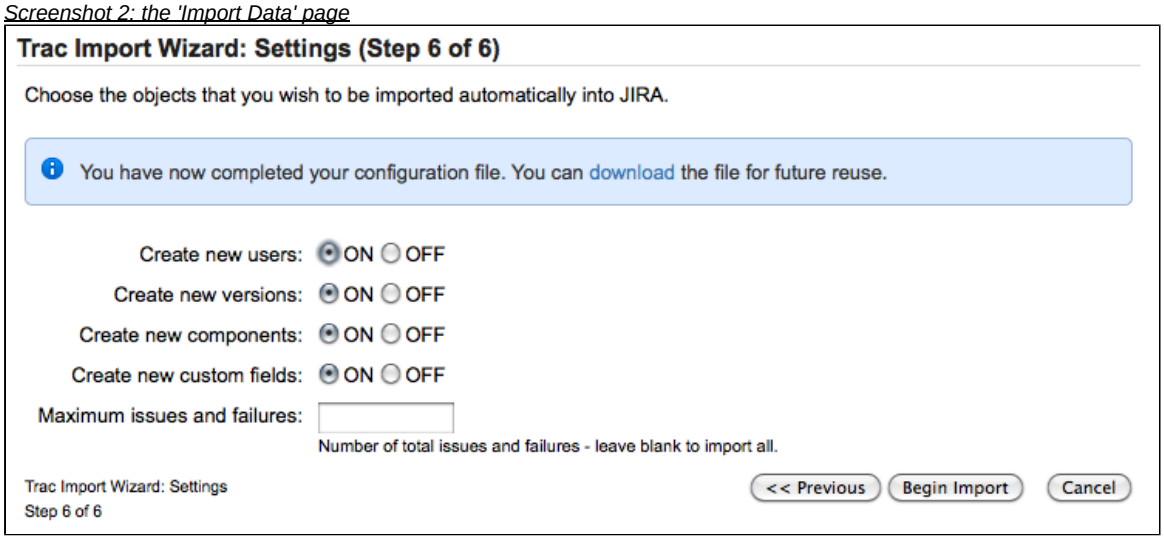
1. Before you begin, please backup your JIRA data.
2. Log in to JIRA as as a user with the **'JIRA Administrators'** global permission.
3. Select **'Administration' > 'System' > 'Import & Export' > 'External System Import' > 'Trac'**.
4. The '**Trac Import Wizard**' page will be displayed.
    - If you don't have a configuration file (or to create a new one, or update the one you have), leave the '**Existing Configuration File**' field blank and click '**Next**'.
      Screenshot 1: the 'Trac Import Wizard' page



5. **Project Key Mappings:** For each Trac project that you wish to import into a new JIRA project, select the JIRA '**Project category**' (if applicable) and type in a unique JIRA 'Name' and 'Key'.
    - The JIRA '**Project key**' will be the prefix for the IDs of all issues in the given project.
    - To import into an existing JIRA project, type a project key and project name that already exists in JIRA.
6. **Custom Fields:** If your Trac system contains any custom fields, you can either choose to import into an existing JIRA custom field or have the importer automatically create a new custom field in JIRA.

    Regardless of whether you specify mapping, the importer will automatically create a JIRA custom field for each extra Trac field, unless you un-check the'**Create new custom fields**' option on the final '**Import Data**' screen (see *Screenshot 2* below).
7. **Field Value Mappings:**
    - **'Priority' field** — If you don't specify mappings, the importer will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority
    - **Usernames** — If you don't specify mapping, the importer will automatically map Trac usernames to JIRA usernames (lowercase).

      Regardless of whether you specify mapping, JIRA will automatically create usernames for missing users unless you un-check the'**Create new users**' option on the final '**Import Data**' screen.
    - **'Status' field** — It is mandatory to map the Trac '**Status**' field to specific values of the JIRA '**Status**' field, as the JIRA '**Status**' field is integral to JIRA workflow (to learn more, please see What is Workflow and Configuring Workflow).
    - **'Resolution' field** — If you don't specify mapping, the importer will create corresponding Resolutions in JIRA instead of using the existing ones.
8. The '**Import Data**' page will be displayed:
   Screenshot 2: the 'Import Data' page



    - '**Create new users/versions**' — Leave 'ON' to create new users/versions/custom fields that do not yet exist in JIRA. (This is the generally recommended setting for these options.) Only select 'OFF' if you do not want JIRA to automatically create these new items for you.
    - '**Create new users/versions/custom fields**' — It is generally recommended that you leave this set to 'ON'. Only select 'OFF' if you do **not** want JIRA to automatically create new versions for Trac '**milestones**' which do not already exist in JIRA.
    - '**Maximum issues and failures**' — If you wish, specify a maximum number of failed issues after which the importer will stop. If you want the import to continue regardless of any failures, leave this field blank. If your Trac instance has a large number of issues, it's generally a good idea to run first the importer on a limited number of issues (e.g. 100), then manually inspect the imported issues to confirm whether your configuration file was specified correctly. When the results are satisfactory, you can run the import with no limit.
9. The importer will display updates as the import progresses, then a success message when the import is complete. You can download the import log if you wish.

Congratulations, you have successfully imported your Trac projects into JIRA! If you have any questions or encounter any problems, please

contact Atlassian support.

Importing Data From CSV

The JIRA Importers plugin, which is bundled with JIRA, allows you to import your data from a comma-separated value (CSV) file. CSV files are text files representing tabulated data and are supported by most applications that handle tabulated data (for e.g. Microsoft Excel, databases, etc.).

The CSV import process consists of:

1. *Preparing your CSV file* (below).
2. *Running the CSV file import wizard* (below). You can choose to map individual fields and field values during the import process.

Please Note:

- Several methods are available for importing data from other issue tracking systems into JIRA. Depending on your other issue tracking system, it may be more appropriate to use one of these other methods than to first export your data from that system to a CSV file and then import that CSV file into JIRA. If your other issue tracking system is listed on the Migrating from Other Issue Trackers page, try using the appropriate method for that issue tracker (which is accessible from that page) to import data into JIRA.
- If you want to raise a bug report or improvement suggestion about this feature, please do so within the JIRA Importers plugin project

On this page:

- Preparing your CSV file
  - CSV file requirements
  - Encapsulating JIRA data in your CSV file
- Running the CSV file import wizard
- Tips for importing CSV data into JIRA fields

*The information on this page does not apply to **JIRA OnDemand**.*

Preparing your CSV file

The JIRA Importers plugin assumes that your CSV file is based off a default Microsoft Excel-styled CSV file. Fields are separated by commas and any content that must be treated literally, such as commas and new lines/'carriage returns' themselves are enclosed in quotes.

For Microsoft Excel and OpenOffice, it is not necessary to quote values in cells as these applications handle this automatically.

CSV file requirements

In addition to being 'well-formed', CSV files have the following requirements.

Each CSV file must possess a heading row with a Summary column

The CSV file import wizard (below) uses a CSV file's header row to determine how to map data from the CSV file's 2nd row and beyond to fields in JIRA.

The header row *should avoid containing any punctuation* (apart from the commas separating each column) or the importer may not work correctly.

The header row *must* contain a column for 'Summary' data.

Commas (as column/field separators) cannot be omitted

For example, this is valid:

```
Summary, Assignee, Reporter, Issue Type, Description, Priority
"Test issue", admin, admin, 1, ,
```

... but this is not valid:

```
Summary, Assignee, Reporter, Issue Type, Description, Priority
"Test issue", admin, admin, 1
```

Encapsulating JIRA data in your CSV file

Capturing data that spans multiple lines

Use quote marks in your CSV file to capture data that spans multiple lines. For example, JIRA treats the following as a valid CSV file with a single record:

```
Summary, Description, Status
"Login fails", "This is on
a new line", Open
```

Treating special characters literally

Use double quote marks (") around a section of text to treat any special characters in that section literally. Once this data is imported into JIRA, these special characters will be stored as part of JIRA's field data. Examples of special characters include carriage returns/enter characters (as shown in the example above), commas, etc.

To treat a double quote mark literally, you can 'escape' them with another double quote mark character. Hence, the CSV value:

- `"Clicking the ""Add"" button results in a page not found error"`
  once imported, will be stored in JIRA as:
- `Clicking the "Add" button results in a page not found error`

Aggregating multiple values into single JIRA fields

You can import multiple values into a JIRA field that accepts multiple values (e.g. 'Fix (for) Version', 'Affects Version', 'Component', 'Labels'). To do this, your CSV file must specify the same column name for each value you wish to aggregate into the mapped JIRA field. The number of column names specified must match the maximum number of values to be aggregated into the mapped field. For example:

```
IssueType, Summary, FixVersion, FixVersion, FixVersion, Component, Component
bug, "First issue", v1, , , Component1,
bug, "Second issue", v2, , , Component1, Component2
bug, "Third issue", v1, v2, v3, Component1,
```

In the above example, the third imported issue will have its 'Fix Version' set to multiple values.

Be aware that only a limited number of JIRA fields support multiple values. The CSV importer will not allow you to import aggregated data into JIRA fields which only support a single value.

Importing attachments

You can attach files to issues created from your CSV file. To do this, specify the URL of your attachment in an 'Attachments' column within your CSV file.

```
Assignee, Summary, Description, Attachment, Comment
Admin, "Issue demonstrating the CSV attachment import", "Please check the attached image
below.", https://jira-server:8080/secure/attachment/image-name.png, "01/01/2012 10:10;Admin;
This comment works"
```

URLs for attachments only support the HTTP and HTTPS protocols.

Creating sub-tasks

You can create sub-tasks of issues, whose structure can be encapsulated in a CSV file. To do this:

- Your CSV file requires two additional columns whose headings should be named similarly to 'Issue Id' and 'Parent Id'.
- Ensure each regular (non sub-task) issue is given a unique (sequential) number in the 'Issue Id' column. Do not include any value in the 'Parent Id' fields for regular issues.
- To create a sub-task of a regular issue in your CSV file, reference the unique 'Issue Id' number of the regular issue in the 'Parent Id' column. Do not include any value in the 'Issue Id' fields for sub-tasks.

For example:

```
IssueType, Summary, FixVersion, FixVersion, FixVersion, Component, Component, Issue ID, Parent
ID, Reporter
bug, "First issue", v1, , , Component1, , 1, , jbloggs
bug, "Second issue", v2, , , Component1, Component2, 2, , fferdinando
bug, "Third issue", v1, v2, v3, Component1, , 3, , fferdinando
subtask, "Fourth issue", v1, v2, , Component2, , , 2, jbloggs
```

Allowing comments in CSV files

Your CSV file can contain comments which are not processed by the JIRA import wizard. Any text appearing to the right of a hash (#) symbol on a line of text in the CSV file will be disregarded by the JIRA import wizard. For example:

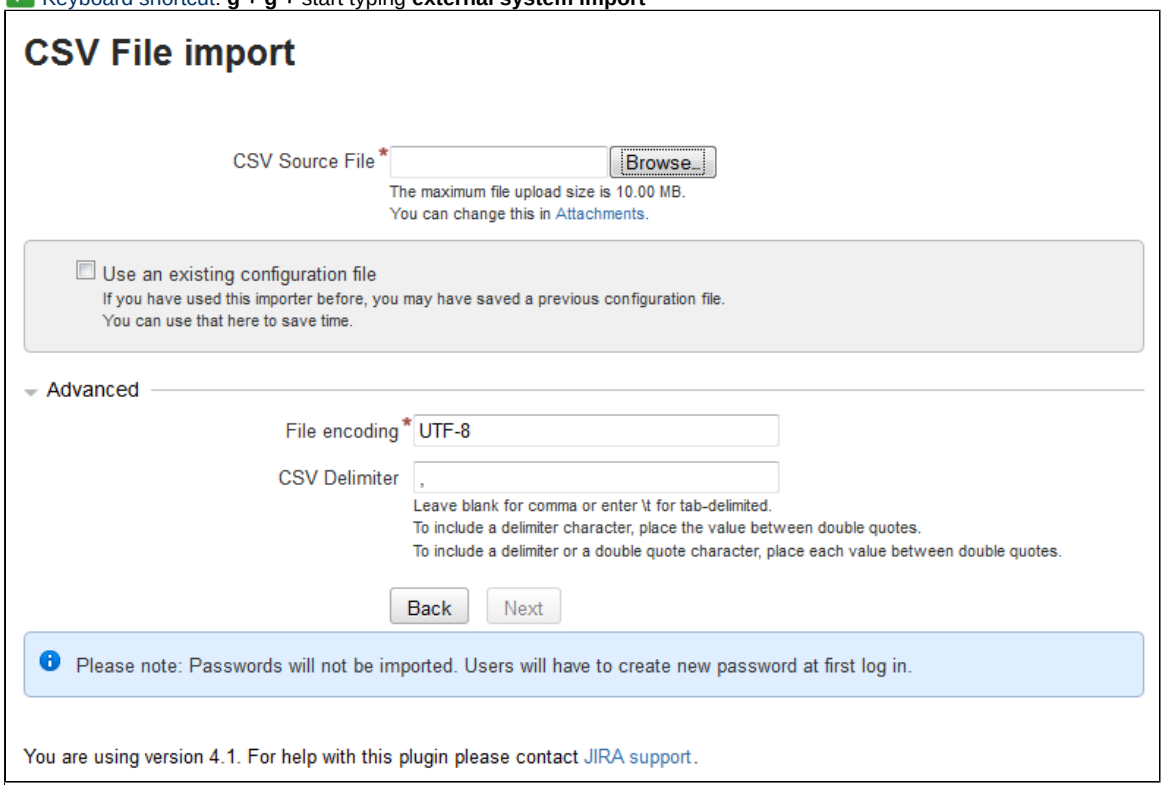
```
Assignee, Summary, Comment
Admin, "Test issue", "This is a comment that will be added to the issue once imported into
JIRA" # And this is a comment that will not be processed by the JIRA import wizard.
```

Comments are useful if you want to add additional information to the CSV file that you do not want JIRA to import.

Running the CSV file import wizard

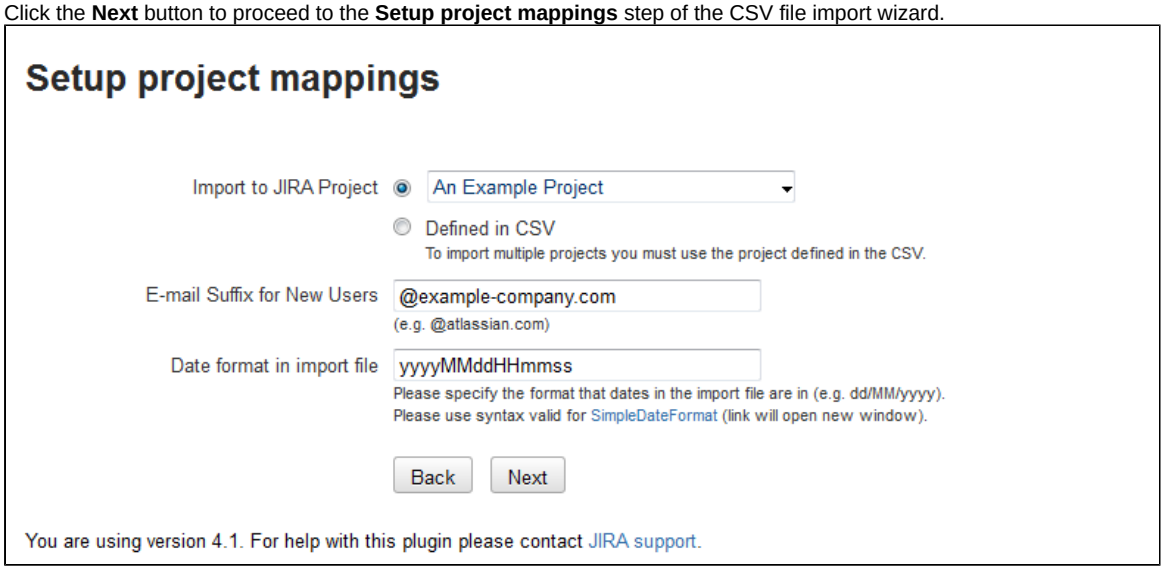
If your JIRA installation has existing data, then before you begin, backup your existing JIRA data.

1. Log in to JIRA as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Import & Export > External System Import > Import** button associated with the Comma-separated values (CSV) option to open the **CSV File import** page.
   Keyboard shortcut: **g + g +** start typing **external system import**



3. On the **CSV File import** page, select your **CSV Source File**. If you want to change the file's encoding and CSV delimiter format, click the **Advanced** heading to reveal this option (as shown in the above screenshot).
   Note:
   - The file will be imported using the **File encoding** you specify here (which is **UTF-8** by default).
   - If your CSV file uses a different separator character other than a comma, specify that character in the **CSV Delimiter** field.
4. Leave the **Use an existing configuration file** option check box cleared if you do not have a configuration file or if you want to create a new configuration file. Configuration files specify a mapping between column names in your CSV file's header row and fields in your JIRA installation.
   Note:
   - If you select this option, you will be asked to specify an **Existing Configuration File**.
   - If you do not select this option, then at the end of the CSV file import wizard, JIRA will create a configuration file which you can use for subsequent CSV imports (at this step of the CSV file import wizard).

5. Click the **Next** button to proceed to the **Setup project mappings** step of the CSV file import wizard.

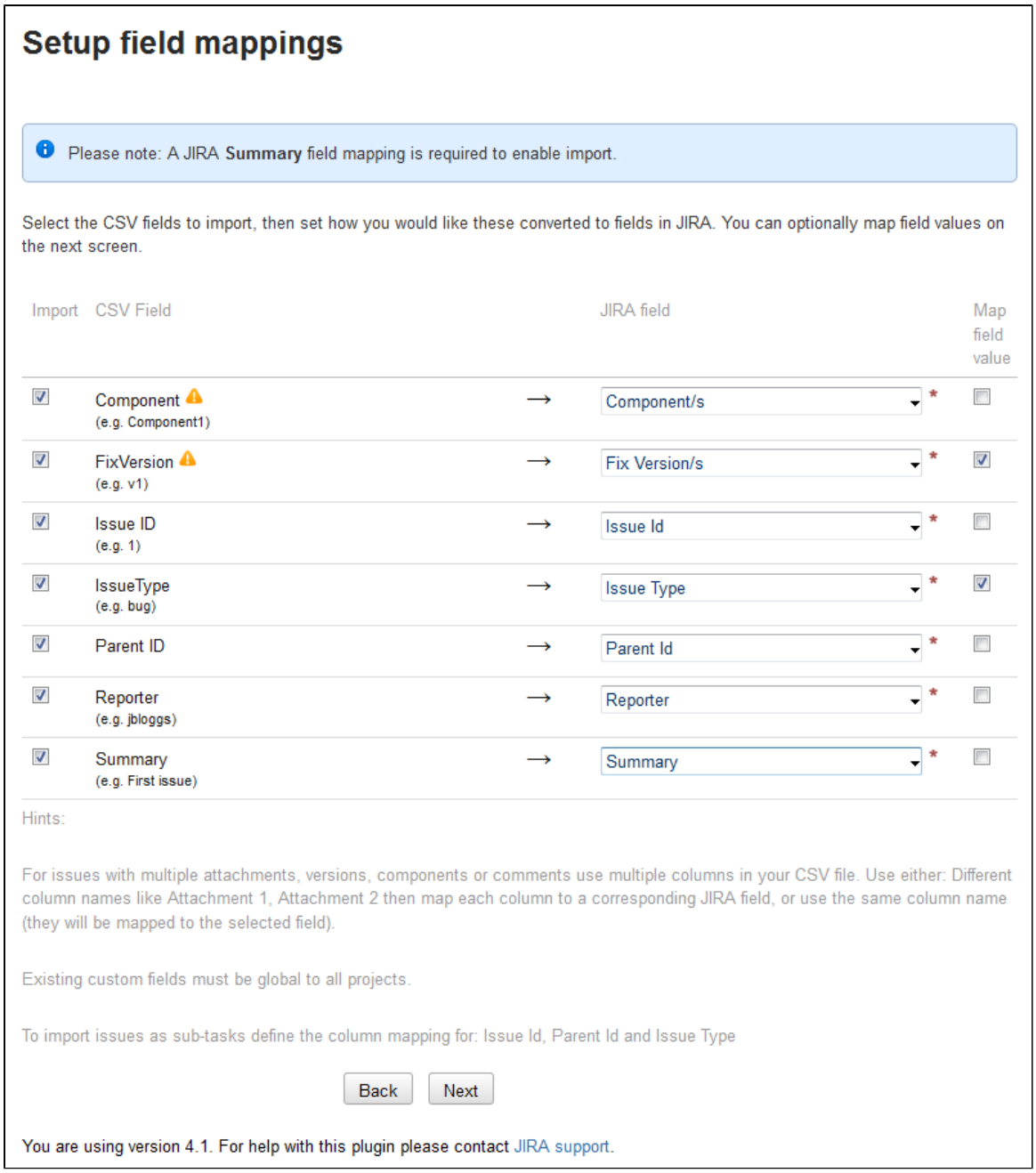


6. On the **Setup project mappings** page, you can either import *all* your issues into one JIRA project (new or existing); or import into multiple projects by including the project data in your CSV file. Complete the following fields/options:

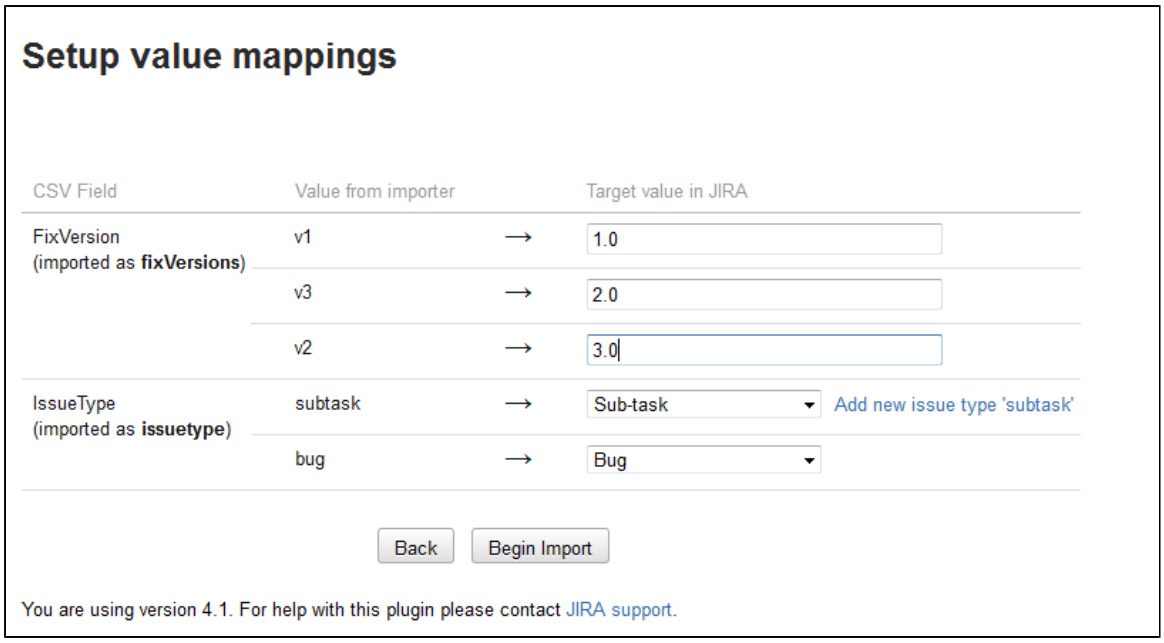
Import to JIRA Project	Choose either of the following:
	• **Select a project** and then do either of the following:
	  • Start typing the name (or key) of a project that already exists in JIRA or use the dropdown menu to select an existing JIRA project.
	  • Select **Create New** from the dropdown menu and in the resulting **Add A New Project** dialog box, type the following:
	  a. A new project **Name**
	  b. A new project **Key**
	  The will be used as the prefix for all issue IDs in your JIRA project.
	  c. The **Project Lead**.
	• **Defined in CSV**. Ensure that every issue in your CSV file includes data for the project name/key.

E-mail Suffix for New Users	Type the email address domain for any new users specified in the CSV file which will be added to JIRA during the import.
Date format in import file	Specify the date format used in your CSV file. Use the syntax that complies with the Java SimpleDateFormat.

7. Click the **Next** button to proceed to the **Setup field mappings** step of the CSV file import wizard.



8. On the **Setup field mappings** page, specify each **CSV Field** (determined by your CSV file's header row) you want to import into your chosen JIRA project by selecting their check boxes under the **Import** column on the left.
   Please Note:
   - At least one of these fields must contain data for JIRA's **Summary** field.
   - If your CSV file contains more than one of the same field name specified in its header row, the CSV file import wizard will aggregate these into a single field, which will be marked by a ⚠ symbol at this step of the wizard.
9. In the **JIRA field** column, select the JIRA fields you want to match to fields defined in your CSV file (i.e. each **CSV Field** you selected in the previous step).
   Please Note:
   - The **Summary** field must be specified for one of your JIRA fields and the **Next** button will remain unavailable until you do so.
   - For CSV fields which have been aggregated by the CSV file import wizard, you will only be able to select JIRA Fields that support multiple values.
10. To modify the values of any fields' data in the CSV file *before* they are imported into JIRA, select the **Map field value** check boxes next to the appropriate fields.
11. Click the **Next** button to proceed to proceed to the **Setup value mappings** step of the CSV file import wizard.

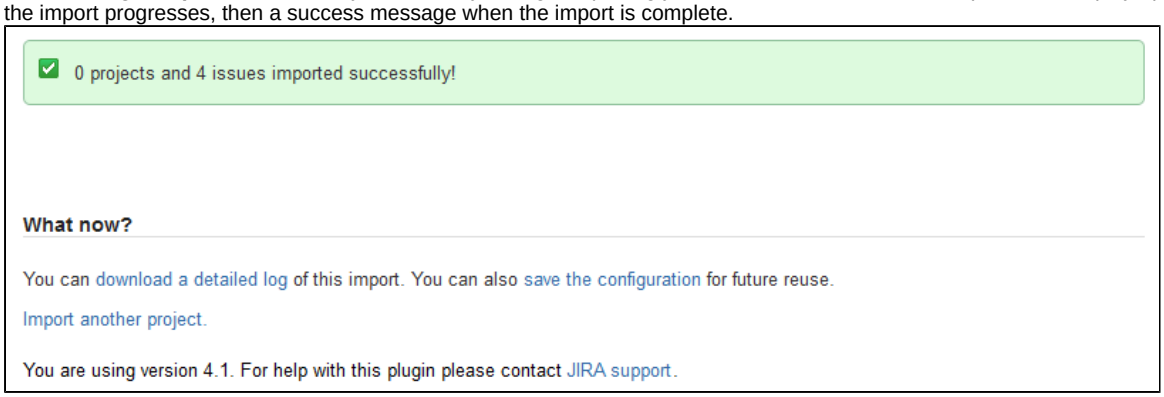


12. On the **Setup value mappings** page, any fields whose **Map field value** check boxes were selected in the previous step of the CSV file import wizard will be presented on this page.

    Please Note:
    - Leave a field cleared or clear any content within it if you wish to import the value 'as is'.
    - You can create new **Priority**, **Resolution** and **Issue Type** values in JIRA (i.e. based on the data in your CSV file) by clicking the **Add new ...** link (e.g. **Add new issue type 'subtask'** shown in the screenshot above) next to the appropriate field.
    - If you are importing a username-based CSV field (e.g. **Reporter** or **Assignee**) and you do not select the **Map field value** check box for this field in the previous step of the CSV file import wizard, then the importer will automatically map imported usernames from the CSV file to (lowercase) JIRA usernames.

      Regardless of whether or not you specify a mapping for these username fields, JIRA will automatically create usernames based on the data in your CSV file if they have not already been defined in JIRA.

13. Click the **Begin Import** button when you are ready to begin importing your CSV data into JIRA. The importer will display updates as the import progresses, then a success message when the import is complete.



Note:
- If you experience problems with the import (or you are curious), click the **download a detailed log** link to reveal detailed information about the CSV file import process.
- If you need to import another CSV file with the same (or similar) settings to what you used through this procedure, click the **save the configuration** link to download a CSV configuration file, which you can use at the first step of the CSV file import wizard.

Congratulations, you have successfully imported your CSV data into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Tips for importing CSV data into JIRA fields

Below are some helpful tips when importing data from your CSV file into specific JIRA fields:

JIRA Field	Import Notes
Project	CSV data is imported on a per-project basis. You can either specify an existing JIRA project(s) as the target, or the importer will automatically create a new project(s) for you at time of import.

Summary	This is the only required field.
Component(s)	You can import issues with multiple components by entering each component in a separate column.
Affects Version(s)	You can import issues with multiple 'Affects Versions' by entering each version in a separate column.
Fix Version(s)	You can import issues with multiple 'Fix Versions' by entering each version in a separate column.
Comment Body	You can import issues with multiple comments by entering each comment in a separate column.
Date Created	Please use the date format specified on the second step of the CSV import wizard.
Date Modified	Please use the date format specified on the second step of the CSV import wizard.
Due Date	Please use the date format specified on the second step of the CSV import wizard.
Issue Type	If not specified in your CSV file, imported issues will be given the default (i.e. first) Issue Type as specified in your JIRA system Defining 'Issue Type' Field Values. You can also create new JIRA values on-the-fly during the import process.
Labels	You can import issues with multiple labels by entering each label in a separate column.
Priority	If not specified in your CSV file, imported issues will be given the default (i.e. first) Priority as specified in your JIRA system Defining 'Priority' Field Values. You can also create new JIRA values on-the-fly during the import process.
Resolution	If not specified in your CSV file, imported issues will be given the default (i.e. first) Resolution as specified in your JIRA system Defining 'Resolution' Field Values. You can also create new JIRA values on-the-fly during the import process.
Status	Can only be mapped to existing workflow statuses in JIRA. If not specified in your CSV file, imported issues will be given the default (i.e. first) Status as specified in your JIRA system.
Original Estimate	The value of this field needs to be specified as number of seconds.
Remaining Estimate	The value of this field needs to be specified as number of seconds.
Time Spent	The value of this field needs to be specified as number of seconds.
Users	You can choose to have the importer automatically create JIRA users for any values of the Assignee or Reporter field. • Users will be created as active accounts in JIRA. Users will need to get their passwords emailed to them the first time they log into JIRA. • Users with no real name will get the portion of their email address (login name) before the "@" character as their Full Name in JIRA. • If you are using External User Management, the import process will not be able to create JIRA users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created.
Other fields	If your wish to import any other fields, you can choose to map them to specific JIRA custom field(s). If your custom fields don't yet exist in JIRA, the importer can automatically create them for you. If your custom field is a date field, please use the date format specified on the second step of the CSV import wizard.

Commonly Asked CSV Questions and Known Issues

- Overview
- Commonly Asked Questions
  - The importer simply doesn't work on my CSV file!
  - The importer fails at date fields, why?
  - Why does the importer always ask me to map values to column (at Step 3 of 5)?

- Known Issues
  - Why couldn't I import from cascading select fields?
  - Why couldn't I import component/version Custom Fields?
  - Known JBoss issue

Overview

This page answers some of the commonly asked CSV questions our technical support staff have encountered. If you are not able to find an answer from this page and our issue tracker, feel free to create a support issue.

Commonly Asked Questions

The importer simply doesn't work on my CSV file!

Please make sure that it is a valid and not-bad-formatted CSV file. You should be able to spot this with by turning on detailed logging and profiling. Also, please double check your configuration file and ensure that it's properly configured, e.g. exact delimiter, date format, etc.

The importer fails at date fields, why?

If you are seeing error message similar to this:

```
[00:55:28] FAILED: Customfield value 01/Nov/06 12:00 AM is invalid
[00:55:28] com.atlassian.jira.issue.customfields.impl.FieldValidationException: Invalid date
format. Please enter the date in the format "MMM/dd/yy".
at
com.atlassian.jira.issue.customfields.converters.DatePickerConverter.getTimestamp(DatePickerConvert
com.atlassian.jira.issue.customfields.impl.DateCFType.getSingularObjectFromString(DateCFType.java:4
com.atlassian.jira.imports.importer.impl.DefaultJiraDataImporter.importIssues(DefaultJiraDataImport
com.atlassian.jira.imports.importer.impl.DefaultJiraDataImporter.doImport(DefaultJiraDataImporter.j
com.atlassian.jira.imports.importer.impl.ImporterThread.run(ImporterThread.java:21)
```

There are a few possible reasons:

- Date format is not correctly set in the import configuration file
- Date Picker and Date Time Picker formats are not consistent, e.g.

```
jira.date.picker.java.format=dd/MMM/yy
jira.date.time.picker.java.format=MMM/dd/yy hh:mm a
```

should be corrected to,

```
jira.date.picker.java.format=dd/MMM/yy
jira.date.time.picker.java.format=dd/MMM/yy hh:mm a
```

See Changing the Due Date Input Format for more information about changing these values.

Why does the importer always ask me to map values to column (at Step 3 of 5)?

It is because you have selected *Map Field Value* for the particular columns. To use the values from the CSV, you need just to map the column to the *Corresponding JIRA field*, otherwise, select the *Map field value* checkbox.

Known Issues

Why couldn't I import from cascading select fields?

This is an open issue being tracked at JIM-231. Feel free to comment and vote on it.

Why couldn't I import component/version Custom Fields?

This issue is being tracked at JIM-233. Feel free to comment on it.

Known JBoss issue

There is a known problem that prevents the CSV Importer from being used with JIRA instances running on JBoss 4.x. This is due to a compatibility issue between the JBoss 4.x commons-collections.jar and the JIRA commons-collections.jar. The workaround is to replace the commons-collections.jar in JBoss 4.x with the more recent JIRA version. Please see JRA-6473 for further details.

How to Import CSV Data with PVCS Command

The content on this page relates to platforms which are not supported for JIRA. Consequently, Atlassian can not guarantee providing any support for it. Please be aware that this material is provided for your information only and using it is done so at your own risk.

Importing from PVCS is not supported yet, but there is a feature request being tracked here. The above problem occurs when the pvcs command is not configured in the CSV configuration.

Resolution

In order to import the author of the comment and the date of the comment successfully, there are a few required conditions:

- Append the settings in the csv configuration file which you have saved the configuration through wizard

```
settings.advanced.mapper.comment : com.atlassian.jira.imports.csv.mappers.PvcsComment
```

For the latest plugin version 2.6.1, please use the configuration below:

```
settings.advanced.mapper.comment :
com.atlassian.jira.plugins.importer.imports.csv.mappers.PvcsComment
```

- Username (Example: eddie) must exists in JIRA
- The format of the comment should be as below:

```
"QA Note on Close: eddie: 4/28/2004 11:54:35 AM: Closing this defect as it is no longer
relevant"
```

Moving or Archiving Individual Projects

Over time, your organisation's requirements may change. This can lead to needing to:

- Archive a completed or obsolete project.
- Split a large JIRA instance into several JIRA instances, with particular projects in each.
- Restore a single project from a backup file into a JIRA instance.
- Restore an entire JIRA instance, from a backup into a new empty JIRA instance.

*The information on this page does not apply to **JIRA OnDemand**.*

Archiving a Project

It is sometimes necessary to archive an old project, while retaining the project's data for future auditing purposes. There are a number of ways to achieve this:

- Online archiving
  - 'Hiding' a project
  - Making a project 'Read-Only'
  - Accessing an archived online project
- Offline archiving
  - Archiving a project offline
  - Accessing an archived offline project
  - Restoring a deleted project

*The information on this page does not apply to **JIRA OnDemand**.*

Online archiving

Archiving a project online means keeping all of the project's issue data in your live JIRA instance. The advantage of archiving a project online is that you can easily make the project accessible again if required.

There are two ways to archive a project online:

'Hiding' a project

A 'hidden' project will still be visible via the 'Administration' menu, but it will no longer appear in the 'Browse Projects' list, and no-one will be able to search, view or modify any of the project's issues.

1. Create a new permission scheme. Leave all of the permissions empty.
2. Associate the new permission scheme with the project that you wish to hide (see Assigning a Permission Scheme to a Project).

Making a project 'Read-Only'

If you make a project read-only, the project will be visible via the 'Administration' menu, and will appear in the 'Browse Projects' list. The project's issues will be searchable and viewable, but noone will be able to modify them.

1. Create a new permission scheme. Grant the **'Browse Project'** permission to everyone who needs to be able to search or browse the project, or view its issues. Leave all of the other permissions empty.
2. Associate the new permission scheme with the project that you wish to hide (see Assigning a Permission Scheme to a Project).

Accessing an archived online project

If you archived a project online, by hiding it or making it read-only, then all of the project's data can be made accessible very easily. Simply associate the project with a permission scheme where the appropriate permissions (e.g. **'Edit Issue', 'Assign Issue', 'Resolve Issue',** etc) are assigned to the appropriate people.

Offline archiving

Archiving a project offline means creating an XML backup, then deleting the project and all of its issue data from your live JIRA instance. The project will no longer be available via the 'Administration' menu or the 'Browse Projects' list, and its issues will no longer exist in your live JIRA system.

The disadvantage of offline archiving is that there is no easy way to restore a deleted project to your live JIRA instance.

> If there is a possibility that you will need to restore the project into your live JIRA instance at some point in the future, then online archiving is recommended. Offline archiving should only be done if you are certain you will never need to restore this project to a live JIRA instance (i.e. you will only ever restore the data to a non-production instance).

Archiving a project offline

1. Create a global XML backup of your entire live JIRA instance.
2. Import the XML backup into a test JIRA instance. **Make sure that the test JIRA instance uses a separate database from your live JIRA instance, as the import will overwrite all data in the database.**
3. In your test JIRA instance, verify that you can view the issues of the project that you are archiving.
4. In your live JIRA instance, select **Projects** from the **Administration** menu, then click the **Delete** link to delete the project and all of its issues.

Accessing an archived offline project

1. Import the XML backup into a test JIRA instance. **Make sure that the test JIRA instance uses a separate database from your live JIRA instance, as the import will overwrite all data in the database.**

Restoring a deleted project

If you wish to restore a project from a backup file, please refer to the instructions in the Restoring a Project from Backup documentation. Note that the JIRA version and database type must be consistent with when the archive was created.

Splitting a JIRA instance

Occasionally an organisation may need to split its existing JIRA instance into two separate instances. For example, there might be a requirement to have some particular projects in one JIRA instance, and other projects in a second instance.

> **Note**
>
> This process requires two separate server licenses.

> *The information on this page does not apply to **JIRA OnDemand**.*

To split a JIRA instance:

1. Back up your database, using your database backup procedures, and verify the backup.
2. Back up your attachments directory and verify the backup.
3. Install JIRA on your new server.
   Please Note:
   - The JIRA version number on your new server must be the same as (or higher than) the version number on your existing server.
   - Do not use the same JIRA Home Directory for the two JIRA instances. Specify a new JIRA home directory for the JIRA on your new server.
   - Do not connect the two JIRA instances to the same external database instance.
4. Create an XML backup from your existing JIRA server, as described in Backing up data.
5. Import the XML backup file into your new server, as described in Restoring data.
6. Copy the attachments directory from your existing server to your new server, and configure your new server to use its own directory (for details please see Enabling File Attachments).
7. At this point you should have two JIRA instances with the same users, projects, issues and attachments. Log in to both instances and perform some random searches to verify that the data is identical in both instances.
8. Delete the non-required projects from each JIRA instance.

Integrating with a Source Control System

JIRA can be easily integrated with many popular source control systems:

- Integrating JIRA with FishEye
- Integrating JIRA with CVS and ViewCVS
- Integrating JIRA with Subversion
- Integrating JIRA with Perforce
- Integrating JIRA with ClearCase

The most scalable and recommended solution is to use FishEye, which supports Subversion, Git, Perforce, Clearcase, CVS, and Mercurial with real-time notifications of code changes plus web-based reporting, visualisation, search and code sharing.

*The information on this page does not apply to **JIRA OnDemand**.*

Integrating JIRA with FishEye

JIRA's FishEye integration allows you to browse your source-control repository from inside JIRA, provided you are using Atlassian FishEye with your source-control repository. FishEye integration is implemented as a plugin (drop-in extension) to JIRA, which ships with JIRA.

FishEye integration allows you to:

- View an Issue's FishEye Changesets
- Browse a Project's FishEye Changesets
- Add the FishEye Charts Gadget to your JIRA Dashboard
- Add the FishEye Recent Changesets Gadget to your JIRA Dashboard

On this page:

- Step 1. Create an Application Link Between a JIRA and a FishEye/Crucible Server
- Step 2. Configure the FishEye Plugin in JIRA
- Step 3. Add Permissions to Users
- Step 4. Link FishEye Repositories to JIRA Projects
- Step 5. (Optional) Subscribe JIRA to FishEye Gadgets
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Step 1. Create an Application Link Between a JIRA and a FishEye/Crucible Server

To begin integrating JIRA with either FishEye or Crucible, you must create an application link between your JIRA server and FishEye or Crucible server.

Before you begin:

- You require JIRA System Administration permissions in order to perform this procedure.
- For JIRA to successfully integrate with FishEye, you must set up the FishEye web server to receive remote API calls. A FishEye administrator can do this by visiting the FishEye web server's 'Server Settings' page (which can be accessed from FishEye's 'Administration' area) and setting the 'Remote API' setting to 'On'. See Configuring the FishEye Web Server for more information.

To create an application link between a JIRA and a FishEye/Crucible server:

1. In JIRA, click '**Administration**' in the top navigation bar to open the JIRA Administration console.
2. Select **'Plugins > Application Links'** from the top menu.
3. The 'Configure Application Links' screen will appear. Click '**Add Application Link**'.
4. The first screen of the 'Add Application Link' wizard will appear. Copy the base URL for your FishEye site (e.g. `http://fisheye.example.com:8060` and paste it into the '**Server URL**' field.
5. Click '**Next**'.
6. The 'Link to FishEye' screen will appear. Enter the following information:
    - Create a link back to this server – This option is selected by default. Leave it selected, if you want create a reciprocal link back from your FishEye server to your JIRA server.
    - Username – Enter the username of the administrator on your FishEye site.
    - Password – Enter the password of the administrator on your FishEye site.
    - Reciprocal Link URL – Leave this field at its default value, pointing to your JIRA site.
7. Click '**Next**'.
8. The 'Set Users and Trust' screen will appear. Enter the following information:
    - The servers have the same set of users – This option is selected by default. Let it remain selected.
    - These servers fully trust each other – This option is selected by default. Let it remain selected.
9. Click '**Create**'. The application link will be created and displayed on the 'Configure Application Links' page.

Step 2. Configure the FishEye Plugin in JIRA

The FishEye plugin for JIRA is bundled as part of the JIRA package, so there is no need to install it. Now you will configure the plugin for your installation and configure JIRA to trust FishEye.

1. Navigate to the JIRA Administration console.
2. In JIRA, click '**Administration**' in the top navigation bar.
3. Select **'Plugins > FishEye Configuration'** from the top menu.
4. The 'JIRA FishEye Plugin' screen will appear. Click '**Edit Primary Configuration**'.
5. Enter the following information:
    - Enable Crucible Integration – Select 'True' if you want to enable Crucible integration (e.g. view reviews related to an issue).
    - Update the other fields as desired.
6. Click '**Update**'.
7. Click '**Application Links Configuration**' at the bottom of the screen.
8. Click '**Trusted Applications**' in the '**Outgoing Authentication**' column for your FishEye/Crucible link.
9. Click '**Modify**' and enter the following information:
    - '**IP Patterns**': Enter the IP addresses for your FishEye/Crucible instance (separated by commas), e.g. 127.0.0.1, 172.20.5.95.
10. Click '**Apply**'.
11. Click '**Incoming Authentication**' in the left menu.
12. Click '**Modify**' and enter the following information:
    - '**IP Patterns**': Enter the IP addresses for your JIRA instance (one per line), e.g. 127.0.0.1, 172.20.5.95.
    - '**URL Patterns**': Enter the following paths (one per line):

```
/secure/CreateSubTaskIssueDetails.jspa
/browse/
/rest
/plugins/servlet/applinks/whoami
/plugins/servlet/streams
/rpc/soap
/sr/jira.issueviews:searchrequest
/secure/RunPortlet
```

13. Click '**Apply**' and then '**Close**'.

Step 3. Add Permissions to Users

Before linking Fisheye Repositories to JIRA Projects, you will need to add the correct permissions to the users that will be able to see the Fisheye information in the JIRA Projec pages and tickets:

1. Go to your JIRA URL in your browser, e.g. `http://jira.mycompany.com:8080`.
2. Click '**Administration**' in the top navigation bar to open the JIRA Administration console.
3. Click **'Projects'** in the top menu and select the JIRA project that you want to associate with a FishEye repository/Crucible project. The JIRA project's configuration page will appear.
4. Locate the '**Permissions**' option and click ' **Default Permission Scheme**' or the specific scheme you are using for the project. The page for configuring permission from your JIRA project will appear.
5. On the right-upper corner open the "**Actions**" menu and click on "**Edit Permissions**".
6. Locate the "**View Version Control**" permission and then click "Add" and add to the list the Users and/or groups that will be able to see the Fisheye and Crucible data in JIRA.

Step 4. Link FishEye Repositories to JIRA Projects

In this step you link your FishEye repository to your JIRA project via a project link. Associating JIRA projects with FishEye repositories or

Crucible projects is recommended as it greatly improves JIRA's responsiveness when your users view the 'Source' or 'Review' tabs on JIRA issues. Otherwise, JIRA must scan for all FishEye repositories/Crucible projects which are accessible to a user, whenever that user views these tabs on a JIRA issue.

This process is mandatory if your FishEye/Crucible server has no publicly accessible repositories/projects and JIRA's application link to this FishEye/Crucible server uses OAuth outgoing authentication.

Before you begin:

- You can only perform this procedure once an application link has been established between your JIRA and FishEye/Crucible servers (above).

To crete a link between a JIRA project and a FishEye/Crucible repository:

1. Go to your JIRA URL in your browser, e.g. `http://jira.mycompany.com:8080`.
2. Click '**Administration**' in the top navigation bar to open the JIRA Administration console.
3. Click **'Projects'** in the top menu and select the JIRA project that you want to associate with a FishEye repository/Crucible project. The JIRA project's configuration page will appear.
4. Locate the 'Application Links' option and click '**Configure Application Links**'. The page for configuring links from your JIRA project to the entities of applications (already linked on your JIRA server) will appear.
5. Create a project link between your JIRA project and the appropriate FishEye repository/Crucible project. (See the procedure on Adding Project Links between Applications from step 4 for details.)

Step 5. (Optional) Subscribe JIRA to FishEye Gadgets

In this optional step, you will make your FishEye gadgets available for use in JIRA. This will allow JIRA users will be able to add any FishEye gadget to their dashboards.

1. Click '**Dashboards**' in JIRA's top navigation bar.
2. Click '**Add Gadget**'.
3. The 'Gadget Directory' popup window will appear. Click '**Gadget Subscriptions**'.
4. The 'Gadget Subscriptions' popup window will appear. Click '**Add Subscription**'.
5. The 'Add Subscriptions' popup window will appear. Copy the base URL for your FishEye site (e.g. `http://fisheye.example.com:8060` and paste it into the text box on the screen.
6. Click '**Add Subscription**'.
7. Click '**Finished**'. The FishEye gadgets are now available in your JIRA gadget directory.

Notes

- **Integrating FishEye with JIRA** — You can also configure FishEye to integrate with JIRA, which enables you to view JIRA data from within FishEye. Please see JIRA Integration in FishEye in the FishEye documentation for instructions.
- **JIRA requires FishEye to manually refresh repository cache when repository changes are made** — When a repository is removed, or when there has been any change in FishEye repositories, JIRA does not update the FishEye repository list cache automatically. You must manually refresh the repository list cache. This is done in JIRA: 'Administration' > 'FishEye Configuration' > 'Refresh Cache' link (next to 'Repository List Cache').
- **Associating a JIRA Project with a Repository Path —** Once an application link has been established between JIRA and a FishEye site, you can associate a Repository Path on that FishEye site with a JIRA project, via the '**Select Path**' link of the 'Repository Path' option in the 'Settings' section of JIRA's Project Configuration area. Specifying a Repository Path changes the behaviour of the 'Source' tab on a JIRA issue:
  - If no Repository Path is specified, the 'Source' tab on a JIRA issue will only show the commits/changes which include that JIRA issue number in the commit log.
  - If a Repository Path is specified, the 'Source' tab on a JIRA issue will show all commits/changes made in this repository path.

Related Topics

- View an Issue's FishEye Changesets
- Browse a Project's FishEye Changesets
- Add the FishEye Charts Gadget to your JIRA Dashboard
- Add the FishEye Recent Changesets Gadget to your JIRA Dashboard

Integrating JIRA with CVS and ViewCVS

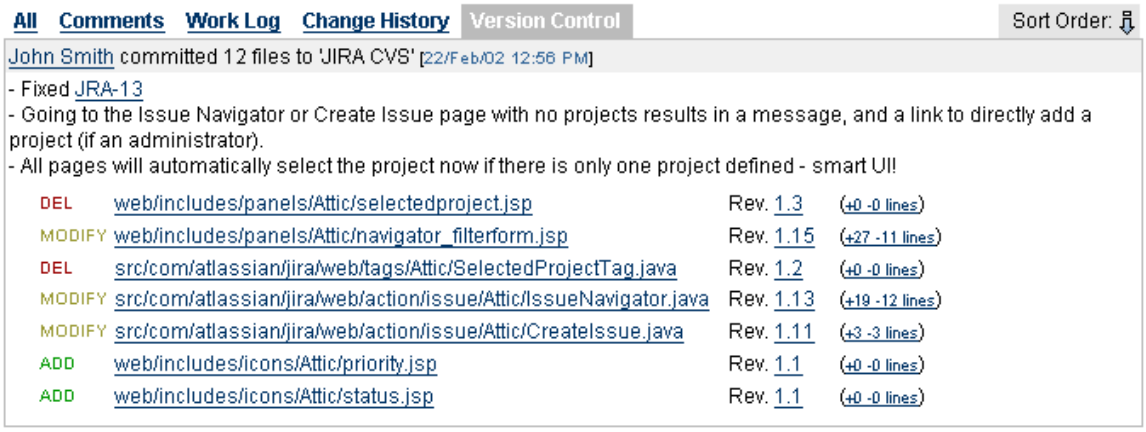
JIRA's CVS integration shows the related CVS commit information for an issue. When a CVS commit message mentions an issue, JIRA picks this up and displays the commit log in a tab in the mentioned issue.

CVS is also supported by Atlassian FishEye, providing a highly scalable and comprehensive integration with JIRA, including real-time notifications of code changes plus web-based reporting, visualisation, search and code sharing. For details, please see Integrating JIRA with FishEye.

JIRA's CVS integration features include:

- Ability to interact with a CVS server log directly via local access, pserver or external (ssh) protocols, or to parse a CVS log file generated by an external process.

- Access to the version control information in JIRA can be easily controlled using flexible permissions. If you are running a public instance of JIRA, and do not want the rest of the world to see the version control information, JIRA can be configured to restrict access to that information to the chosen users.
- ViewCVS or FishEye are supported out-of-the-box; and Subversion is available as a plugin (drop-in extensions to JIRA).
- If CVS integration is configured, the files and revisions in JIRA are linked to the relevant pages. E.g. the following screenshot shows a JIRA project:



  Because ViewCVS is configured, JIRA has turned the displayed commit information into ViewCVS links.
  - Clicking the name of the file will take the user to the ViewCVS file summary page.
  - Clicking the revision will take the user to the page that shows the contents of the file as it was at that revision.
  - Clicking the 'diff' summary will show the ViewCVS 'diff' page between the shown revision of the file and its previous revision
- Each project in JIRA can be associated with a CVS module. A project can also have multiple modules.

There are 3 steps to configure CVS integration in JIRA:

1. **Create a CVS module**
2. **Associate project(s) with CVS module(s)**
3. **Grant permission to view CVS information**

On this page:

- How JIRA's CVS integration works
- Step 1. Create a CVS Module in JIRA
- Step 2. Associate Project(s) with CVS Modules
- Step 3. Configuring Permissions
- Disabling Automatic CVS Log Retrieval
- Adjusting the Frequency of Module Updates
- CVS Aliases

*The information on this page does not apply to **JIRA OnDemand**.*

How JIRA's CVS integration works

JIRA retrieves the CVS commit information for an issue by parsing the output of the 'cvs rlog' (or cvs log) command of each associated CVS module and scanning it for the issue's key. If an issue key is found in the commit message, the commit message is displayed on the Version Control tab for the issue.

If you have allowed JIRA to automatically synchronise with the CVS repository, JIRA will periodically run the 'cvs rlog' command for the module and store the results in a file which path is specified by the module's *Log File Path* attribute. The file is then parsed for commit information.

Even if you are using local repository access JIRA will obtain the CVS log for the module and then parse it. JIRA does not access the CVS repository directly.

If you have chosen to update the log manually, JIRA will only periodically parse the CVS log specified by the module's *Log File Path* attribute.

As JIRA parses the module's CVS log and keeps relevant commits in memory, the required memory for JIRA is relative to the size of the CVS module.

Please note:

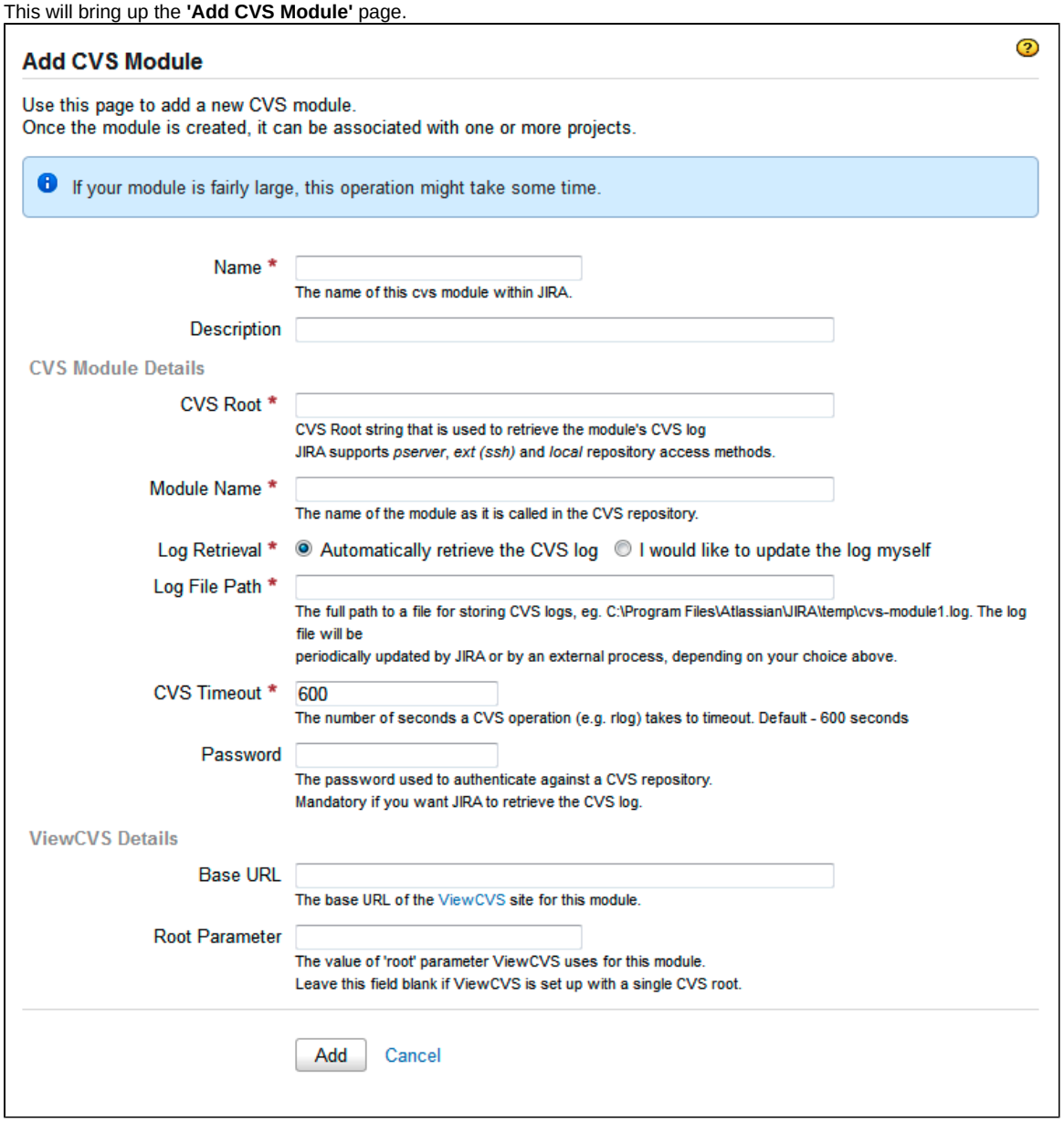
- Currently, JIRA is able to retrieve CVS log data via local access, pserver protocol or ssh (ext method). If your CVS is not reachable

by these methods you can disable automatic log retrieval (see below).
- If you would like JIRA to automatically keep synchronised with your CVS repository, the communication between JIRA and the CVS server might be fairly bandwidth intensive as JIRA will periodically retrieve the CVS module's log data from the CVS repository. If this is causing problems, consider adjusting the frequency (see below) or disabling CVS log retrieval.
- JIRA loads and parses the output of the 'cvs log' command for each CVS module and keeps 'relevant' commits in memory. Therefore JIRA's memory requirements depend on the number of relevant commits found in the CVS module. Relevant commits are CVS commits which have at least one potential JIRA key in their commit messages.
- Only commit messages which contain a possible JIRA issue key are linked to an issue.
- JIRA's 'System encoding' is used when parsing the CVS logs, so it needs to match that of the CVS log. The system encoding can be seen at Admin -> System -> System Info. See also how to set the system encoding.

Step 1. Create a CVS Module in JIRA

A CVS 'module' refers to a top-level directory in a CVS repository. To create a CVS module:

1. Create or decide which existing directory will be used to store CVS module's log data (The file with the output of the 'cvs log' command). JIRA must have read and write access to the directory. The write access is required even if you choose to update the CVS log manually as JIRA needs to use this directory to create a lock file in order to synchronise access to the CVS module's log.
2. Log in as a user with the **'JIRA System Administrators'** global permission.
3. Click the **'Administration'** link on the top bar.
4. Select **'Plugins' > 'CVS Modules'** from the top menu. This should bring up the **'CVS Modules'** page.
5. Click on the **'Add new CVS module'** link on this page.
6. This will bring up the **'Add CVS Module'** page.



Fill in as follows:

a. For **'Name'** put a short descriptive name, possibly just the name of the CVS module as it appears in your CVS repository.
b. (Optional) For **'Description'** put a short phrase that describes this CVS module.
c. Specify **'CVS Root'** that will be used to retrieve the CVS module's log or was used to retrieve the log. The CVS Root is needed while parsing the log data so it is required even if you choose to retrieve CVS log manually. Please provide 'full' CVS Root details. For example:
- /some/local/path (for local repository access)

- :pserver:username@hostname:port/some/path (for pserver access)
- :ext:username@hostname:/some/path (for ssh access)If JIRA finds trouble understanding your local CVS Root (e.g. on Windows systems) please prefix the path with *:local:*. For example, :local:d:\some\path.

a. For **'Module Name'** specify the name of the module as it is called in the CVS repository. This will usually be the top-level directory (eg. `myproject`), but can also include subdirectories (`myproject/subproject/src/java`) - basically anything that can be parsed to a `cvs checkout` command.This information is required to retrieve the CVS log as well as to parse it, so you will need to provide the module's CVS name even if you choose to retrieve the CVS log manually.
b. For **'Log Retrieval'** choose whether you would like JIRA to automatically synchonise with the CVS repository. If you choose **'Automatically retrieve the CVS log',** JIRA will periodically retrieve the CVS log for the module automatically and then parse it for commit information. If you choose **'I would like to update the log myself',** JIRA will not retrieve the log, but will periodically just parse it. If you choose this option you will need to update the CVS log by other means (e.g. manually or using a scheduled script) to keep the CVS information in JIRA current.
c. For **'Log File Path'** specify the full path to the **file** that will contain the CVS log data. This file should be located in a directory mentioned in step 1. If you would like JIRA to periodically update the contents of the log this file does not need to exist at the moment, as JIRA will automatically create it. If you choose to manually update the file please ensure that the log file already exists at the specified path and is readable by JIRA.
d. For **'CVS Timeout',** specify how many seconds it takes the CVS operation (e.g. rlog) to timeout
e. The **'Password'** needs to be provided only if you let JIRA automatically retrieve the module's CVS log. Please specify the password that is needed to retrieve the log using the method specified in the CVS Root. If no password is required, leave the field empty.
f. (Optional) For **'Base URL'** in the **'ViewCVS Details'**section of the page, enter the fully qualified URL (i.e. include 'http://' or 'https://' at the beginning) to the ViewCVS site of the CVS module. The URL needs to point to the root of the module on the ViewCVS site.

> If you are integrating with FishEye you do not need to perform any special steps. FishEye can resolve all the URLs that ViewCVS expects. You just need to enter the fully qualified URL to your FishEye installation and the specific repository you wish to view. This is the same URL you would get if you were to browse to the project within FishEye.

g. (Optional) For **'Root Parameter'** in the **'ViewCVS Details'** section of the page, enter the name of the Project Root that is used in ViewCVS to navigate the CVS module. This parameter is required only if ViewCVS has been set up to work with multiple CVS modules, and this module is not the default module on the ViewCVS server. The value that should be placed in this field is the same as the value of the 'root' URL parameter that appears on every ViewCVS URL (e.g. when viewing a file). If the URL that appears in your browser when viewing a file from this CVS module on ViewCVS does not have the 'root' parameter, leave this field blank.

7. Click the **'Add'** button.
8. This should bring you back to the **'CVS Modules'** page, where you should see the new CVS module listed. You can edit and delete this module here.

> If JIRA has trouble understanding your local CVS Root (e.g. on Windows systems) please prefix the path with :local:. For example, :local:d:\some\path

Step 2. Associate Project(s) with CVS Modules

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click the **'Administration'** link on the top bar.
3. Click **'Projects'** (in the top bar) and select the project that you would like to associate with the CVS module.
4. The project's summary page will be displayed. Next to **'CVS Modules'**, click the **'Change'** link. This will display the **'Select Version Control Modules'** page, where you can associate the project with a CVS module (or with multiple CVS modules).
5. Select the appropriate module(s), and click the **'Select'** button.

Step 3. Configuring Permissions

The 'View Version Control' permission needs to be given to users/groups/roles that should be allowed to see CVS commit information. Note: by default this permission is given to the 'jira-developers' group. Please read the Project Permissions section, and follow the instructions given there to assign the 'View Version Control' permission.

Disabling Automatic CVS Log Retrieval

To disable automatic CVS log retrieval for a CVS module please choose the **'I would like to update the log myself'** option for the module's **'Log Retrieval'** attribute.

If you have disabled automatic CVS log retrieval for the CVS module, JIRA will only *parse* the CVS log periodically. Therefore, for the new commit information to appear in JIRA, the log needs to be updated by other means. This can either be done manually, or a scheduled CVS update script can be used.

Before updating the module's CVS log, please check for the existence of a lock file with name **'cvslog.write.lock'** in the *same* directory as the CVS log file. If the lock file exists, please wait until it is removed before updating the log.

When updating the CVS log for a module, please create a lock file with the name **cvslog.write.lock** in the *same* directory as the CVS log file to ensure that JIRA does not start parsing the log while it is still being updated. Please do not forget to remove the lock file after the update has finished.

Adjusting the Frequency of Module Updates

To minimise the network traffic between JIRA and the CVS server, JIRA updates and re-parses the commit information of the associated CVS modules only once during the specified period of time. By default, this period of time is 1 hour, but it can be adjusted if required.

When the first CVS module is created in JIRA, a background service is automatically started. The service is called **'VCS Update Service'.** To change the frequency of the module updates, follow these steps:

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Click the **'Administration'** link on the top bar.
3. # Select **''System' > 'Advanced' > Services'**. A page showing all the configured services will appear. If at least one CVS module has been configured, the **'VCS Update Service'** should be present in the list.
4. Click the **'Edit'** link in the right-most column of the **'VCS Update Service'.** This will display a page where you can set the delay for the service.
5. Change the value as required. Remember that the delay is specified in minutes.
6. Click the **'Update'** button to make the changes take effect.

Please keep in mind:

- The CVS modules are updated one after another every specified period of time. That is, it is not possible to specify a different update delay for each configured CVS module.
- If you are using automatic log retrieval for your CVS modules and you set the delay to a very low value, the bandwidth consumption between JIRA and the CVS server might be very high.
- If the delay is set to a very large value, the 'new' cvs commit messages will not appear in JIRA for some time.

CVS Aliases

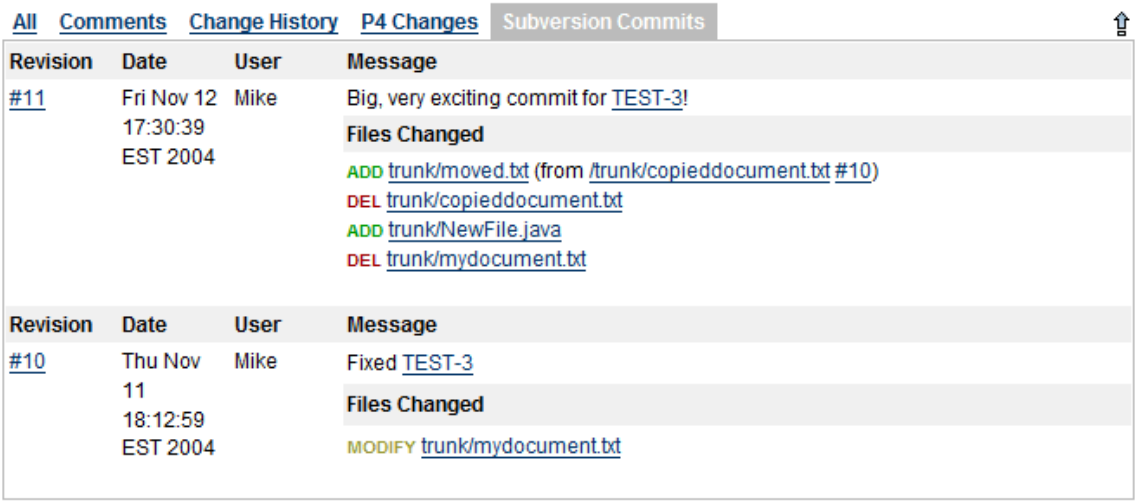
JIRA does not currently support CVS aliases. If you have a CVS alias that references more than one module, please create each CVS module in JIRA and then associate each module with the relevant JIRA project(s).

The feature request for adding CVS alias module support to JIRA is JRA-4586. Please vote for the issue to increase its popularity. Please refer to Implementation of New Features Policy which describes the way Atlassian implements new features and improvements.

Integrating JIRA with Subversion

JIRA's Subversion integration lets one see Subversion commit information relevant to each issue. Subversion integration can be implemented either by using Atlassian FishEye or the Subversion plugin (drop-in extension) mentioned below. The Fisheye integration offers greater scalability, insight and flexibility into your source code and related integration with JIRA but both are excellent to make sure that JIRA is connected to the related code changes.

*The information on this page does not apply to **JIRA OnDemand**.*



Commits will appear in this tab if the commit log mentions the issue key ('TEST-3' above).

For more information, see the Subversion plugin page online.

Integrating JIRA with Perforce

Perforce is supported by Atlassian FishEye, providing comprehensive integration with JIRA including real-time notifications of code changes plus web-based reporting, visualisation, search and code sharing. For details, please see Integrating JIRA with FishEye.

The original Perforce Plugin for JIRA is **deprecated** and has been superseded by the JIRA FishEye Plugin, which is included with JIRA and provides Perforce integration (see the documentation).

> *The information on this page does not apply to **JIRA OnDemand**.*

Integrating JIRA with ClearCase

ClearCase is supported by Atlassian FishEye, providing comprehensive integration with JIRA including real-time notifications of code changes plus web-based reporting, visualisation, search and code sharing. For details, please see Integrating JIRA with FishEye.

Alternatively, there is a JIRA ClearCase plugin which shows ClearCase checkins associated with JIRA issues. Please note that this plugin is not developed or supported by Atlassian.

> *The information on this page does not apply to **JIRA OnDemand**.*

Integrating with a Build Management System

JIRA integrates with Bamboo, Atlassian's Continuous Integration server.

- Integrating JIRA with Bamboo

> *The information on this page does not apply to **JIRA OnDemand**.*

Integrating JIRA with Bamboo

Integrating Atlassian's Bamboo with your JIRA server allows users to:

- add the Bamboo Charts gadget to their JIRA dashboards
- add the Bamboo Plan Summary gadget to their JIRA dashboards
- add the Bamboo Plans gadget to their JIRA dashboards
- browse a project's Bamboo builds
- browse a version's Bamboo builds
- view the Bamboo builds related to an issue
- trigger Bamboo builds when releasing a JIRA version

For full details on how to install the Bamboo plugin, please see the Bamboo documentation on Integrating Bamboo with JIRA.

*Some functionality described on this page is restricted in **JIRA OnDemand**.*

Configuring Global Settings

- Configuring Time Tracking
- Configuring JIRA Options
- Setting Properties and Options on Startup
  - Recognized System Properties for JIRA
- Advanced JIRA Configuration
  - Changing the constraints on historical time parameters in gadgets
  - Changing the Default Order for Comments from Ascending to Descending
  - Limiting the number of issues returned from a search view such as an RSS feed
- Configuring File Attachments
- Configuring Application Links
  - Adding an Application Link
  - Configuring Authentication for an Application Link
    - Configuring Basic HTTP Authentication for an Application Link
    - Configuring OAuth Authentication for an Application Link
    - Configuring Trusted Applications Authentication for an Application Link
    - Incoming and Outgoing Authentication
  - Editing an Application Link
  - Making an Application Link the Primary Link
  - Relocating an Application Link
  - Upgrading an Application Link
  - Deleting an Application Link
  - Configuring Project Links across Applications
    - Adding Project Links between Applications
    - Making a Project Link the Primary Link
    - Deleting a Project Link
- Configuring Issue Cloning
- Configuring Issue Linking
- Configuring the Whitelist
- Configuring Sub-tasks
- Managing Shared Filters
- Managing Shared Dashboards

Configuring Time Tracking

JIRA's Time Tracking feature enables users to record the time they spend working on issues (see Logging Work on an Issue).

Note: Before users can specify time estimates and log work, they must be granted the **Work On Issues** permission for the relevant project(s).

Disabling Time Tracking

Time Tracking is ON by default (as shown in screenshot 1 below). However, this feature can be disabled from the **Time Tracking** administration page.
Time tracking will be OFF by default if your JIRA installation was upgraded from a version prior to 4.2 that had time tracking either disabled or never enabled.

To disable Time Tracking:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'System' > 'Issue Features' > 'Time Tracking'** (tab) to open the 'Time Tracking' page.
   Keyboard shortcut: **'g' + 'g' +** type **'time t'**
3. Click the '**Dectivate**' button to turn Time Tracking OFF.

You will not lose any existing Time Tracking data by disabling/re-enabling Time Tracking.

On this page:

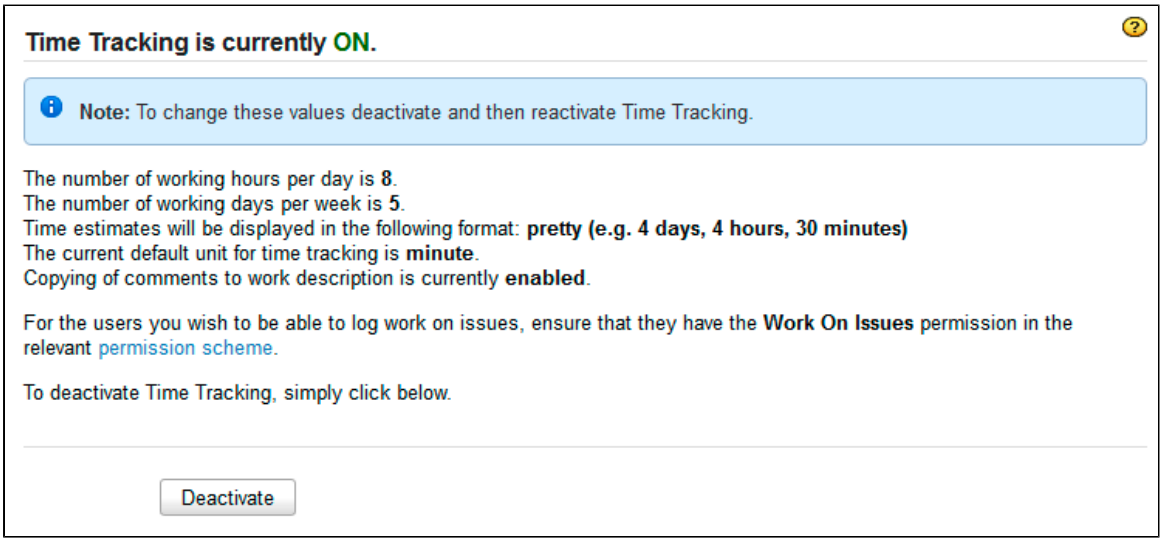
- Disabling Time Tracking
- Enabling Time Tracking
- Configuring Time Tracking Settings
- About 'Legacy Mode'
- Related Topics

Enabling Time Tracking

To enable Time Tracking:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'System' > 'Issue Features' > 'Time Tracking'** (tab) to open the 'Time Tracking' page.
   Keyboard shortcut: **'g' + 'g' +** type **'time t'**
3. Click the '**Activate**' button to turn time tracking ON.

Screenshot 1: Time Tracking is ON



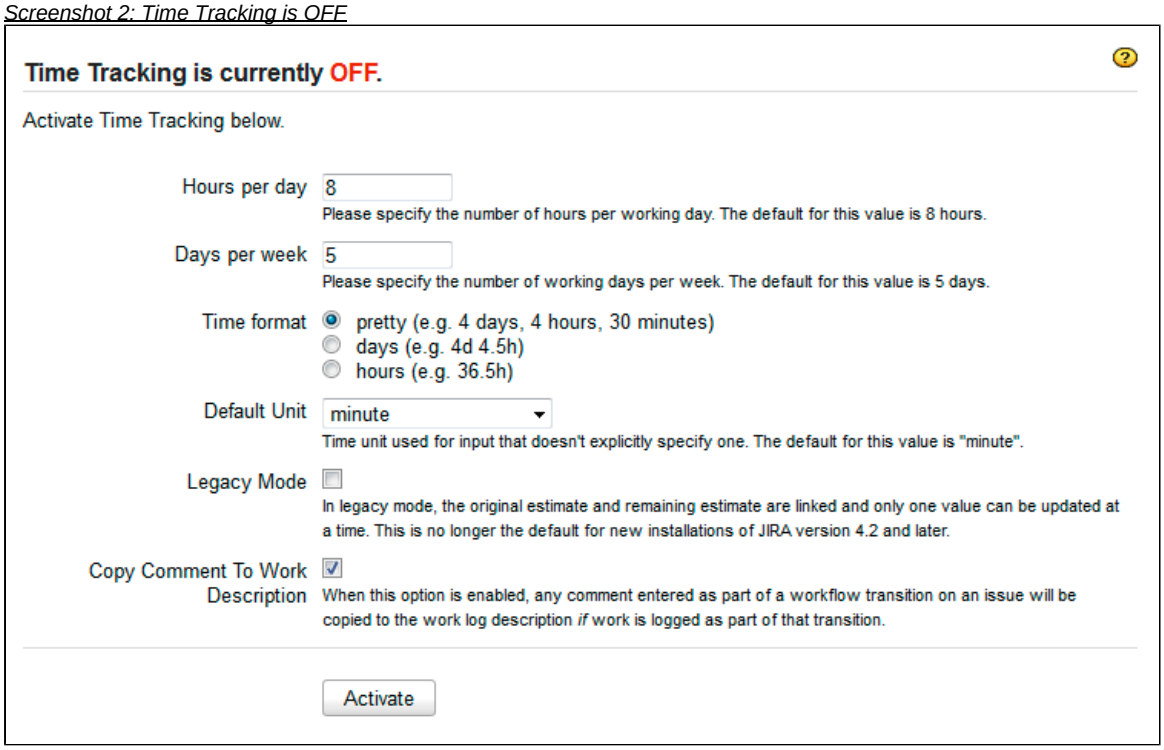
Configuring Time Tracking Settings

To edit JIRA's **Time Tracking** settings, it must first be disabled. Once you have changed the settings, you will then need to re-enable Time Tracking so that users can log work on issues.

You will not lose any existing Time Tracking data by disabling/re-enabling Time Tracking.

To configure Time Tracking settings:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'System' > 'Issue Features' > 'Time Tracking'** (tab) to open the 'Time Tracking' page.
   Keyboard shortcut: **'g' + 'g' +** type **'time t'**
3. If Time Tracking is ON (refer to the indication at the top of the **Time Tracking** screen), click the '**Deactivate**' button to turn Time Tracking OFF.
4. The Time Tracking settings will now be editable as shown in the following screenshot.
   Screenshot 2: Time Tracking is OFF



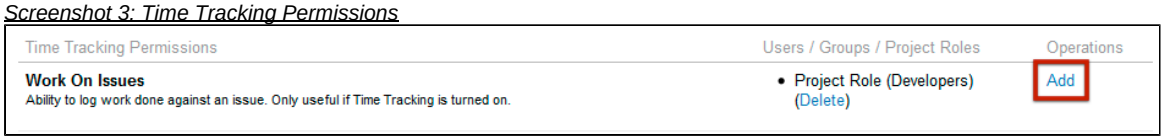
5. Configure Time Tracking settings by editing the following fields:
   - 'Hours per day' — enter a suitable value (e.g. 8). You can enter fractions if you wish.

- 'Days per week' — enter a suitable value (e.g. 5). You can enter fractions if you wish.
- 'Time format' — select **pretty/days/hours**. This will determine the format of the 'Time Spent' field when an issue is displayed.
- 'Default Unit' — select **minutes/hours/days/weeks**. This will be applied whenever your users log work on an issue without specifying a unit.
- 'Legacy Mode' — select this check box if you prefer to use JIRA's time tracking features as they operated prior to JIRA version 4.2. For more details about this option, please see About 'Legacy Mode' (below).
- 'Copy Comment To Work Description' — select this check box to ensure that any content entered into a **Comment** field while logging work as part of an issue operation, is also copied across to the **Work Description**.

  When 'Copy Comment To Work Description' is enabled, your user's work log entries will be visible only to members of the project role or group selected in the padlock icon dropdown on their issue operation screen. If 'Copy Comment To Work Description' is disabled, your user's work log entries will be visible to anyone by default.

6. Click the '**Activate**' button to turn time tracking ON.
   If the permission schemes used by your project(s) already have the appropriate **Work On Issues** permissions, then there is no need to proceed any further.
   However, if you need to configure these permissions, proceed with the remaining steps below:
7. Click the '**permission scheme**' link as shown in screenshot 1 (above). The 'Permissions Scheme' page will be displayed.
8. Click the '**Permissions**' link of the permission scheme associated with the project(s) where you wish to specify **Work On Issues** permissions. The 'Edit Permissions' page is displayed for your chosen permission scheme.

   See Managing Project Permissions for details about the various permissions.
9. Check whether the row labelled 'Work On Issues' contains the appropriate users, groups or project roles who need to specify time estimates or log work. If it does not, click the '**Add**' link in the 'Operations' column:
   Screenshot 3: Time Tracking Permissions



10. Select the users, groups or project roles to whom you want to allow time tracking and work logging on issues.
11. Click the '**Add**' button.

About 'Legacy Mode'

- If Legacy Mode is disabled, your users will be able to change the **Original Estimate** value *irrespective of any work being logged on an issue*. Legacy Mode is disabled by default on new installations of JIRA version 4.2 or later.
- If Legacy Mode is enabled, your users can only specify an **Original Estimate** *before* they start logging work on an issue. This value cannot be changed once *any* work has been logged, unless all work logs for that issue are first deleted.
- By default,
  - **Legacy Mode** is disabled if your JIRA 4.2 installation was conducted cleanly (that is, without upgrading from an earlier version of JIRA).
  - **Legacy Mode** is enabled if you upgraded JIRA from a version prior to 4.2.
- Please refer to the Logging Work on an Issue and the JIRA 4.2 Release Notes for more information about logging work and modifying time estimates.

Related Topics

- Please see the section Adding Time Tracking capabilities to a screen on the Defining a Screen page.

Configuring JIRA Options

JIRA has a number of configuration options that allow your JIRA server to be customised for use within your organisation. These options can be accessed and edited on JIRA's 'General Configuration' page.

Editing JIRA's General Configuration

To access and edit options on the 'General Configuration' page:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > General Configuration** to open the 'General Configuration' page as shown in Screenshot 1 (below).
   Keyboard shortcut: **g + g +** start typing **'general configuration'**
3. Scroll to the end of the page and click the **Edit Configuration**button to edit the three sections as described below:
   - Settings
   - Internationalisation
   - Options

On this page:

- Editing JIRA's General Configuration
  - Settings
  - Internationalisation
  - Options
- Configuring Advanced Settings
- See Also

Screenshot 1: General Configuration

Settings

Title	Example Company JIRA
Mode	Public
Maximum Authentication Attempts Allowed	3
CAPTCHA on signup	ON
Base URL	http://blowfly:8080
Email from	${fullname} (JIRA)
Introduction	

Internationalisation

Additional language packs can be installed through the Plugins tab above. Atlassian Translations is collaborative effort to localise Atlassian products, we welcome your contribution.

Indexing language	english
Installed languages	Catalan (Spain) Chinese (China) Chinese (Taiwan) Czech (Czech Republic) Danish (Denmark) Dutch (Belgium) English (UK) English (United States) French (France) German (Germany) German (Switzerland) Hungarian (Hungary) Italian (Italy) Japanese (Japan) Norwegian (Norway) Polish (Poland) Portuguese (Brazil) Russian (Russia) Slovak (Slovakia) Spanish (Spain) Turkish (Turkey)
Default language	English (Australia)
Default user time zone	System default: (GMT+10:00) Sydney

Options

Allow users to vote on issues	ON
Allow users to watch issues	ON
Allow unassigned issues	OFF
External user management	OFF
Logout confirmation	Never
Use gzip compression	OFF
Accept remote API calls	ON
User email visibility	Public
Comment visibility	Project Roles only
Exclude email header "Precedence: bulk"	OFF
Issue Picker Auto-complete	ON
User Searching By Full Name	ON
JQL Auto-complete	ON
Internet Explorer MIME Sniffing Security Hole Workaround Policy	Work around Internet Explorer security hole
Contact Administrators Form	OFF
Contact Administrators Message	

Edit Configuration

Settings

Setting	Description

Title	This is the title that will be displayed on the JIRA login page and the dashboard. It helps identify your installation and its purpose. Also see logo, which is displayed on every JIRA page.
Mode	JIRA can operate in two modes: - **Public** — Anyone can sign themselves up and create issues (within the bounds of your JIRA system's permissions). - **Private** — Useful for internal issue-tracking systems where you do not want public users to login. Self-signup is disabled; only Administrators can create new users. *Default:* ***Public***
Maximum Authentication Attempts Allowed	The maximum authentication attempts that are allowed before CAPTCHA is shown to a user. If you leave it blank then CAPTCHA will never be shown and users will have unlimited authentication attempts. It is recommended that you set this to a small number (e.g. below 5). *Default:* ***3*** *(for new installations of JIRA)* If you upgrade from a version of JIRA prior to 4.1.1, this option will be reset to 3 for security reasons mentioned in the JIRA 4.1.1 Upgrade Guide.
CAPTCHA on signup	If you are running JIRA in Public mode (see above), it is strongly recommended that you enable CAPTCHA. This will show a CAPTCHA image on signup to prevent spambots from signing up. *Default:* ***ON***
Base URL	The base URL of this JIRA installation. You can only configure JIRA to respond to a single URL and this setting must match the URL that your users request for accessing your JIRA site. You cannot (for example) have a different hostname or URL for internal and external users. This is especially important for JIRA 4.0.x or higher, as any mismatch between this Base URL setting and the URL requested by your JIRA users will cause problems with dashboard gadgets. This URL is also used in outgoing email notifications as the prefix for links to JIRA issues.
Email from	Specifies the **From:** header format in notification emails. Default is of the form "John Doe (JIRA) <jira@company.com>". Available variables are '${fullname}', '${email}' and '${email.hostname}'. Note that the actual address (e.g. 'jira@company.com') cannot be specified here - it is determined by the mail server or individual project configuration.
Introduction	A short introduction message displayed on the dashboard. Also see the announcement banner, which is displayed on *every* JIRA page. You can include HTML, but ensure all tags are correctly closed.

Internationalisation

Setting	Description
Indexing language	JIRA uses Lucene, a high-performance text search engine library, in full-text searches over the issues stored in JIRA. Indexing and searching for issues is affected by the language that the issues are entered in. Please choose the language that issues are entered in. If multiple languages are chosen, choose 'Other'. Note: You will need to re-index JIRA if you change this value.
Installed languages	This section lists all language packs available within the JIRA system. (Note: to install additional languages, see Internationalisation.)
Default language	The language used throughout the JIRA interface (as selected from the list displayed in **Installed Languages** above). Users can override the default language by using the **Language** setting in their user profile.
Default user time zone	This is the time zone used throughout the JIRA interface. Users can override the default language by using the **Time Zone** setting in their user profile. (To choose the time *format* see Configuring the Layout and Design.) Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information) so are not affected by time zone settings.

Options

Setting	Description

Allow users to vote on issues	Controls whether voting is enabled in JIRA. Voting allows users to indicate a preference for issues they would like to be completed or resolved. See also the 'View Voters and Watchers' permission. *Default:* ***ON***
Allow users to watch issues	Controls whether watching is enabled in JIRA. Users can 'watch' issues which they are interested in. Users watching an issue will be notified of all changes to it. See also the 'View Voters and Watchers' and 'Manage Watcher List' permissions. *Default:* ***ON***
Allow unassigned issues	When turned **ON**, JIRA will allow issues to be unassigned or assigned to 'no-one'.When turned **OFF**, issues must always be assigned to someone - by default, the assignee will be the **Project Lead** as defined for each project. *Default:* ***OFF***
External user management	When turned **ON**, you will no longer be able to create, edit or delete users/groups from within JIRA (or via email or import); but you can still assign users/groups to project roles, and create/edit/delete user properties. Additionally, JIRA will not display options for users to change their password, or edit their profile. Generally you would only turn this **ON** if you are managing all your users from outside JIRA (e.g. using Crowd, Microsoft Active Directory or another LDAP directory). *Default:* ***OFF***
Logout confirmation	Controls whether to obtain user's confirmation when logging out: **NEVER COOKIE** - prompt for confirmation if the user was automatically logged in (via a cookie). **ALWAYS** *Default:* ***NEVER***
Use gzip compression	Controls whether to compress the web pages that JIRA sends to the browser. It is recommended that this be turned ON, unless you are using mod_proxy. *Default:* ***OFF***
Accept remote API calls	Controls whether to allow remote client access (via XML-RPC or SOAP) to this JIRA installation, for authenticated users. *Default:* ***OFF***
User email visibility	Controls how users' email addresses are displayed in the user profile page. - **PUBLIC** - email addresses are visible to all. - **HIDDEN** - email addresses are hidden from all users. - **MASKED** - the email address is masked (e.g. 'user@example.com' is displayed as 'user at example dot com'). - **LOGGED IN USERS ONLY** - only users logged in to JIRA can view the email addresses. *Default:* ***PUBLIC***
Comment visibility	Determines what will be contained in the list that is presented to users when specifying comment visibility and worklog visibility. - **Groups & Project Roles** - the list will contain groups and project roles. - **Project Roles only** - the list will only contain project roles. *Default:* ***Project Roles only***
Exclude email header 'Precedence: bulk'	Controls whether to prevent the **Precedence: Bulk** header on JIRA notification emails. This option should only be enabled when notifications go to a mailing list which rejects 'bulk' emails. In normal circumstances, this header prevents auto-replies (and hence potential mail loops). *Default:* ***OFF***
Issue Picker Auto-complete	Provides auto-completion of issue keys in the 'Issue Picker' popup screen. Turn OFF if your users' browsers are incompatible with AJAX. *Default:* ***ON***
JQL Auto-complete	Provides auto-completion of search terms when users perform an advanced (JQL) search. Turn OFF if you prefer not to use this feature, or are experiencing a performance impact. *Default:* ***ON***

Internet Explorer MIME Sniffing Security Hole Workaround Policy	Attachment viewing security options for cross-site site scripting vulnerabilities present in Internet Explorer 7 and earlier. Changes the default browser action for attachments in JIRA. Options are: - **Insecure: inline display of attachments** - allows all attachments to be displayed inline. Only select this option if you fully understand the security risks. See JIRA Security Advisory 2008-08-26 for further details. - **Secure: forced download of all attachments for all browsers** - force the download of all attachments. This is the most secure option, but is less convenient for users. - **Work around Internet Explorer security hole** - forced download of high-risk attachments (IE-only Workaround) - for IE browsers, force the download of attachments that IE would mistakenly detect as an HTML file. Declared HTML attachments are also never displayed inline. Use this option to reduce the risk of attacks to IE users via attachments. *Default:* ***Work around Internet Explorer security hole***
Contact Administrators Form	Provides an email form for users to fill in when they click the 'Contact Administrators' link (which appears when necessary throughout the JIRA interface, e.g. on the Login screen). Applies only if outgoing email is enabled. Can be used with or without the custom 'Contact Administrators Message' below. *Default:* ***OFF***
Contact Administrators Message	Displays a custom message when users click the 'Contact Administrators' link (which appears when necessary throughout the JIRA interface, e.g. on the Login screen). The 'Contact Administrators Message' will be displayed at the top of the 'Contact Administrators Form', if the form is enabled (see above), or by itself if the form is not enabled.

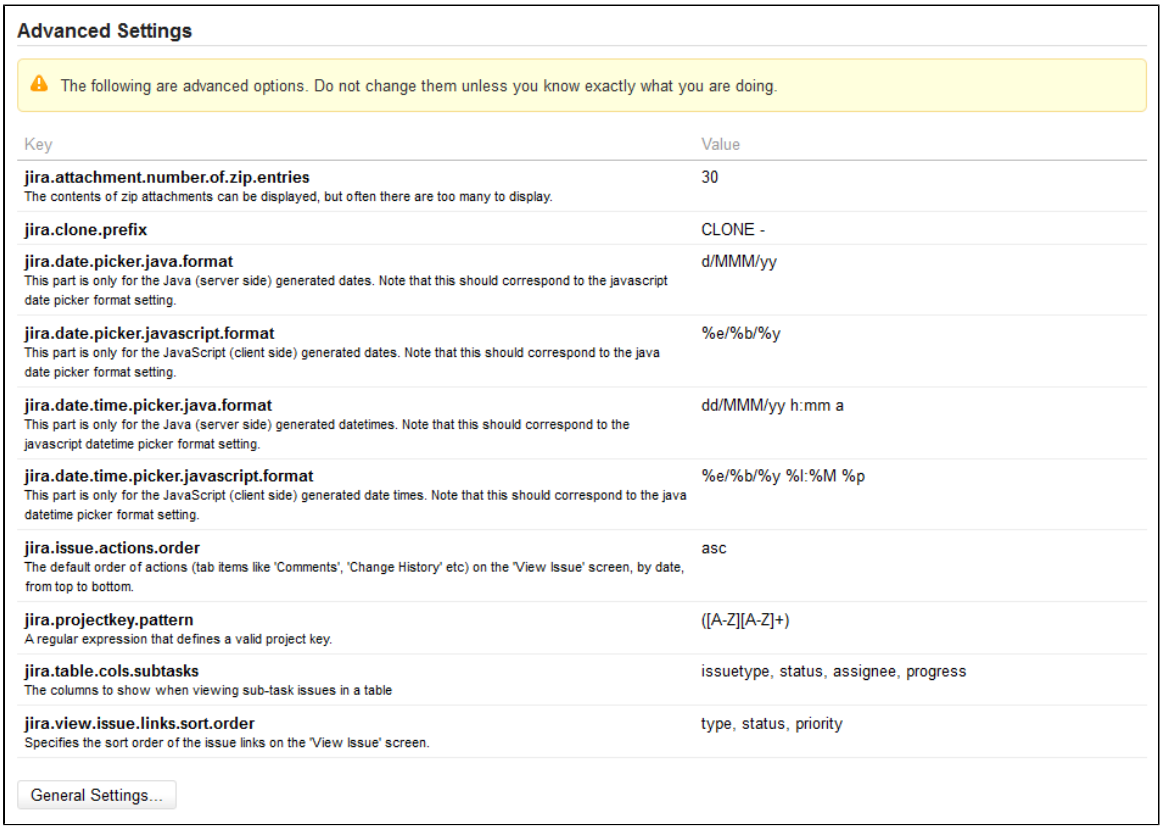
Configuring Advanced Settings

JIRA has a small number of commonly edited advanced configuration options, which are stored in the JIRA database. These options can be accessed and edited from the 'Advanced Settings' page.

To access and edit options on the 'Advanced Settings' page:

1. Click the '**Advanced**' button at the end of the 'General Configuration' page to display the 'Advanced Settings' page as shown in Screenshot 2 (below).
2. Edit the value of a property/key by clicking its value on the right of the page and modifying the existing value.
3. Click the '**Update**' button on the right to save the new value in the JIRA database.
   Any changes you make to these properties/keys become effective immediately.

Screenshot 2: Advanced Settings



See Also

There are a handful of other advanced configuration options (which are of little interest to most JIRA system administrators) whose default values can be customised in the `jira-config.properties` file located in the JIRA Home Directory, which you may want to edit. For details, please see Advanced JIRA configuration.

Setting Properties and Options on Startup

This page describes how to set Java properties and options on startup for JIRA.

On this page:

- Linux
- Windows (starting from .bat file)
- Windows Service
  - Setting Properties for Windows Services via Command Line
  - Setting Properties for Windows Services via the Windows Registry
- Verifying Your Settings
- List of Startup Parameters

Linux

To Configure System Properties in Linux Installations,

1. From `<jira-install>/bin` (or `<Tomcat-home>/bin` for JIRA WAR installations), open **setenv.sh**.
2. Find the section **JVM_SUPPORT_RECOMMENDED_ARGS=**
3. Refer to the list of parameters below.

Add all parameters in a space-separated list, inside the quotations.

Windows (starting from .bat file)

To Configure System Properties in Windows Installations When Starting from the .bat File,

1. From `<jira-install>/bin` (or `<Tomcat-home>/bin` for JIRA WAR installations), open **setenv.bat**.
2. Find the section **set JVM_SUPPORT_RECOMMENDED_ARGS=**
3. Refer to the list of parameters below.

Add all parameters in a space-separated list, inside the quotations.

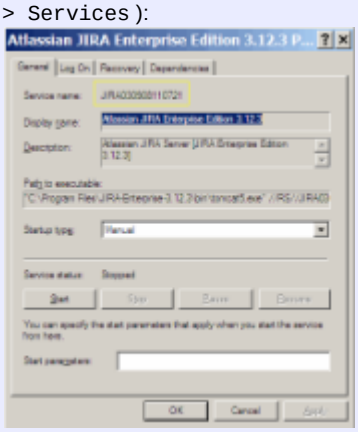
Windows Service

There are two ways to configure system properties when starting Running JIRA as a Service, either via command line or in the Windows Registry

Setting Properties for Windows Services via Command Line

Setting Properties for Windows Services via Command Line

1. Identify the name of the service that JIRA is installed as in Windows (`Control Panel > Administrative Tools > Services`):

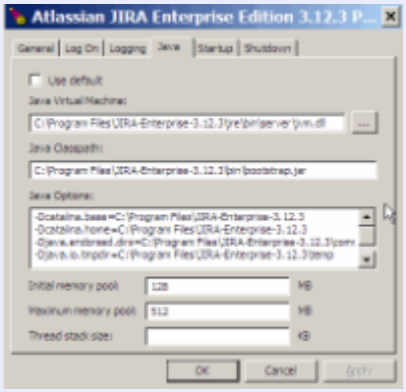


   In the above example, the **SERVICENAME** is: `JIRA030908110721`
2. Open the command window from `Start >> Run >> type in 'cmd' >> Enter`
3. `cd` to the `bin` directory of your JIRA Installation Directory (or the `bin` directory of your Tomcat installation if your are running JIRA WAR).
4. Run:

```
tomcat6w //ES//%SERVICENAME%
```

   In the above example, it would be `tomcat6w //ES//JIRA030908110721`

5. Click on the `Java` tab to see the list of current start-up options:



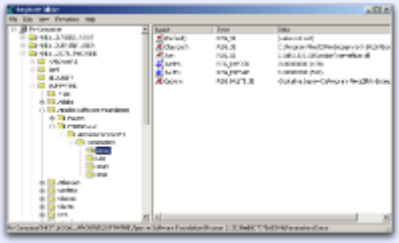
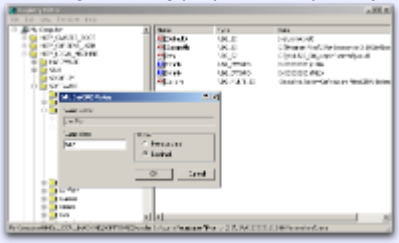
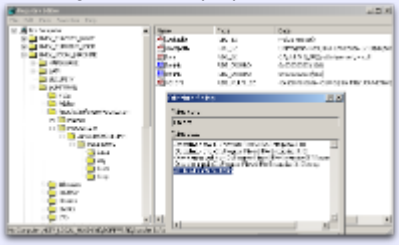
6. Append any new option on its own new line by adding to the end of the existing Java Options. Refer to the list of parameters below.

Setting Properties for Windows Services via the Windows Registry

In some versions of Windows, there is no option to add Java variables to the service. In these cases, you must add the properties by viewing the option list in the registry.

To Set Properties for Windows Services via the Windows Registry,

1. Go to {{Start >> Run, and run "regedit32.exe".

2. Find the Services entry:
32-bit: `HKEY_LOCAL_MACHINE >> SOFTWARE >> Apache Software Foundation >> Procrun 2.0 >> JIRA`
64-bit: `HKEY_LOCAL_MACHINE >> SOFTWARE >> Wow6432Node >> Apache Software Foundation >> Procrun 2.0 >> JIRA`

3. To change existing properties, especially increasing Xmx memory, double-click the appropriate value.

4. To change additional properties, double-click options.

5. Refer to the list of parameters below. Enter each on a separate line.

Verifying Your Settings

To verify what settings are in place, check the `<jira-home>/logs/atlassian-jira.log` or `catalina.out` file. A section in the startup appears like this:

```
JVM Input Arguments : -Djava.util.logging.config.file=/usr/local/jira/conf/logging.properties
-XX:MaxPermSize=256m -Xms256m -Xmx384m -Djava.awt.headless=true -Datlassian.standalone=JIRA
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true -Dmail.mime.decodeparameters=true
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djava.endorsed.dirs=/usr/local/jira/endorsed -Dcatalina.base=/usr/local/jira
-Dcatalina.home=/usr/local/jira -Djava.io.tmpdir=/usr/local/jira/temp
```

This display is also available by Viewing your System Information.

List of Startup Parameters

Memory Property	Notes	Related Pages

-Xmx -Xms XX:MaxPermSize	These properties are pre-existing. See related pages for instructions.	Increasing JIRA Memory
-XX:+PrintGCTimeStamps -verbose:gc -Xloggc:gc.log -XX:+HeapDumpOnOutOfMemoryError	Set these for Garbage Collection tuning.	Garbage Collection Performance Problems Using Memory Dumps to Analyze OutOfMemoryErrors
-agentlib:yjpagent=onexit=memory,dir=/path/to/write/snapshots		Profiling Memory and CPU Usage with YourKit
Mail Property	**Notes**	**Related Pages**
-Datlassian.mail.senddisabled -Datlassian.mail.fetchdisabled -Datlassian.mail.popdisabled	Set to 'true' to disable mail. In Linux setenv.sh, there is a pre-existing flag to uncomment.	Migrating JIRA to Another Server Notifications Are Issued for Incorrect Issues
-Dmail.debug	If set to "true", logs statements related to mail	Configuring JIRA's SMTP Mail Server to Send Notifications Creating Issues and Comments from Email
-Dmail.mime.decodetext.strict		Unable to Decode Mail Subject or Body when Creating Issue From Email
-Dmail.imap.auth.plain.disable -Dmail.imaps.auth.plain.disable		Authenticate Failed Error when Connecting to Exchange
-Dmail.imap.starttls.enable		'javax.mail.MessagingException No login methods supported' Due to IMAP over SSL
-Dmail.mime.decodeparameters	Sets mail handler to work correctly with emails from RFC 2231-compliant mail clients.	Installing JIRA WAR-EAR Installing JIRA on Tomcat 5.5 JIRA 4.0 Upgrade Guide
-Dmail.smtp.localhost		Problems Sending Email from JIRA - EHLO requires domain address
Encoding Property	**Notes**	**Related Pages**
-Dfile.encoding	Set to utf-8 for encoding consistency	Integrating JIRA with CVS and ViewCVS Characters Not Supported by ASCII are Being Displayed as Question Marks Internalisation and Encoding Troubleshooting SQL Exception when Entering, Updating or Importing an Issue in JIRA with MySQL Due to Encoding International Characters in Notification Email Subject Lines Are Being Replaced with Question Mark
Other Properties	**Notes**	**Related Pages**
-Duser.timezone		Incorrect Times Displayed in JIRA

-Dsvnkit.http.methods	Values include Basic,Digest,Negotiate,NTLM	JIRA Startup Fails Due to 'java.lang.SecurityException Unable to locate a login configuration' Subversion Plugin Displays 'An unknown error occurred - actions == null' Due to SVN Authentication
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER	false	OutOfMemory Due to Tomcat Memory Leak JRA-10145
-ea/-da	Enable/Disable assertions	java.lang.AssertionError When Sending Mail Via SMTP
-Djava.net.preferIPv4Stack		SocketException to Announce 'Invalid argument' for an Available Port
-Djavax.net.ssl.trustStore		Connecting to SSL services Unable to Send Email 'javax.net.ssl.SSLException' Due to SMTP Server via SSL
-Djava.awt.headless	Ships with true by default. Allows thumbnail generation.	

Recognized System Properties for JIRA

JIRA supports some configuration and debugging settings that can be enabled through Java system properties. System properties are usually set by passing the `-D` flag to the Java virtual machine in which JIRA is running. See Setting Properties and Options on Startup.

List of Startup Parameters

Memory Property	**Notes**	**Related Pages**
-Xmx -Xms XX:MaxPermSize	These properties are pre-existing. See related pages for instructions.	Increasing JIRA Memory
-XX:+PrintGCTimeStamps -verbose:gc -Xloggc:gc.log -XX:+HeapDumpOnOutOfMemoryError	Set these for Garbage Collection tuning.	Garbage Collection Performance Problems Using Memory Dumps to Analyze OutOfMemoryErrors
-agentlib:yjpagent=onexit=memory,dir=/path/to/write/snapshots		Profiling Memory and CPU Usage with YourKit
Mail Property	**Notes**	**Related Pages**
-Datlassian.mail.senddisabled -Datlassian.mail.fetchdisabled -Datlassian.mail.popdisabled	Set to 'true' to disable mail. In Linux setenv.sh, there is a pre-existing flag to uncomment.	Migrating JIRA to Another Server Notifications Are Issued for Incorrect Issues
-Dmail.debug	If set to "true", logs statements related to mail	Configuring JIRA's SMTP Mail Server to Send Notifications Creating Issues and Comments from Email
-Dmail.mime.decodetext.strict		Unable to Decode Mail Subject or Body when Creating Issue From Email

-Dmail.imap.auth.plain.disable -Dmail.imaps.auth.plain.disable		Authenticate Failed Error when Connecting to Exchange
-Dmail.imap.starttls.enable		'javax.mail.MessagingException No login methods supported' Due to IMAP over SSL
-Dmail.mime.decodeparameters	Sets mail handler to work correctly with emails from RFC 2231-compliant mail clients.	Installing JIRA WAR-EAR Installing JIRA on Tomcat 5.5 JIRA 4.0 Upgrade Guide
-Dmail.smtp.localhost		Problems Sending Email from JIRA - EHLO requires domain address
Encoding Property	**Notes**	**Related Pages**
-Dfile.encoding	Set to utf-8 for encoding consistency	Integrating JIRA with CVS and ViewCVS Characters Not Supported by ASCII are Being Displayed as Question Marks Internalisation and Encoding Troubleshooting SQL Exception when Entering, Updating or Importing an Issue in JIRA with MySQL Due to Encoding International Characters in Notification Email Subject Lines Are Being Replaced with Question Mark
Other Properties	**Notes**	**Related Pages**
-Duser.timezone		Incorrect Times Displayed in JIRA
-Dsvnkit.http.methods	Values include Basic,Digest,Negotiate,NTLM	JIRA Startup Fails Due to 'java.lang.SecurityException Unable to locate a login configuration' Subversion Plugin Displays 'An unknown error occurred - actions == null' Due to SVN Authentication
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER	false	OutOfMemory Due to Tomcat Memory Leak JRA-10145
-ea/-da	Enable/Disable assertions	java.lang.AssertionError When Sending Mail Via SMTP
-Djava.net.preferIPv4Stack		SocketException to Announce 'Invalid argument' for an Available Port
-Djavax.net.ssl.trustStore		Connecting to SSL services Unable to Send Email 'javax.net.ssl.SSLException' Due to SMTP Server via SSL
-Djava.awt.headless	Ships with true by default. Allows thumbnail generation.	

Advanced JIRA Configuration

JIRA has a number of advanced configuration options, each of which is defined as an individual property (or 'key') associated with a value. These key-value pairs are stored in one of three areas for use by JIRA:

- The JIRA Database
- The `jira-config.properties` file
- The `jpm.xml` file

The JIRA Database

The values of a small number of most commonly edited advanced configuration options are stored in the JIRA database. These values can be edited from the 'Advanced Settings' page of JIRA's administration area (accessed from the **'System' > 'General Configuration'** menu > **'Advanced'** button). Once any of these properties' values are changed, they become effective immediately. See Configuring Advanced Settings for details.

The `jira-config.properties` file

Custom values for JIRA's remaining advanced configuration options (i.e. not stored in the JIRA database) are stored as individual key-value pairs in a file called `jira-config.properties` (located in the JIRA Home Directory). Typically, these options are of little interest to most JIRA system administrators. While these key-value pairs can be edited, JIRA must be restarted for any changed values to take effect.

In new JIRA installations, this file may not initially exist and if so, needs to be created manually.

The `jpm.xml` file

Default values for all* of JIRA's available advanced configuration options are stored in a file called `jpm.xml` (located in the `<jira-application-dir>/WEB-INF/classes` subdirectory of the JIRA Installation Directory). These default values are only used by JIRA if a property's value has not already been customised in either the JIRA database (via JIRA's 'Advanced Settings' page) or the `jira-config.properties` file.

The `jpm.xml` file should not be edited because any values that you customise in it will not be migrated automatically during subsequent JIRA upgrades. To change the value of a property for an advanced configuration option in JIRA, override the value of this property by redefining it in either:

- The JIRA database (via JIRA's 'Advanced Settings' page).
  OR
- The `jira-config.properties` file.

* JIRA recognises a small number of properties, which can be set in your `jira-config.properties` file *but have no definition in the `jpm.xml` file*. These properties:

- typically represent advanced configuration options that are disabled when they are not defined in your `jira-config.properties` file and
- when not specified in your `jira-config.properties` file, typically affect JIRA's behaviour differently to when they are specified in your `jira-config.properties` file *with no value*.

> *The information on this page does not apply to **JIRA OnDemand**.*

Making changes to the `jira-config.properties` file

To make changes to the `jira-config.properties` file:

1. Shut down JIRA (for example, by executing either the `/bin/stop-jira.sh` or `\bin\stop-jira.bat` file in your JIRA Installation Directory, or by stopping the JIRA service).
2. Open the `jira-config.properties` file (located at the root of your JIRA Home Directory) in a text editor.
   This file may not exist if you are using a new JIRA installation or an upgraded JIRA installation where your previous JIRA version(s) had never been customised. If this file does not exist, create it using a text editor.
3. Edit the appropriate properties in this file.
   Editing tips:
   - To determine the default value of a property whose value you wish to redefine, search for that property in the `<jira-application-dir>/WEB-INF/classes/jpm.xml` file (of your JIRA Installation Directory). The default value is defined in the `<default-value/>` sibling element of the relevant property's `<key/>` element.
   - To override a property's default value in `jpm.xml` (which is not already defined in your `jira-config.properties` file or available on the 'Advanced Settings' page):
     a. Copy the value of the relevant property's `<key/>` element from the `jpm.xml` file to the `jira-config.properties` file.
     b. In the `jira-config.properties` file, add an '=' after that property's key, followed by your custom value.
   - To disable a custom property's value in the `jira-config.properties` file, either 'comment out' the property with a preceding '#' symbol or remove the property from the file.

4. Save your modifications to the `jira-config.properties` file.
5. Restart JIRA.

See also

Setting Properties and Options on Startup — for changes like setting available memory, disabling email, enabling Jelly, etc.

Changing the constraints on historical time parameters in gadgets

A number of JIRA gadgets show historical data from your JIRA server.

You can generally configure the time constraints on this data via gadget parameters, such as those parameters defining how far back should data be retrieved. For instance, the 'Time Since Issues' gadget allows you to specify how far back issue data should be retrieved via the 'Days Previously' parameter.

> *The information on this page does not apply to **JIRA OnDemand**.*

For performance reasons, however, the JIRA server can impose an overriding maximum limit on historical data retrieved by gadgets. Hence, if you tried entering a 'Days Previously' value greater than 300 in the 'Time Since Issues' gadget, a validation message will be shown. You will not be permitted to save your configuration changes without changing the 'Days Previously' value to a lower one.

These maximum limits imposed by the JIRA server are defined by the following advanced configuration options in JIRA and can be customised in your `jira-config.properties` file (located in the JIRA Home Directory).

```
jira.chart.days.previous.limit.yearly=36500
jira.chart.days.previous.limit.quarterly=22500
jira.chart.days.previous.limit.monthly=7500
jira.chart.days.previous.limit.weekly=1750
jira.chart.days.previous.limit.daily=300
jira.chart.days.previous.limit.hourly=10
```

To update these properties:

1. Shut down your JIRA server.
2. Edit your `jira-config.properties` file in your JIRA Home Directory.
   See Making changes to the `jira-config.properties` file for more information.
3. Locate these properties.
   If any of these properties do not exist in your `jira-config.properties` file, add them to the file.
4. Update the values of these properties as desired.
5. Save your changes to the `jira-config.properties` file.
6. Restart your JIRA server.

Changing the Default Order for Comments from Ascending to Descending

To change the default order from Ascending to Descending so that the latest comments are shown first, follow these steps:

1. Access JIRA's 'Advanced Settings' page. (See Configuring Advanced Settings for more information.)
2. Edit the value of the `jira.issue.actions.order` property by clicking the existing value and changing it from `asc` to `desc`
3. Click the '**Update**' button to save the new value in the JIRA database.

> *The information on this page does not apply to **JIRA OnDemand**.*

Limiting the number of issues returned from a search view such as an RSS feed

JIRA allows you to view search results in several different formats, including Word, Excel, RSS or XML.

A search view that returns too many issues can take a long time for JIRA to complete and can use a large amount of memory. It can be a factor in OutOfMemoryErrors in JIRA.

An large RSS feed of search results can be particularly problematic, because:

- the user's RSS reader will continue to make the request periodically (for example, every hour)
- since the RSS reader makes the request, not the user directly, the user is unaware that the request takes a long time or is failing

You can use the following three properties in `jira-config.properties` to limit the number of issues returned by a search view.

See Making changes to `jira-config.properties` for the details of how to make and apply changes to your `jira-config.properties` file.

> ⚠ *The information on this page does not apply to **JIRA OnDemand**.*

`jira.search.views.default.max`

The `jira.search.views.default.max` property sets a 'soft' limit on the number of issues returned. It has a default value of 1000. You can set it to 100 (for example), by specifying the following in your `jira-config.properties` file:

```
jira.search.views.default.max = 100
```

For an RSS or XML view, JIRA applies the limit by appending the `tempMax` parameter to the URL of the search view. For example:

- http://jira.atlassian.com/sr/jira.issueviews:searchrequest-xml/temp/SearchRequest.xml?&type=2&pid=10240&resolution=-1&sorter/fie

In the above example, JIRA will limit the number of issues returned to 200 (in this example).

However users can override this 'soft' default by removing the `tempMax` parameter from the URL or by increasing the value of `tempMax`.

`jira.search.views.max.limit`

The `jira.search.views.max.limit` property sets a 'hard' limit on the number of issues returned. It has a default value of 1000. You can set this property's value to 200 (for example), by specifying the following in your `jira-config.properties` file:

```
jira.search.views.max.limit = 200
```

If a user makes an issue view request that would return more than 200 issues (in this example), JIRA does not return the issues but instead returns a 403 (Forbidden) error. While the user might not be happy, it prevents JIRA from consuming lots of resources and possibly running out of memory.

Make sure you set the value of `jira.search.views.max.limit` to greater than or equal to the 'soft' limit set by `jira.search.views.default.max`. Otherwise all search views that would return issues limited by the default 'soft' limit will instead return a 403 (Forbidden) error.

`jira.search.views.max.unlimited.group`

You may have a requirement for most users to have the limit imposed on them, but a few users to be exempt from the limit. One example of this is if your JIRA instance is Internet facing. You may want external (Internet) users to have the limit imposed on them, but for internal users to be able to produce unlimited search views. You can use the `jira.search.views.max.unlimited.group` property to achieve this.

The `jira.search.views.max.unlimited.group` property is disabled by default, by being either absent from your `jira-config.properties` file or present but disabled with a preceding '#'. If you enable this property in your `jira-config.properties` file, you must specify a valid group for its value or leave it empty. For example:

```
jira.search.views.max.unlimited.group = jira-administrators
```

> ℹ Users exempted from the limit via this technique will still have to add the `tempMax` parameter to the URL for an RSS or XML view, as described above, in order to exceed the `jira.search.views.default.max` soft limit.

Configuring File Attachments

When file attachments are enabled, your users will be allowed to attach files and screenshots to JIRA issues. This requires space on the server for storing the attachments.

> ⚠ Attachments are not stored in JIRA's database and so will need to be backed up separately.

File attachments are enabled by default. If you wish, you can configure the way JIRA handles attachments, or disable this feature altogether.

Note:

- your users must also have the **Create Attachments** permissions to attach files to issues
- to allow users to attach a file *when creating a new issue*, you need to ensure that the **Attachment** field is *not hidden* within the field configuration(s) associated with the specific issue type(s).

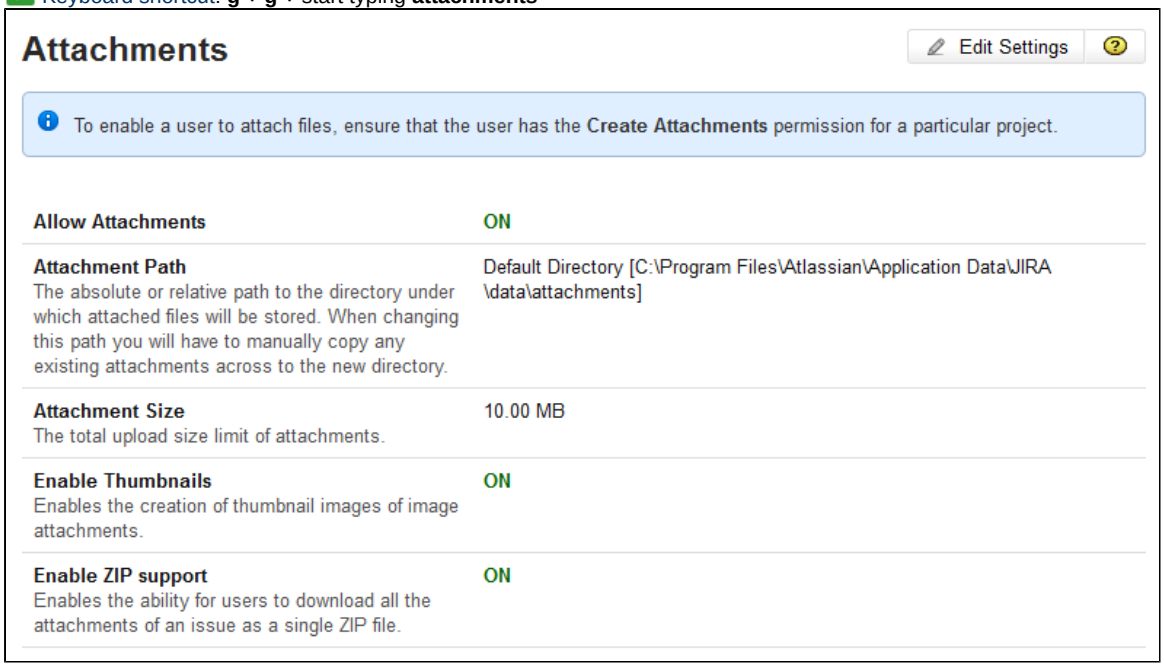
On this page:

- Configuring Attachment Settings
- Advanced Configurations
  - Configuring Thumbnail Size
  - Configuring ZIP-Format File Accessibility
  - Configuring ZIP-Format File Content Numbers on Issues

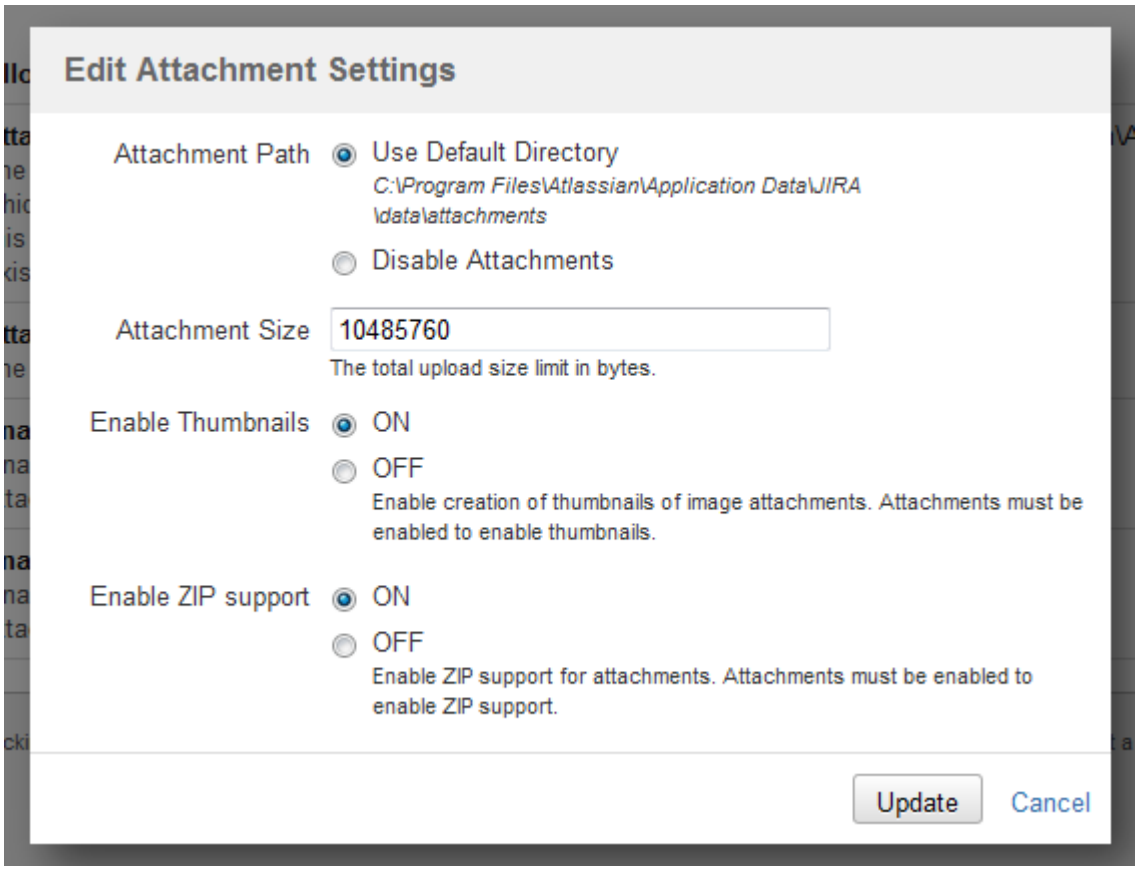
*Some functionality described on this page is restricted in **JIRA OnDemand**.*

Configuring Attachment Settings

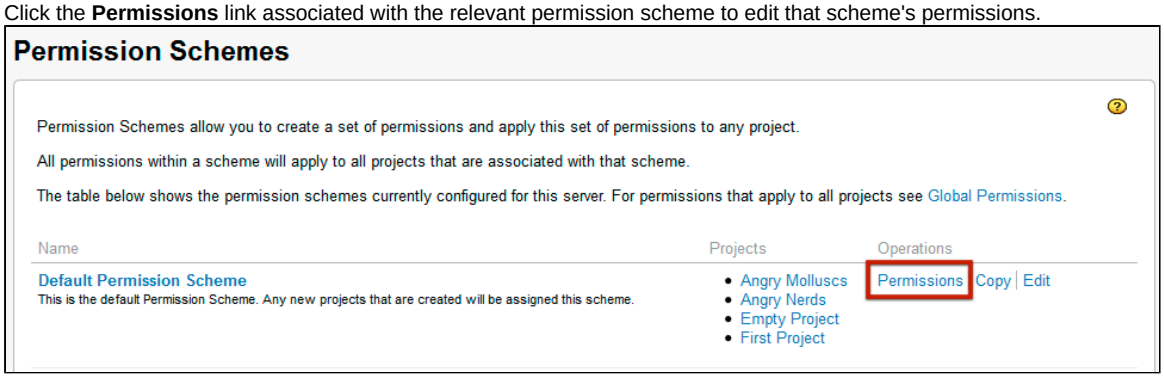
1. Log in as a user with the **JIRA System Administrators** or **JIRA Administrators** global permission.
   If you do not have the **JIRA System Administrators** global permission, you will not be able to enable or disable file attachments.
2. Select **Administration > System > Advanced > Attachments** (tab) to open the 'Attachment' page, which states whether attachments are on or off.
   Keyboard shortcut: **g + g +** start typing **attachments**



3. Click the **Edit Settings** button, which opens the **Edit Attachment Settings** dialog box:

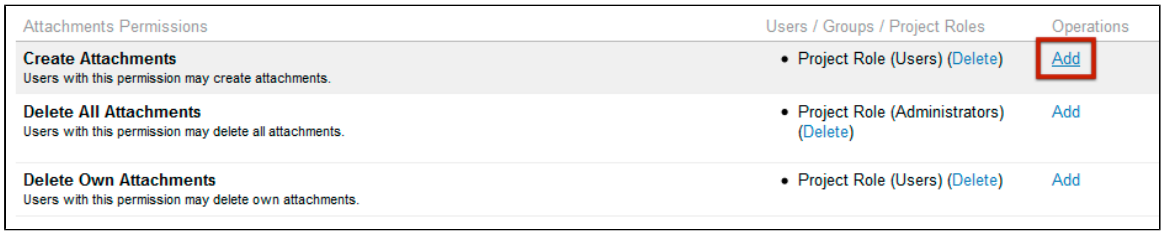


4. In the **Attachment Path** field, choose the **Use Default Directory** option. If you see more attachment path options than what is shown in the screenshot above, please refer to the note below.
   As mentioned above, if you have not logged in as a user with the **JIRA System Administrators** global permission, then this option will not be available to you.
5. In the **Attachment Size** field, specify the maximum attachment size. The default is 10485760 bytes (10 MB).
6. *(Optional)* In the **Enable Thumbnails** field, ensure that **ON** is selected if you wish to display image file attachments as thumbnails (or miniature previews) when viewing an issue. When this setting is enabled, JIRA automatically creates thumbnails of the following types of image attachments:
   - GIF
   - JPEG
   - PNG
   Please refer to the info note below for more information about thumbnails. If you use Linux, please refer to the Linux note below.
7. *(Optional)* In the **Enable ZIP Support** field, ensure that **ON** is selected if you wish to view the contents of zip files attached to an issue and allow all files attached to an issue to be downloaded as a single ZIP file.
8. Click the **Update** button to update JIRA's attachment settings.
   To attach files to issues, the appropriate users, groups or project roles must first be assigned the **Create Attachments** permission for the relevant project(s).
   To allow these users or group/project role members to delete their own attached files from issues, they must also be assigned the **Delete Own Attachments** permission for these projects too.
   There is no need to proceed any further if:
   - the permission schemes used by your project(s) already have the **Create Attachments** (and **Delete Own Attachments**) permission, or
   - your project(s) use JIRA's built-in **Default Permission Scheme**.
   However, if you wish to configure these permissions, proceed with the remaining steps below.
9. Select **Administration > Issues > Permission Schemes** to open the **Permission Schemes** page, which displays a list of all permission schemes in your JIRA system and the projects which use each scheme.
   Keyboard shortcut: **g + g +** start typing **permission schemes**
10. For each relevant permission scheme:
    a. Click the **Permissions** link associated with the relevant permission scheme to edit that scheme's permissions.

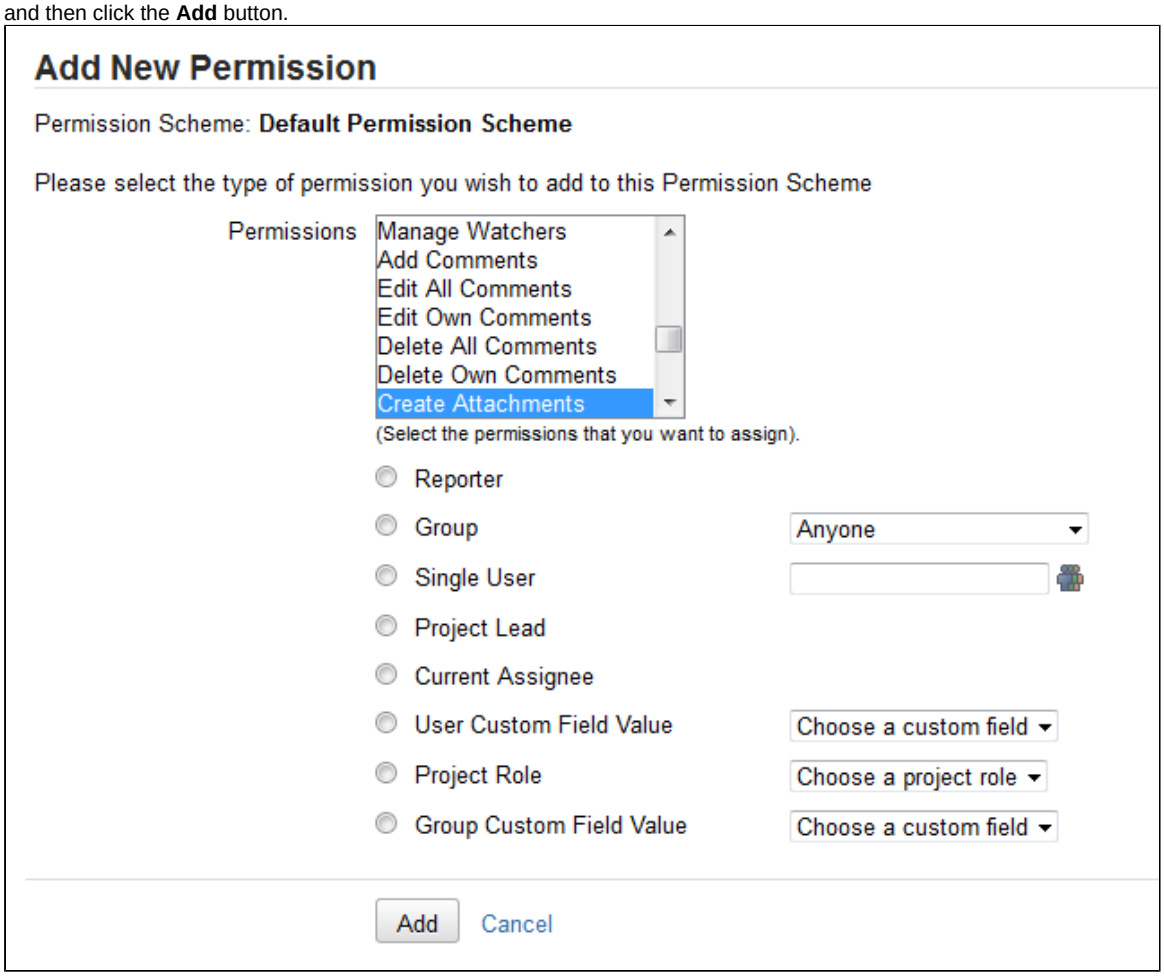


    b. On the **Edit Permissions** page, locate **Create Attachments** within the **Attachment Permissions** section and click the **Add**

link.



c. In the user selection options on the right of the **Add New Permission** page, select the relevant (groups of) users or roles and then click the **Add** button.



To allow these users or group/project role members to delete their own attachments, do not forget to assign them the **Delete Own Attachments** permission too.

Choosing a custom Attachment Path:

- If you upgraded JIRA with an XML backup from a JIRA version prior to 4.2 and used a custom directory for your attachment path, you can choose between using this custom directory (which cannot be edited) or the default directory for your attachment path location. However, once you switch to using the default directory, you can no longer choose the custom directory option.
- The default directory location is the `data/attachments` subdirectory of the JIRA Home Directory.

More information about thumbnails:

- You can configure the Issue Navigator column layout to display the thumbnails in an `Images` column.
- All thumbnail images are stored in JPEG format in the `attachments` directory, together with the original attachments. The thumbnail images are denoted by '`_thumb_`' in their file names.

Thumbnail image generation on Linux:

- Your system must have X11 support. This web page details the minimum set of libraries needed to use JDK 1.4.2 under RedHat Linux 9.0.
- The following java system property must be set: `-Djava.awt.headless=true`

Advanced Configurations

You can implement the following advanced configurations to modify the way JIRA handles attachments. However, these are not accessible through JIRA's attachment settings (above). One of these advanced configurations can be modified as an 'Advanced Setting' in JIRA's administration area, although the remaining two are implemented by defining properties in your `jira-config.properties` file.

Configuring Thumbnail Size

By default, thumbnails are 200 pixels wide and 200 pixels high. To change the dimensions of thumbnail images:

1. Stop JIRA.
2. Edit the `jira-config.properties` file in your JIRA Home Directory.
   See Making changes to the `jira-config.properties` file for more information.
3. Edit the values of the following properties:
   - `jira.thumbnail.maxwidth` — thumbnail width in pixels
   - `jira.thumbnail.maxheight` — thumbnail height in pixels

   If neither of these properties exist in your `jira-config.properties`file, add them to the file. For example, specify the following for a thumbnails that are 100 pixels wide:

   ```
   jira.thumbnail.maxwidth = 100
   ```

4. Delete all existing thumbnail images within the `attachments` directory (that is, those containing '`_thumb_`' in the filename).
5. Restart JIRA.

After restarting JIRA, all thumbnails will be recreated automatically using the new dimensions.

Configuring ZIP-Format File Accessibility

By default, JIRA allows you to access common ZIP-format files, with file extensions like '`.zip`' and '`.jar`' (Java archive files). However, there are numerous other ZIP-format files to which JIRA does not permit access by default. You can permit access to these files by doing the following:

1. Stop JIRA.
2. Edit the `jira-config.properties` file in your JIRA Home Directory.
   See Making changes to the `jira-config.properties` file for more information.
3. Remove the extensions from the `jira.attachment.do.not.expand.as.zip.extensions.list` property of the file types whose contents you wish to access in JIRA.
   If this property does not exist in your `jira-config.properties` file, add the name of this property, followed '=', followed by the content of the `<default-value/>` element copied from your JIRA installation's `jpm.xml` file. Then begin removing the exensions of file types whose contents you wish to access in JIRA.
4. Restart JIRA.

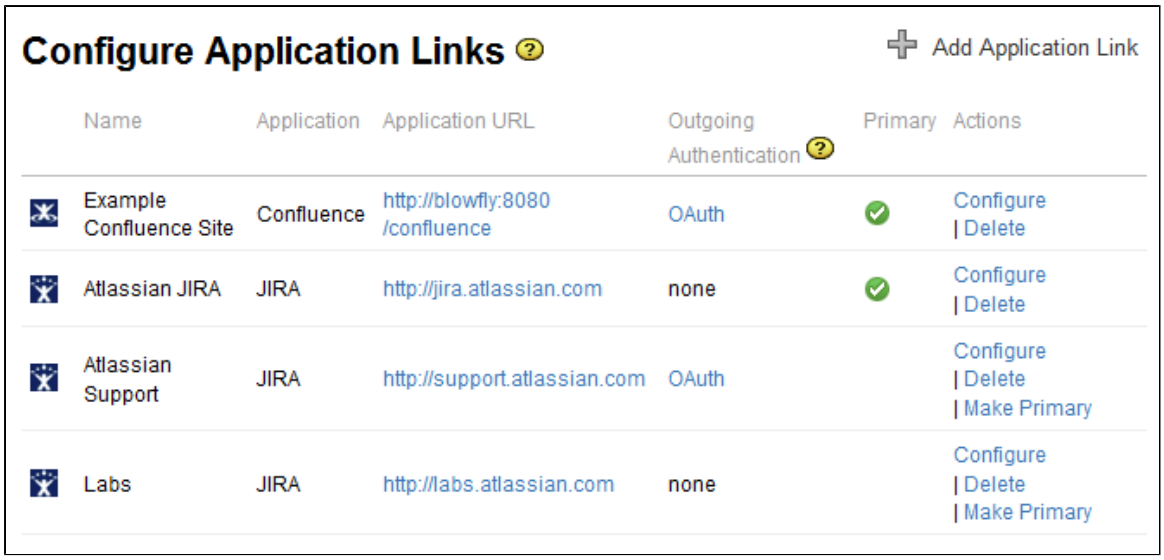
Configuring ZIP-Format File Content Numbers on Issues

By default, JIRA shows a maximum of 30 files in the content of ZIP-format files attached to an issue. To change this maximum value:

1. Access JIRA's 'Advanced Settings' page. (See Configuring Advanced Settings for more information.)
2. Edit the value of the `jira.attachment.number.of.zip.entries` property by clicking the existing value and specifying the maximum number of attachments you want to show on an issue.
3. Click the **Update** button to save the new value in the JIRA database.

Configuring Application Links

An application link is a trust relationship between two applications. Linking two applications allows you to share information and access one application's functions from within the other.



Screenshot above: Application links on a JIRA server

> *The information on this page does not apply to **JIRA OnDemand**.*

Related Topics

For information on integrating JIRA with other applications, see the following topics:

- Adding an Application Link
- Configuring Authentication for an Application Link
- Editing an Application Link
- Making an Application Link the Primary Link
- Relocating an Application Link
- Upgrading an Application Link
- Deleting an Application Link
- Configuring Project Links across Applications

Adding an Application Link

This page describes how to add a new application link in JIRA. The process for adding an application link is different depending on whether or not the application you are linking JIRA to supports Atlassian's Application Links.

If you are linking JIRA to an application without Application Links, you will need to perform additional configuration steps in that application, since the Application Links function in JIRA is unable to automatically configure authentication in applications that do not support Application Links.

Please read the appropriate set of instructions below:

- Linking to an application that supports Application Links.
- Linking to an application that does not support Application Links.

On this page:

- Adding an Application Link to an Application That Supports Application Links
- Adding an Application Link to an Application That Does Not Support Application Links
- Notes

> *The information on this page does not apply to **JIRA OnDemand**.*

Adding an Application Link to an Application That Supports Application Links

Before you begin:

- Make sure that the base URL is set correctly in JIRA. See Configuring JIRA Options for instructions.
- Make sure that the base URL is set correctly in the application which you intend to link to. See the appropriate instructions: Confluence instructions | FishEye/Crucible instructions | Bamboo instructions). This is required for synchronisation to work correctly.

To link to an application that supports Application Links:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click '**Add Application Link**'. Step 1 of the link wizard will appear.
3. Enter the **server URL** of the application that you want to link to (the 'remote application').
4. Click the '**Next**' button. Step 2 of the link wizard will appear.
5. Enter the following information:
    - '**Also create a link from 'XYZ' back to this server**' – Select this option if you want to create a two-way link between the remote application (which in this case is called 'XYZ') and your application. If you want to do this, you will need to enter the username and password of an administrator for the remote application.

      Please Note:
        - These credentials are not saved. They are only used at this step of the wizard to authenticate with the remote application, so that a reciprocal Application Link can be created in the remote application back to your application.
        - If the the remote application is JIRA or Confluence, these credentials need to be a user account with the system administrator global permission.
    - '**Reciprocal Link URL**' – The URL you give here will override the base URL specified in your remote application's administration console, for the purposes of the application links connection. Application Links will use this URL to access the remote application.
6. Click the '**Next**' button. Step 3 of the link wizard will appear.
7. Enter the information required to configure authentication for your application link:
    - '**The servers have the same set of users *and* usernames**' or '**The servers have either different sets of users or usernames**' – Select one of these options depending on how you manage users between the two applications.
    - '**These servers fully trust each other**' – Select this option if you fully understand and trust the behaviour of both applications at all times and are sure that each application will maintain the security of their private key.
      For more information about configuring authentication, see Configuring Authentication for an Application Link.
8. Click the '**Create**' button to create the application link.

Step 1 Step 2 Step 3

Screenshots above: Adding an application link to an application that supports Application Links (click to view full-sized images)

Adding an Application Link to an Application That Does Not Support Application Links

Before you begin:

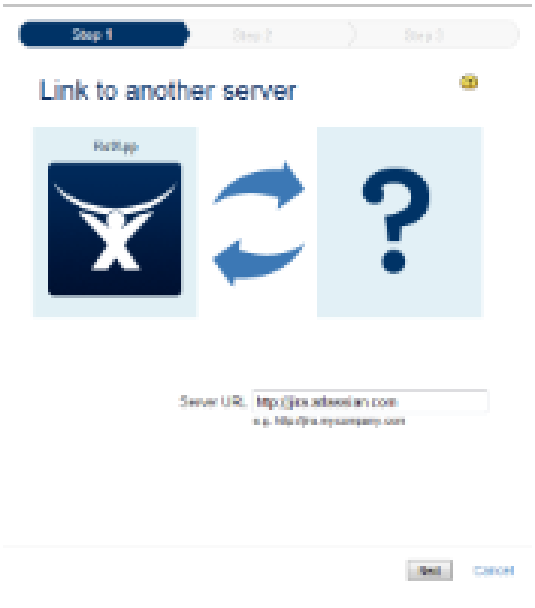
- Make sure that the base URL is set correctly in JIRA. See Configuring JIRA Options for instructions.
- Make sure that the base URL is set correctly in the application which you intend to link to. See the appropriate instructions: Confluence instructions | FishEye/Crucible instructions | Bamboo instructions). This is required for synchronisation to work correctly.

To link to an application that does not support Application Links:

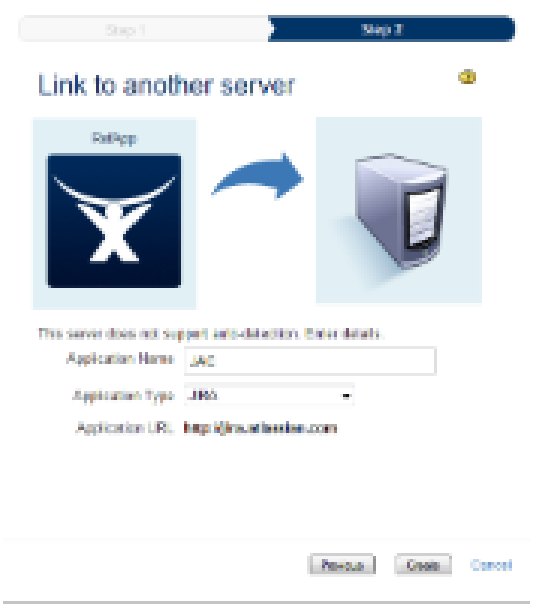
1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click '**Add Application Link**'. Step 1 of the 'Link to another server' dialogue will be displayed.
3. Enter the server URL of the application that you want to link to, in the '**Server URL**' field. Click the '**Next**' button. Step 2 of the 'Link to another server' dialogue will be displayed.
4. Fill out the fields, as follows:
    - '**Application Name**' — Enter the name by which this remote application will be referred to, in your application.

- '**Application Type**' — Select the type of application that you are linking to: Bamboo, FishEye/Crucible, JIRA, Confluence, Subversion.
- '**Application URL**' — This will be set to the server URL you entered in the previous step and will not be editable.

5. Click the '**Create**' button to create the application link. The 'Configure Application Links' page will be displayed, listing all of the application links that have currently been set up for your application including the one you just added.
6. Configure the desired authentication type (Trusted Applications, OAuth, basic HTTP, none) for your new application link. See Configuring Authentication for an Application Link.
7. In your application that does not support Application Links, configure the same type of authentication that you configured for your application link's *outgoing* authentication (in the previous step). For example, if you configured outgoing Trusted Applications authentication in your Application-Links-enabled application, you also need log into your non-Application-Links application and manually configure Trusted Applications (see the relevant administrator's documentation for the application).



Step 1



Step 2

Screenshots above: Adding an application link to an application that supports Application Links (click to view full-sized images)

Notes

Related Topics

- Making an Application Link the Primary Link
- Configuring Authentication for an Application Link
- Configuring Project Links across Applications

Configuring Authentication for an Application Link

Configuring authentication for an application link is essentially defining the level of trust between JIRA and another application.

On this page:

- Choosing Authentication for an Application Link
- Security Implications for each Authentication Type
- About Primary Authentication Types
- About Impersonating and Non-Impersonating Authentication Types

*The information on this page does not apply to **JIRA OnDemand**.*

Choosing Authentication for an Application Link

The level of authentication that you should configure for your application link depends on a number of factors.

- **Do the two applications you are linking trust each other?** i.e. are you sure that the code in the application will behave itself at all times and that the application will maintain the security of its private key?
- **Do the two applications you are linking share the same set of users *and* user names?**
- **Do you have administrative access to the application you are linking to JIRA?**

Common scenarios include:

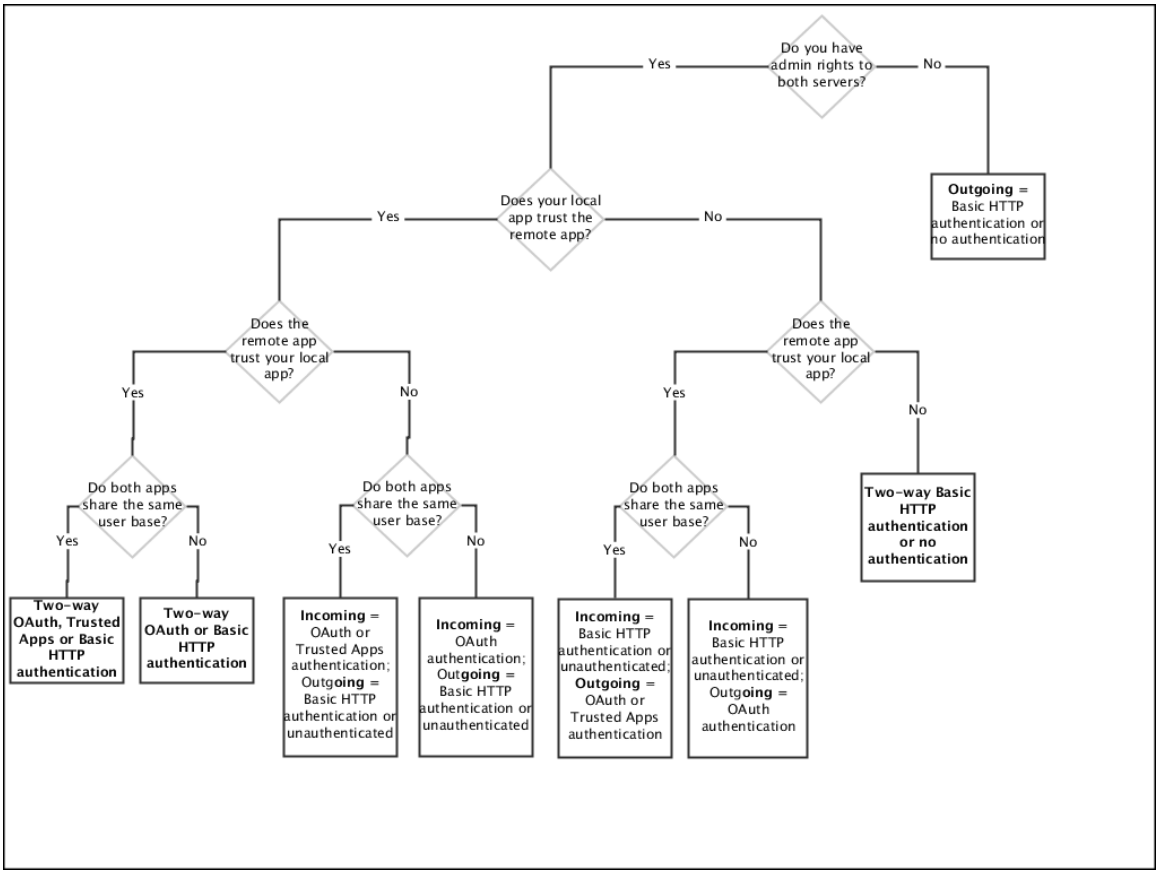
- If the two applications you are linking **trust each other** and **share the same set of users *and* user names**, configure **two-way authentication using Trusted Applications** for both incoming and outgoing authentication. For example, you may link your internal JIRA server to an internal FishEye server.

- If the two applications you are linking **trust each other** but **do not share the same set of users *or* user names**, configure **two-way authentication using OAuth** for both incoming and outgoing authentication. For example, you may link your internal JIRA server to an external (customer-facing) Confluence server.
- If you **do not have administrative rights to the application that you are linking to** (e.g. linking to a public FishEye server), configure a **one-way outgoing link** authenticated using **basic HTTP authentication** or **do not configure any authentication** for the link. For example, you may link your external Confluence server to a partner organisation's Confluence server. An unauthenticated link will still allow the local application to render hyperlinks to the remote application or query anonymously-accessible APIs.

The flowchart below provides a guide to what authentication you should configure for your application link.

Read the following topics for information on how to configure authentication for an application link:

- Configuring Basic HTTP Authentication for an Application Link
- Configuring OAuth Authentication for an Application Link
- Configuring Trusted Applications Authentication for an Application Link
- Incoming and Outgoing Authentication



Flowchart above: Determining what authentication to configure for an Application Link

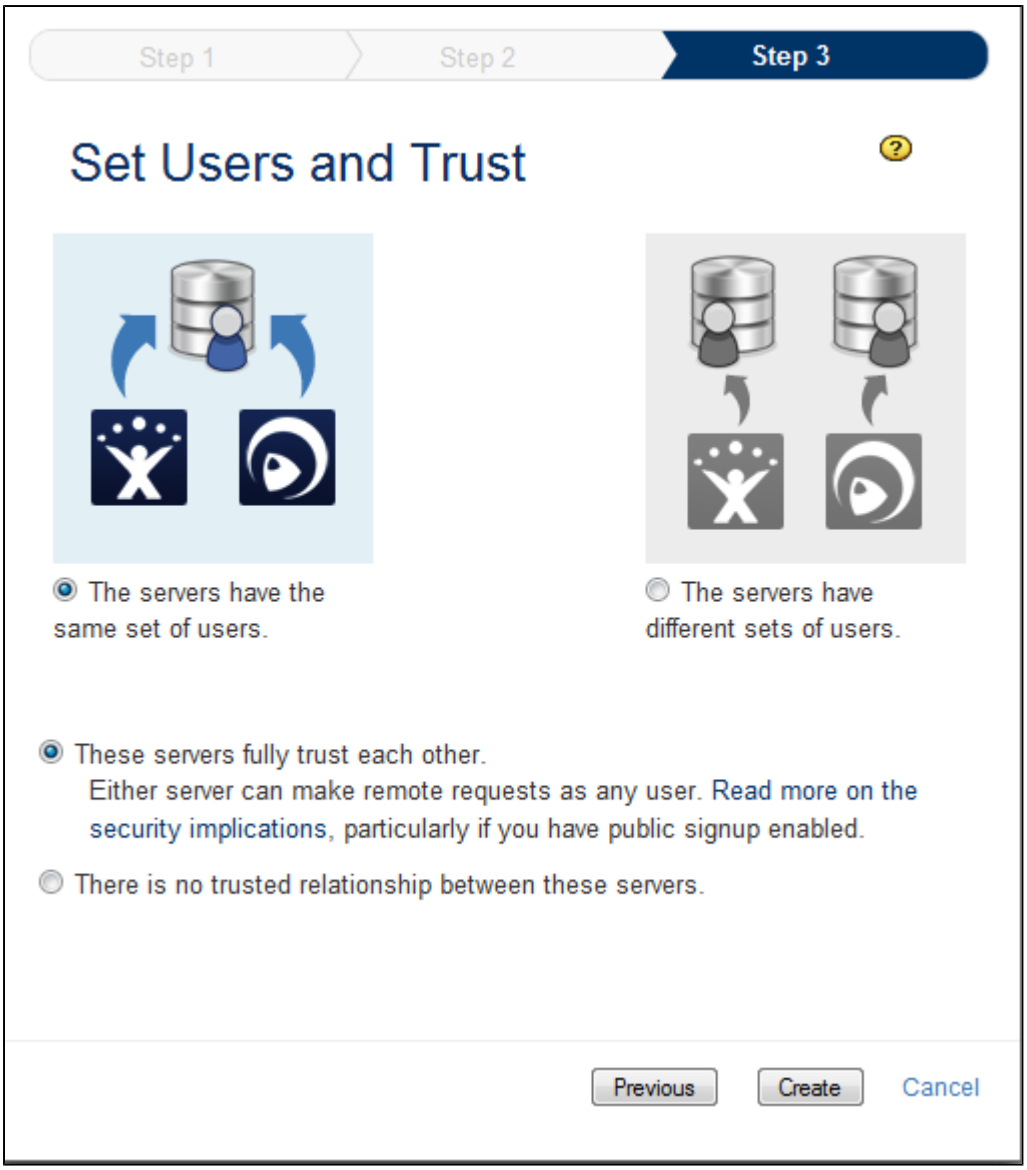
Security Implications for each Authentication Type

If you configure **Trusted Applications authentication** (i.e. both applications fully trust each other and have the same set of users *and* user names), please be aware of the following security implications:

- Trusted applications are a **potential security risk**. When you configure Trusted Applications authentication, you are allowing one application to access another as any user. This allows all of the built-in security measures to be bypassed. Do not configure a trusted application unless you know that all code in the application you are trusting will behave itself at all times, and you are sure that the application will maintain the security of its private key.

If you configure **OAuth authentication** (i.e. both applications fully trust each other but have different sets of users or user names), please be aware of the following security implications:

- Adding an OAuth consumer requires the transmission of sensitive data. To prevent 'man-in-the-middle' attacks, it is recommended that you **use SSL** for your applications while configuring OAuth authentication.
- Do not link to an application using OAuth authentication, unless you **trust all code in the application** to behave itself at all times. OAuth consumers are a potential security risk to the applications that they are linked to.



Screenshot above: Configuring authentication during application link setup

About Primary Authentication Types

You can configure multiple authentication types for each application link. When a feature makes a request using an Application Link, it will use one of the configured authentication types. If more than one authentication type is configured, it will by default use the authentication type that is marked as the primary authentication type. The default authentication type is indicated by the green tick next to the authentication type on the list application link screen.

You **cannot** configure which authentication type is the primary authentication type. The primary authentication type is determined automatically by Application Links and depends on a weight defined by each authentication type method. However, every feature that uses Application Links can also choose to use a specific authentication type and might not use the default primary authentication type.

About Impersonating and Non-Impersonating Authentication Types

Applications Links allows you to configure 'impersonating' and 'non-impersonating' authentication types:

- **Impersonating authentication types** make requests on behalf of the user who is currently logged in. People will see only the information that they have permission to see. This includes OAuth and Trusted Applications authentication.
- **Non-impersonating authentication types** always use a pre-configured user when making a request. Everyone logged into the system will see the same information. This includes basic HTTP authentication.

Configuring Basic HTTP Authentication for an Application Link

The instructions on this page describe how to configure **Basic HTTP authentication** for outgoing authentication and/or incoming authentication of an application link.

Basic HTTP authentication allows JIRA to provide user credentials to a remote application and vice versa. Once authenticated, one application can access specified functions on the other application on behalf of that user. For example, if you supply the credentials of a JIRA

administrator on your JIRA server to a remote application, the remote application will be able to access all functions on your JIRA server that the JIRA administrator can access.

This method of authentication relies on the connection between JIRA and the remote application being secure. We recommend that you use Trusted Applications authentication or OAuth authentication for your application link instead, if possible.

On this page:

- Before You Begin
- Configuring Basic HTTP Authentication for Outgoing Authentication
- Configuring Basic HTTP Authentication for Incoming Authentication
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Before You Begin

- The instructions assume that **both of the applications that you are linking have the Application Links plugin installed**. If the remote application that you are linking to supports Basic HTTP authentication, but does not have the Application Links plugin installed, you will need to configure Basic HTTP authentication from within the remote application (see the relevant administrator's documentation for the application). This is in addition to configuring the outgoing/incoming authentication for the application link (as described below).
- You must be a JIRA system administrator to configure Basic HTTP authentication for an application link.

Configuring Basic HTTP Authentication for Outgoing Authentication

Configuring **outgoing basic http authentication** will allow JIRA to trust a remote application (i.e. allow the remote application to access specified functions in JIRA).

To configure basic HTTP authentication for an outgoing application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to configure authentication for.
3. Click the '**Outgoing Authentication**' tab. The outgoing authentication page will be displayed.
4. Click the '**Basic Access**' tab.
5. Click the '**Configure**' button and enter the credentials (username and password) that the remote application will use to log into your application .
6. Click the '**Apply**' button to save your changes.

Configuring Basic HTTP Authentication for Incoming Authentication

Configuring **incoming basic http authentication** will allow the remote application that you are linking to, to trust JIRA (i.e. allow JIRA to access specified functions on the remote application it is linked to).

To configure basic HTTP authentication for an incoming application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to configure authentication for.
3. Click the '**Incoming Authentication**' tab. The incoming authentication page will be displayed.
4. Click the '**Basic Access**' tab.
5. Click the '**Configure**' button and enter the credentials (username and password) that the your application will use to log in to the remote application.
6. Click the '**Apply**' button to save your changes.

Notes

Related Topics

Configuring OAuth Authentication for an Application Link
Configuring Trusted Applications Authentication for an Application Link

Configuring OAuth Authentication for an Application Link

The instructions on this page describe how to configure **OAuth** for outgoing authentication and/or incoming authentication of an application link.

OAuth is a protocol that allows a web application to share data/resources with any other OAuth-compliant external application. These external applications could be another web application (such as a Confluence installation or an iGoogle home page), a desktop application o a mobile device application, provided that they are accessible from within your network or available on the Internet.

For example, you could set up an application link between JIRA and an iGoogle page using OAuth authentication. This would allow you to view data from your JIRA server in a JIRA gadget on the iGoogle page.

If you were setting up an application link between two trusted applications which do not share the same set of users (and both applications have the Application Links plugin installed), you would typically configure OAuth for both outgoing authentication and incoming authentication. See Configuring Authentication for an Application Link for other configurations.

Key OAuth Terminology

- **Service provider** — An application that shares ('provides') its resources.
- **Consumer** — An application that accesses ('consumes') a service provider's resources.
- **User** — An individual who has an account with the Service Provider.

For more information about OAuth, see Configuring OAuth as well as the OAuth specification.

On this page:

- Before You Begin
- Configuring OAuth for Outgoing Authentication
- Configuring OAuth for Incoming Authentication
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Before You Begin

- Adding an OAuth consumer requires the transmission of sensitive data. To prevent 'man-in-the-middle' attacks, it is recommended that you **use SSL** for your applications while configuring OAuth authentication.
- Do not link to an application using OAuth authentication, unless you **trust all code in the application** to behave itself at all times. OAuth consumers are a potential security risk to the applications that they are linked to.
- The instructions assume that **both of the applications that you are linking have the Application Links plugin installed**. If the remote application that you are linking to supports OAuth, but does not have the Application Links plugin installed, you will need to configure OAuth from within the remote application (see the relevant administrator's documentation for that application) in addition to configuring the outgoing/incoming authentication for the application link (as described below).
- You must be a JIRA system administrator to configure OAuth authentication for an application link.

Configuring OAuth for Outgoing Authentication

Configuring **outgoing OAuth authentication** will allow JIRA to access specific functions and data on a remote application, on behalf of any registered user of that remote application.

To configure OAuth authentication for an outgoing application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to configure OAuth for.
3. Click the '**Outgoing Authentication**' tab. The outgoing authentication page will be displayed.
4. Click the '**OAuth**' tab.
5. If you are not currently logged in to the remote application (or you logged in to the remote application under a variant of the application's hostname, such as the IP address), a login dialogue will display.
    - Enter the '**Username**' and '**Password**' for the remote server, not your local server, and click the '**Login**' button. The remote server needs to learn the identity of your local server for the OAuth protocol to work and your admin credentials are used to store your local server's public key on the remote server. If you are already logged into your remote server, then the appropriate changes can be made without having to log in again.
6. Click the '**Enable**' button to enable OAuth authentication for the outgoing link. Your application will be automatically set up to be the 'consumer' and the remote application as a 'service provider'.

Configuring OAuth for Incoming Authentication

Configuring **incoming OAuth authentication** will allow the remote application that you are linking to, to access specific functions and data in JIRA on behalf of any JIRA user.

To configure OAuth authentication for an incoming application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to configure OAuth for.
3. Click the '**Incoming Authentication**' tab. The incoming authentication page will be displayed.
4. Click the '**OAuth**' tab.
5. Click the '**Enable**' button to enable OAuth authentication for the incoming link. The remote application will be automatically set up to be the 'consumer' and your local application as a 'service provider'.

Notes

Related Topics

Configuring Basic HTTP Authentication for an Application Link
Configuring Trusted Applications Authentication for an Application Link

Configuring Trusted Applications Authentication for an Application Link

The instructions on this page describe how to configure **Trusted Applications** for outgoing authentication and/or incoming authentication of an application link.

Trusted Applications authentication allows one Atlassian application access to specified functions and data in another Atlassian application on behalf of any user. The user only needs to log in to one application, without needing to log in to the other. For this authentication to succeed, however, the user must have an account on both applications with the same user name.

For example, if Trusted Applications authentication were configured between a JIRA server and a Confluence server *and* every user had the same user name on both servers, any of these users (logged in only to Confluence) will see same list of issues in a Confluence 'JIRA Issues' macro as they would through the JIRA Issue Navigator when logged in to JIRA independently. This includes issues restricted from public view, which these users have permission to view.

If you were setting up an application link between two trusted applications which have the same set of users *and* user names (and both applications have the application links plugin installed), you would typically configure Trusted Applications for both outgoing authentication and incoming authentication. See Configuring Authentication for an Application Link for other configurations.

On this page:

- Before You Begin
- Configuring Trusted Applications for Outgoing Authentication
- Configuring Trusted Applications for Incoming Authentication
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Before You Begin

- Trusted applications are a **potential security risk**. When you configure Trusted Applications authentication, you are allowing one application to access another as any user. This allows all of the built-in security measures to be bypassed. Do not configure a trusted application unless you know that all code in the application you are trusting will behave itself at all times, and you are sure that the application will maintain the security of its private key.
- The instructions below assume that **both of the applications you are linking have the Application Links plugin installed**. If the remote application that you are linking to supports Trusted Applications, but does not have the Application Links plugin installed, you will need to configure Trusted Applications from within the remote application (see the relevant administrator's documentation for the application) in addition to configuring the outgoing/incoming authentication for the application link (as described below).
- You must be a JIRA system administrator to configure Trusted Applications authentication for an application link.

Configuring Trusted Applications for Outgoing Authentication

Configuring **outgoing Trusted Applications authentication** will allow JIRA to access functions and data on a remote application, on behalf of a user whose user names are the same on both applications.

To configure Trusted Applications authentication for an outgoing application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to configure Trusted Applications authentication for.
3. Click the '**Outgoing Authentication**' tab. The outgoing authentication page will show, with the '**Trusted Applications**' tab displayed.
4. If you are not currently logged into the remote application (or you logged into the remote application under a variant of the application's hostname, e.g. the IP address), a login dialogue will display.
    - Enter the '**Username**' and '**Password**' for the remote server, (not your local server), and click the '**Login**' button. You need to enter the credentials for the remote server, as the remote server needs to be instructed to trust your local server for the Trusted Applications protocol to work. If you are already logged into your remote server, then the appropriate changes can be made without having to log in again.
5. Configure the settings for the Trusted Applications authentication:
    - '**IP Patterns**' — Enter the IP addresses (IPv4 only) from which the remote application will accept requests (this effectively is

the IP address your local server). You can specify wildcard matches by using an asterisk (*), e.g. '`192.111.*.*`' (note, you cannot use netmasks to specify network ranges). If you are entering multiple IP addresses, separate them with commas or spaces.

⚠ *Please note, if you are setting up Trusted Applications between two applications that both have the Application Links plugin installed, you can leave this field blank (or explicitly use `*.*.*.*`). However, if your remote application does not have the Application Links plugin installed and you are configuring the IP Patterns in the remote application (not the Application Links plugin),* ***you must not leave this field blank nor use `*.*.*.*`****. Failure to configure IP address restrictions in this scenario is a security vulnerability, allowing an unknown site to log into your site under a user's login ID.* Consider the following scenarios, if you want to limit access by using this field:

- If your local application is using a proxy server, you need to add the proxy server's IP address to this field.
- If your local application is a clustered instance of Confluence, you need to configure the remote server to accept requests from each cluster node. If you do not set up each node appropriately, your Confluence users may not be able to view any information from the remote server. You can set this up by either specifying each individual IP address for each node of the cluster (e.g. `172.16.0.10, 172.16.0.11, 172.16.0.12`), or specifying the IP address for the clustered Confluence instance using wildcards (e.g. `172.16.0.*`).

- **'URL Patterns'** — Enter the URLs in the remote application that your local application will be allowed to access. Each URL corresponds to a particular application function. Enter one URL per line, as follows:
  - If your remote application is JIRA, enter the following URL Patterns: `/plugins/servlet/streams`, `/sr/jira.issueviews:searchrequest`, `/secure/RunPortlet`, `/rest`, `/rpc/soap`
  - If your remote application is Confluence, enter the following URL Patterns: `/plugins/servlet/streams`, `/plugins/servlet/applinks/whoami`
- '**Certificate Timeout (ms)**' — Enter the certificate timeout. The default is 10 seconds. The certificate timeout is used to prevent replay attacks. For example, if a Trusted Applications request is intercepted and (maliciously) re-sent, the application will be able to check when the request was first sent. If the second request is sent more than 10 seconds (or whatever the certificate timeout is set to) after the initial request, it will be rejected. Please note, you should not have to change the default value of this field for most application links. Note that the certificate timeout relies on the clocks on both servers being synchronised.

6. Click the '**Apply**' button to save your changes.

Configuring Trusted Applications for Incoming Authentication

Configuring **incoming Trusted Applications authentication** will allow the remote application that you are linking to, to access functions and data in JIRA, on behalf of a user whose user names are the same on both applications.

To configure Trusted Applications authentication for an incoming application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to configure Trusted Applications authentication for.
3. Click the '**Incoming Authentication**' tab. The imconing authentication page will show, with the '**Trusted Applications**' tab displayed.
4. The tab will show whether Trusted Applications is currently enabled or not. Use the '**Modify**' or '**Configure**' button to configure Trusted Applications. The Trusted Applications configuration settings will be displayed:
   - '**IP Patterns**' — Enter the IP addresses (IPv4 only) from which our application will accept requests. You can specify wildcard matches by using an asterisk (*), e.g. '`192.111.*.*`' (note, you cannot use netmasks to specify network ranges). If you are entering multiple IP addresses, separate them with commas or spaces.

     ⚠ *Please note, if you are setting up Trusted Applications between two applications that both have the Application Links plugin installed, you can leave this field blank (or explicitly use `*.*.*.*`). However, if your remote application does not have the Application Links plugin installed and you are configuring the IP Patterns in the remote application (not the Application Links plugin),* ***you must not leave this field blank nor use `*.*.*.*`****. Failure to configure IP address restrictions in this scenario is a security vulnerability, allowing an unknown site to log into your site under a user's login ID.* Consider the following scenarios, if you want to limit access by using this field:
     - If the remote application is using a proxy server, you need to add the proxy server's IP address to this field.
     - If the remote application is a clustered instance of Confluence, you need to accept requests from each cluster node. If you do not specify each node's address, Confluence users may not be able to view any data from your application. You can set this up by either specifying each individual IP address for each node of the cluster (e.g. 172.16.0.10, 172.16.0.11, 172.16.0.12), or specifying the IP address for your clustered Confluence instance using wildcards (e.g. 172.16.0.*).
   - **'URL Patterns'** — Enter the local URLs that the remote application will be allowed to access. Each URL corresponds to a particular application function. Enter one URL per line, as follows:
     - If your local application is JIRA, enter the following URL Patterns — `/plugins/servlet/streams`, `/sr/jira.issueviews:searchrequest`, `/secure/RunPortlet`, `/rest`, `/rpc/soap`
     - If your local application is Confluence, enter the following URL Patterns — `/plugins/servlet/streams`, `/plugins/servlet/applinks/whoami`
   - '**Certificate Timeout (ms)**' — Enter the certificate timeout. The default is 10 seconds. The certificate timeout is used to prevent replay attacks. For example, if a Trusted Applications request is intercepted and (maliciously) re-sent, the application will be able to check when the request was first sent. If the second request is sent more than 10 seconds (or whatever the certificate timeout is set to) after the initial request, it will be rejected. Please note, you should not have to change the default value of this field for most application links. Note that the certificate timeout relies on the clocks on both servers being synchronised.
5. Click the '**Apply**' button to save your changes.

Notes

Related Topics

Configuring Basic HTTP Authentication for an Application Link
Configuring OAuth Authentication for an Application Link

Incoming and Outgoing Authentication

When you configure authentication for an application link, you are defining the level of trust between the two linked servers. When configuring a link from one application to another, you can set up:

- **Incoming authentication** (authentication of requests coming from a linked application into this application).
- **Outgoing authentication** (authentication of requests sent from this application to a linked application).

See Configuring Authentication for an Application Link.

*The information on this page does not apply to **JIRA OnDemand**.*

Editing an Application Link

You can change the details of an existing application link, such as the application's name and its display URL.

On this page:

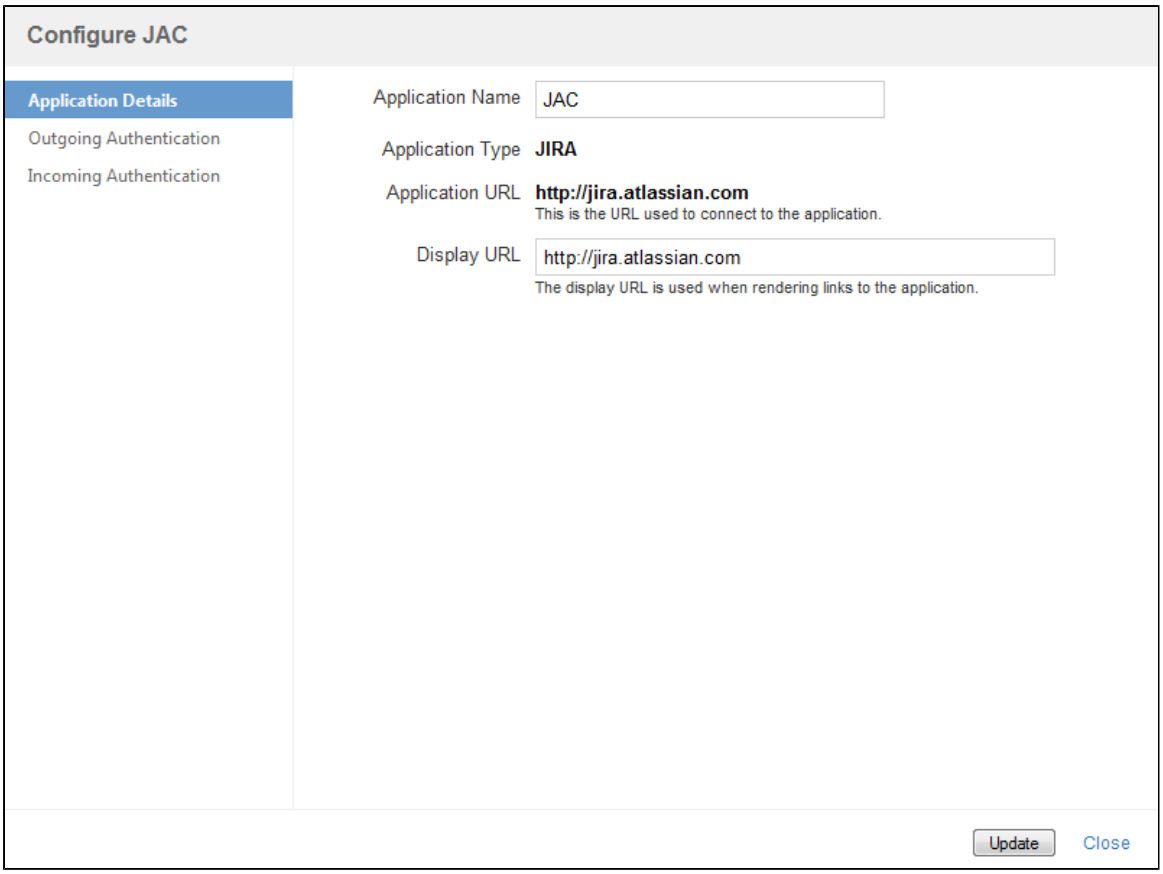
- Editing an Application Link
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Editing an Application Link

To edit an application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Configure**' link next to the application link that you want to edit the details for. The application details for the application link will be displayed.
3. Update the application details as desired. Please note, you cannot update the Application Type nor the Application URL.
    - '**Application Name**' — Update this field to change the display name for the application that you are linking to.
    - '**Display URL**' — This URL is used when displaying links to the application in the browser. When creating the application link, you may have used a URL that is not accessible to other users, such as an internal IP address. If so, you can change the display URL to an address in a domain that is accessible to other users.
4. Click the '**Update**' button to save your changes.



Screenshot above: Editing an application link

Notes

Related Topics

Configuring Authentication for an Application Link
Making an Application Link the Primary Link
Relocating an Application Link

Making an Application Link the Primary Link

If you have set up application links from JIRA to more than one of the same type of application servers, e.g. you have linked JIRA to two FishEye servers, then one of these servers will be marked as the 'Primary' link. This means that any outgoing requests will be directed to the primary link's server.

For example, if you have set up a JIRA server that is linked to two FishEye servers with two-way authentication for both links, you can nominate an application link to one of the FishEye servers as the primary link. Every time JIRA requests FishEye information (e.g. for a FishEye repository changeset), JIRA will request this information from the primary link's FishEye server. Note, both FishEye servers can still make requests of the JIRA server.

On this page:

- Making an Application Link the Primary Link
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Making an Application Link the Primary Link

To make an application link the primary link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Make Primary**' link next to the application link that you want to make the primary link. A '✅' symbol will display in the 'Primary' column next to the application link.
   The 'Primary' column and 'Make Primary' link will only display if you have set up application links to more than one of the same application type, e.g. you have linked your application to two JIRA servers.

Notes

Please read Making a Project Link the Primary Link for information on how primary project links also influence the information shared between servers.

Related Topics

Making a Project Link the Primary Link

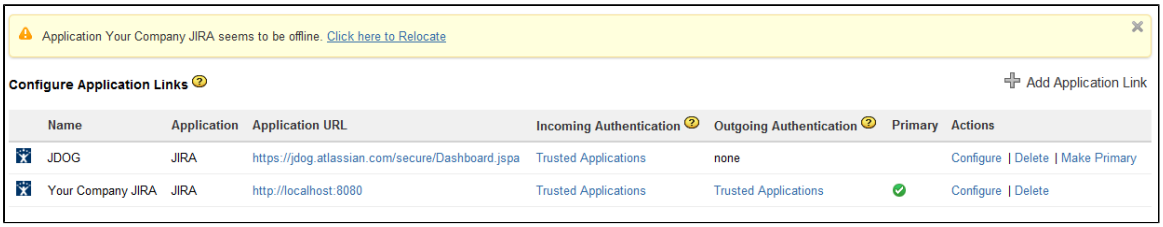
Relocating an Application Link

This page describes how to change the location of an application link. You will need to relocate an application link if the target application has been moved to a new address.

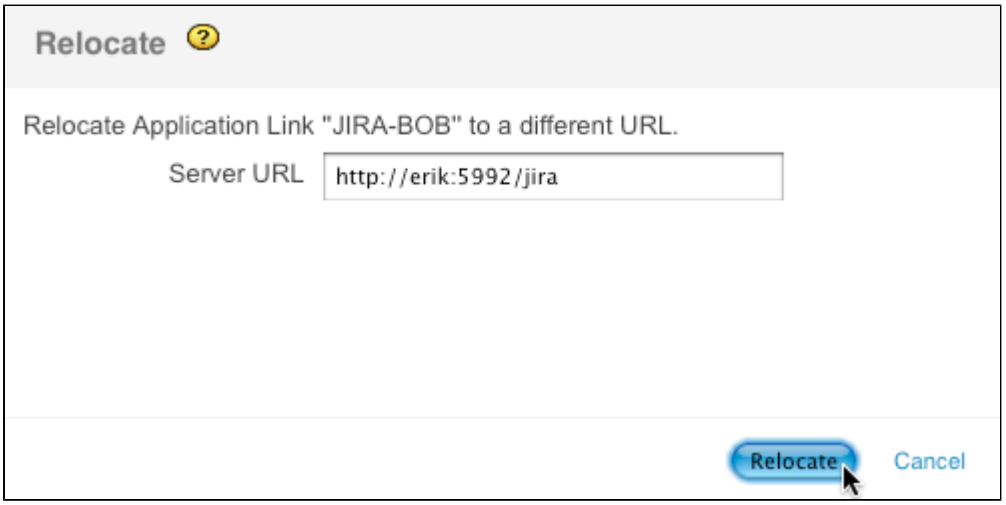
To relocate an application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. If the remote application for an application link cannot be reached by your application, the '**List Application Links**' page will display a warning message (see 'Relocate Link - Warning Message' screenshot below).
3. If your remote application has been moved to a different address (rather than just being offline temporarily), click the '**Relocate**' link in the warning message (see 'Relocate Link - Updating URL' screenshot below).
4. Enter the new URL for the remote application of your application link and click '**Relocate**'.
5. You will need to confirm the relocation, if the new URL cannot be contacted. Otherwise, the application link will be updated.

> *The information on this page does not apply to **JIRA OnDemand**.*



Screenshot above: Relocate link – The warning message



Screenshot above: Relocate link – Updating the URL

Related Topics

Making an Application Link the Primary Link

Upgrading an Application Link

The instructions on this page describe how to upgrade an existing application link. You may want to upgrade an application link in either of the following situations:

- **Your JIRA installation has been upgraded from a version which does not support/include Application Links to a version that does.**
  For example, you may have configured Trusted Applications or OAuth relationships in a JIRA 4.2 installation (which did not include

Application Links) and upgraded it to JIRA 4.3 or later (which includes Application Links).

- **Your remote application (to which JIRA has existing application links), has been upgraded from a version which does not support/include Application Links to a version that does.**
  For example, you had set up an application link in a JIRA 4.3 installation or later (which includes Application Links) to Confluence 3.4 installation (which did not include Application Links) and that Confluence installation was upgraded to Confluence 3.5 or later (which includes Application Links).

On this page:

- Upgrading an Application Link (Local App Upgraded to Include Application Links)
- Upgrading an Application Link (Remote App Upgraded to Include Application Links)
- Notes

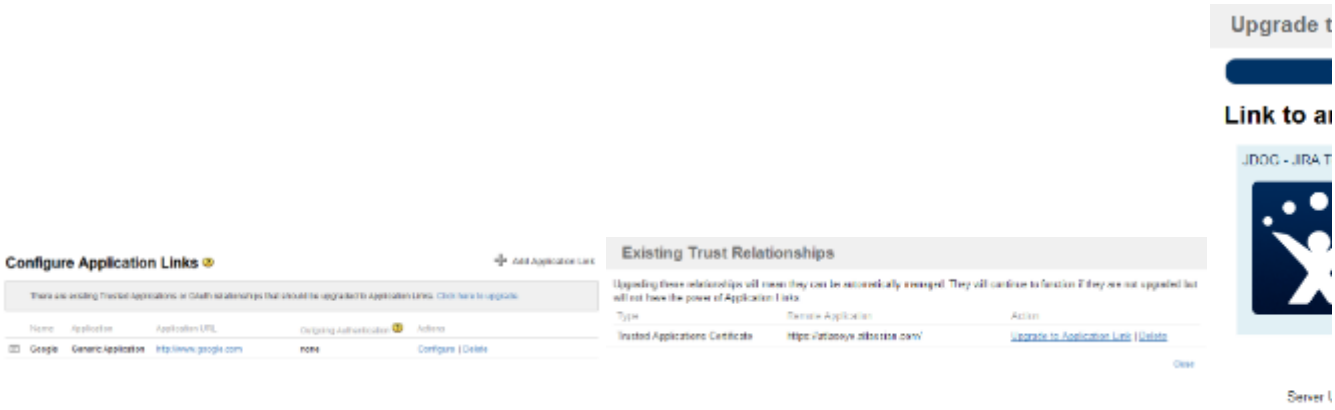
*The information on this page does not apply to **JIRA OnDemand**.*

Upgrading an Application Link (Local App Upgraded to Include Application Links)

When you upgrade a JIRA version that does not include Application Links to a version that does, you will have the option of converting any Trusted Applications or OAuth links to Application Links. Converting these links greatly simplifies the link configuration process.

To upgrade an application link when your local application has been upgraded to include Application Links:

1. After your application upgrade, navigate to the administration console.
2. Click '**Application Links**'. The 'Configure Application Links' screen will be displayed with the following message:
   *"There are existing Trusted Applications or OAuth relationships that should be upgraded to Application Links. **Click here to upgrade.**"*
3. Click the '**Click here to upgrade**' link. The 'Existing Trust Relationships' screen will be displayed showing all Trusted Applications and OAuth relationships that can be upgraded to Application Links.
4. Click the '**Upgrade to Application Link**' link next to the desired trust relationship. The 'Upgrade to Application Link' wizard will be displayed.
5. Complete the wizard. The process will be similar to adding a new link (described on Adding an Application Link), except that most fields should be pre-filled.



Step 1 Step 2 Step 3

Screenshots above: Upgrading an application link for local application

Upgrading an Application Link (Remote App Upgraded to Include Application Links)

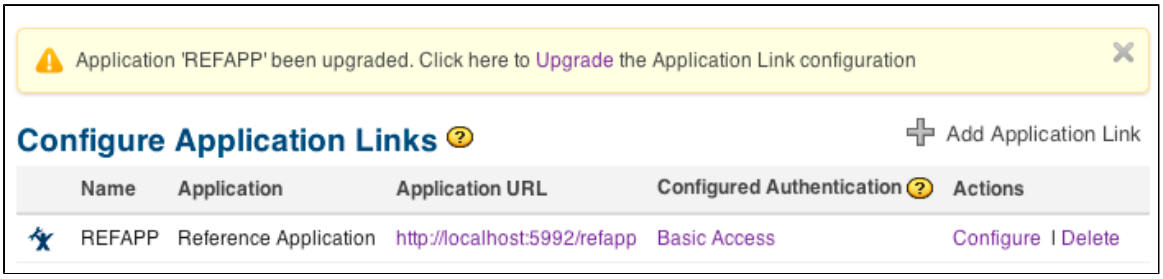
When an application link is created between a version of JIRA that supports Application Links, and a remote legacy application (either a non-Atlassian product, or an older version of an Atlassian product that did not ship with Application Links), this link is configured to run in "legacy mode". While there is no distinguishable difference to a user, the connection and configuration without Application Links is a little different. For example:

- Setting up OAuth requires manual configuration by the administrator. In OAuth authentication where both applications support Application Links, exchange of the consumer keys and public keys is done automatically.
- The Trusted Applications protocol (Atlassian-specific) will not be available for authentication.

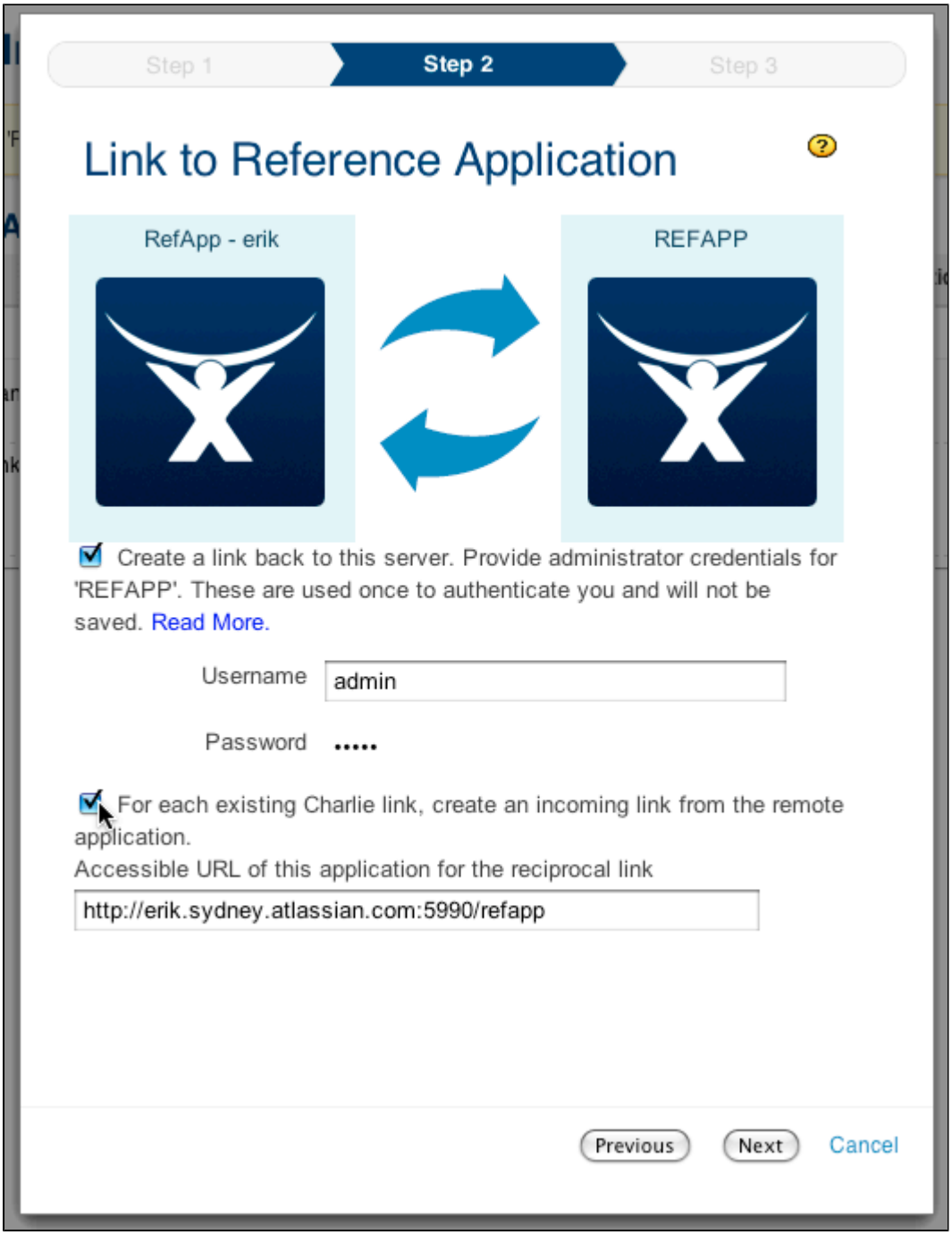
If you upgrade your remote application to a version that does include Application Links, the application link will continue to work. However, upgrading your link may simplify link configuration and make additional authentication protocols available (as mentioned above).

To upgrade an application link when your remote application has been upgraded to include Application Links:

1. After you have upgraded your remote application to a version that includes Application Links, go to the administration console of your local application. A warning will be displayed, requesting that you upgrade the link to full Application Links mode.
2. Click '**Upgrade**' in the warning message to start the upgrade wizard. Note the following:
    - You will be prompted to make your application link a reciprocal link. You will need to provide administrator credentials for your remote application, if you choose to do so.
    - If you make your application link a reciprocal link, you will also be able to make reciprocal links for your project links. For example, you may be able to link your JIRA project to a FishEye repository and also make a link from your FishEye repository back to the JIRA project.



Screenshot above: Upgrading an application link for remote application



Screenshot above: Upgrading an application link wizard

Notes

Related Topics

Adding an Application Link
Configuring Authentication for an Application Link

Deleting an Application Link

Deleting an application link stops two linked applications from sharing information. Once an application link is deleted, you will no longer be able to make requests from one application to the other and vice versa. This means that certain features may not work, for e.g. Inserting JIRA issues in Confluence, Confluence Page Gadget in JIRA, etc.

If you have set up application links to multiple servers of the same application type, e.g. you have linked your application to multiple JIRA servers, deleting the primary link will mean that another of the links will be made the primary link.

Deleting an application link will also delete all project links set up for that application link.

> *The information on this page does not apply to **JIRA OnDemand**.*

To delete an application link:

1. Log in as a system administrator and go to the administration page. Click 'Application Links' in the administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
2. Click the '**Delete**' link next to the application link that you want to delete. A confirmation screen will be displayed.
3. Click the '**Confirm**' button to delete the application link.

Related Topics

Editing an Application Link
Relocating an Application Link

Configuring Project Links across Applications

Let's assume that you are managing a project or team. You would like to connect your project's Confluence space with your JIRA project, and link up your team's source repository too.
When you have connected your applications via Application Links, you can also connect the areas of those applications that contain information relating to your project or team. Using **project links** (also called **entity links**) you can associate one or more projects, spaces and repositories across the linked applications.

To connect all the information relating to the project or team that you are managing, you can link one or more of the following:

- JIRA projects.
- Confluence spaces.
- FishEye repositories.
- FishEye projects. A FishEye 'project' is the Crucible project if you have installed FishEye and Crucible, otherwise it is the paths associated via the 'FishEye Project Content' function in FishEye.
- Crucible projects.
- Bamboo projects.

On this page:

- Uses for Project Links
- Managing Project Links

> *The information on this page does not apply to **JIRA OnDemand**.*

Uses for Project Links

The following integration features use project links:

- Activity streams. For example, the project links determine the activity retrieved from JIRA to display in the activity stream of a FishEye repository or a Crucible project.
- The JIRA FishEye plugin. For example:
  - The link between a JIRA project and a FishEye repository determines the repository searched for a particular issue key when displaying the FishEye source tab in JIRA.
  - The link between a JIRA project and a Crucible project determines the Crucible project scanned for review activity when displaying the Crucible reviews tab in JIRA.
  - When you create a defect in Crucible, Crucible will know which JIRA project to put it in.
- Third-party plugins may make use of project links to enrich their functionality too.

Managing Project Links

- Adding Project Links between Applications
- Making a Project Link the Primary Link
- Deleting a Project Link

RELATED TOPICS

Adding an Application Link

Adding Project Links between Applications

Let's assume that you are managing a project or team. You would like to connect your project's Confluence space with your JIRA project, and link up your team's source repository too.
When you have connected your applications via Application Links, you can also connect the areas of those applications that contain information relating to your project or team. Using **project links** (also called **entity links**) you can associate one or more projects, spaces and repositories across the linked applications.

To connect all the information relating to the project or team that you are managing, you can link one or more of the following:

- JIRA projects.
- Confluence spaces.
- FishEye repositories.
- FishEye projects. A FishEye 'project' is the Crucible project if you have installed FishEye and Crucible, otherwise it is the paths associated via the 'FishEye Project Content' function in FishEye.
- Crucible projects.
- Bamboo projects.

On this page:

- Adding a Project Link
- Notes

*The information on this page does not apply to **JIRA OnDemand**.*

Adding a Project Link

To link a JIRA project to a project or space in another application:

1. Log in as a user with the **'JIRA Administrators'** global permission and navigate to the administration page for the project. See Defining a Project for details.
2. On the the 'Projects' page, choose the JIRA project that you want to link to another project.
3. On the project's configuration page, locate the 'Application Links' section and click '**Configure Project Links**'.
4. The instructions for adding a project link will vary depending on whether the target application has the Application Links functionality installed:
    - If the target application has Application Links:
        a. Click '**Add Link**'. A dropdown menu will appear listing the applications you have already linked to.
        b. In the dropdown menu, click the application that contains the project you want to link to. For example, if you want to link to a specific JIRA project, click the JIRA site that contains that project. If you want to link to a Confluence space, click the Confluence site that contains that space.
        c. Click one of the options on the 'Authorization required' screen:
            - '**Authorize**' — Click this option if you want to grant your project authorised access to the target project. The target application will open in a new window, so that you can log in and authorise access.
            - '**Skip – your access is anonymous**' — Click this option if you only want to allow anonymous access to the target project.
        d. In the '**Name or Key**' field, enter the name/key of the project in the remote application that you want to link to. For example, if you want to link to a JIRA project, enter the project key. If you want to link to a Confluence space, enter the space key.
        e. Click the '**Create**' button to create the project link.
    - If the target application does not have Application Links:
        a. Click '**Add Link**'. A dropdown menu will display listing the applications you have already linked to.
        b. In the dropdown menu, click the application that contains the project you want to link to. For example, if you want to link to a specific JIRA project, click the JIRA site that contains that project. If you want to link to a Confluence space, click the Confluence site that contains that space.
        c. In the '**Key**' field, enter the name/key of the project in the remote application that you want to link to. For example, if you want to link to a JIRA project, enter the project key. If you want to link to a Confluence space, enter the space key.
        d. *(optional)* Enter the alias for the project in the '**Alias**' field. This is the display name for the project in your administration console.
        e. Click the '**Create**' button to create the project link.

Step 1 Step 2 Step 3

Screenshots above: Linking to another JIRA project (where the target JIRA server supports Application Links)

Notes

Related Topics

Making a Project Link the Primary Link
Deleting a Project Link

Making a Project Link the Primary Link

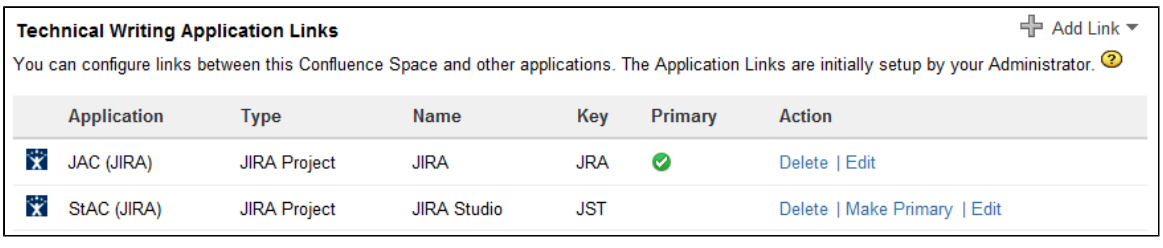
If you have set up project links to more than one project in the same application, for example you have linked your JIRA project to two Confluence spaces, then one of the project links will be marked as the primary link. All outgoing requests will be directed to the primary link.

For example, if you have a JIRA project that is linked to two Confluence spaces, you can nominate the link to Confluence spaces as the primary link. Every time JIRA requests Confluence information, it will request it from the primary link's Confluence space. Note, both Confluence spaces can still request information from the JIRA project.

> *The information on this page does not apply to **JIRA OnDemand**.*

To make a project link the primary link:

1. Log in as a user with the **'JIRA Administrators'** global permission and navigate to the administration page for the project. See Defining a Project for details.
2. On the the 'Projects' page, choose the JIRA project that you want to link to another project.
3. On the project's configuration page, locate the 'Application Links' section and click '**Configure Application Links**'.
4. Click the '**Make Primary**' link in the '**Action**' column for the project link that you want to make the primary link. A symbol will display in the 'Primary' column next to the link.
   Note: The 'Primary' column and 'Make Primary' link will appear only if you have set up multiple project links to the same application, for example you have linked a Confluence space to a number of JIRA projects.



Screenshot above: Viewing the project links for a JIRA project

Related Topics

Adding Project Links between Applications
Deleting a Project Link

Deleting a Project Link

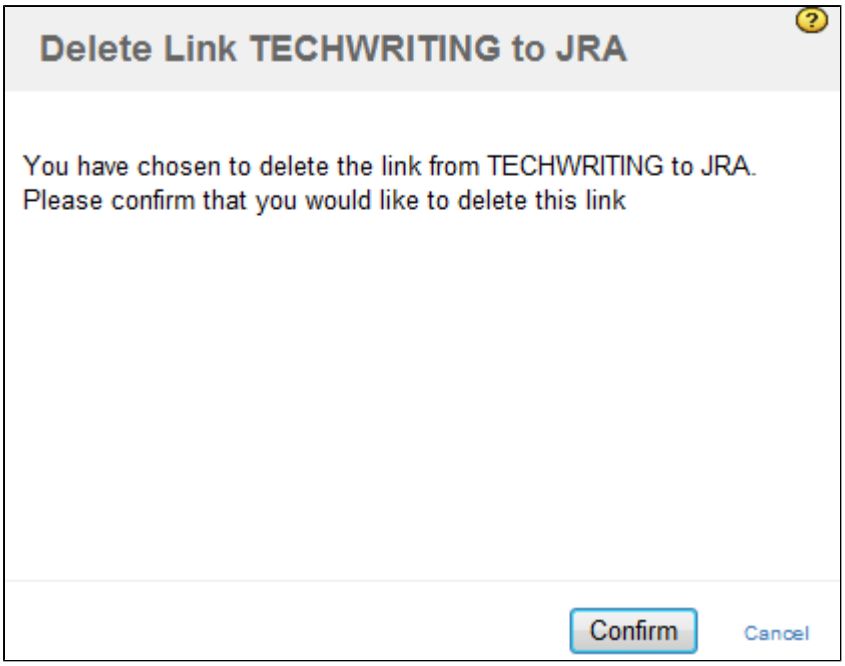
Deleting a project link stops the two projects from sharing information.

If you have set up multiple project links to the same application, for example you have linked a JIRA project to multiple Confluence spaces, deleting the primary link will mean that another of the links will be made the primary link.

> *The information on this page does not apply to **JIRA OnDemand**.*

To delete a project link:

1. Log in as a user with the **'JIRA Administrators'** global permission and navigate to the administration page for the project. See Defining a Project for details.
2. On the the 'Projects' page, choose the JIRA project that you want to link to another project.
3. On the project's configuration page, locate the 'Application Links' section and click '**Configure Application Links**'.
4. Click the '**Delete**' link next to the project link that you want to delete. A confirmation screen will be displayed.
5. Click the '**Confirm**' button to delete the project link.



Screenshot above: Confirming the deletion of a project link

Related Topics

Adding Project Links between Applications
Making a Project Link the Primary Link

Configuring Issue Cloning

JIRA's issue cloning behaviour can be modified by JIRA system administrators.

> *The information on this page does not apply to **JIRA OnDemand**.*

Configuring cloned issue linking behaviour

By default, when an issue is cloned, JIRA will automatically create a link between the original and cloned issue using the pre-existing link type name 'Cloners'.

You can change this default behaviour by editing the `jira.clone.linktype.name` property of your jira-config.properties file.
If this property does not exist in your `jira-config.properties` file, add it to the file.

- If this property has a value, JIRA will use the pre-existing link type whose name is the value specified for this property.
- If this property has no value, JIRA will not create links between original and cloned issues.

Configuring the cloned issue summary field prefix

By default, the 'Summary' field of a cloned issue is prefixed with the string 'CLONE - ' to indicate that the issue is a clone.

To change this prefix or prevent the addition of prefixes on cloned issues:

1. Access JIRA's **Advanced Settings** page. (See Configuring Advanced Settings for more information.)
2. Edit the value of the `jira.clone.prefix` property by clicking the existing value and specifying a different prefix for the 'Summary' field of cloned issues.
   Specifying no value prevents a prefix being added to the 'Summary' field of cloned issues.
3. Click the '**Update**' button to save the new value in the JIRA database.

Configuring Issue Linking

About issue linking

Issue linking allows you to create an association between issues on either the same or different JIRA servers. For instance, an issue may *duplicate* another, or its resolution may *depend* on another's. New installations of JIRA come with four default types of links:

- **relates to** / **relates to**
- **duplicates** / **is duplicated by**
- **blocks** / **is blocked by**
- **clones** / **is cloned by** (this is used when issues are cloned)

Issue linking also allows you to:

- Create an association between a JIRA issue and a Confluence page.
- Link a JIRA issue to any other web page.

You can add, edit or delete link types to suit your organisation, as described below.

Note:

- Your users must have the **Link Issues** permission before they can link issues.
- Issue linking must be enabled in order for your users to be able to link issues. **Issue linking is enabled by default.** If your organisation does not require the ability to link issues, you can disable it globally for all users as described below.
- If you want to link JIRA issues to those on a different JIRA server or to Confluence pages, see Configuring issue linking for external applications (below) for details on how to set this up.

On this page:

- About issue linking
- Adding a link type
- Editing or deleting a link type
- Configuring issue linking for external applications
- Disabling issue linking

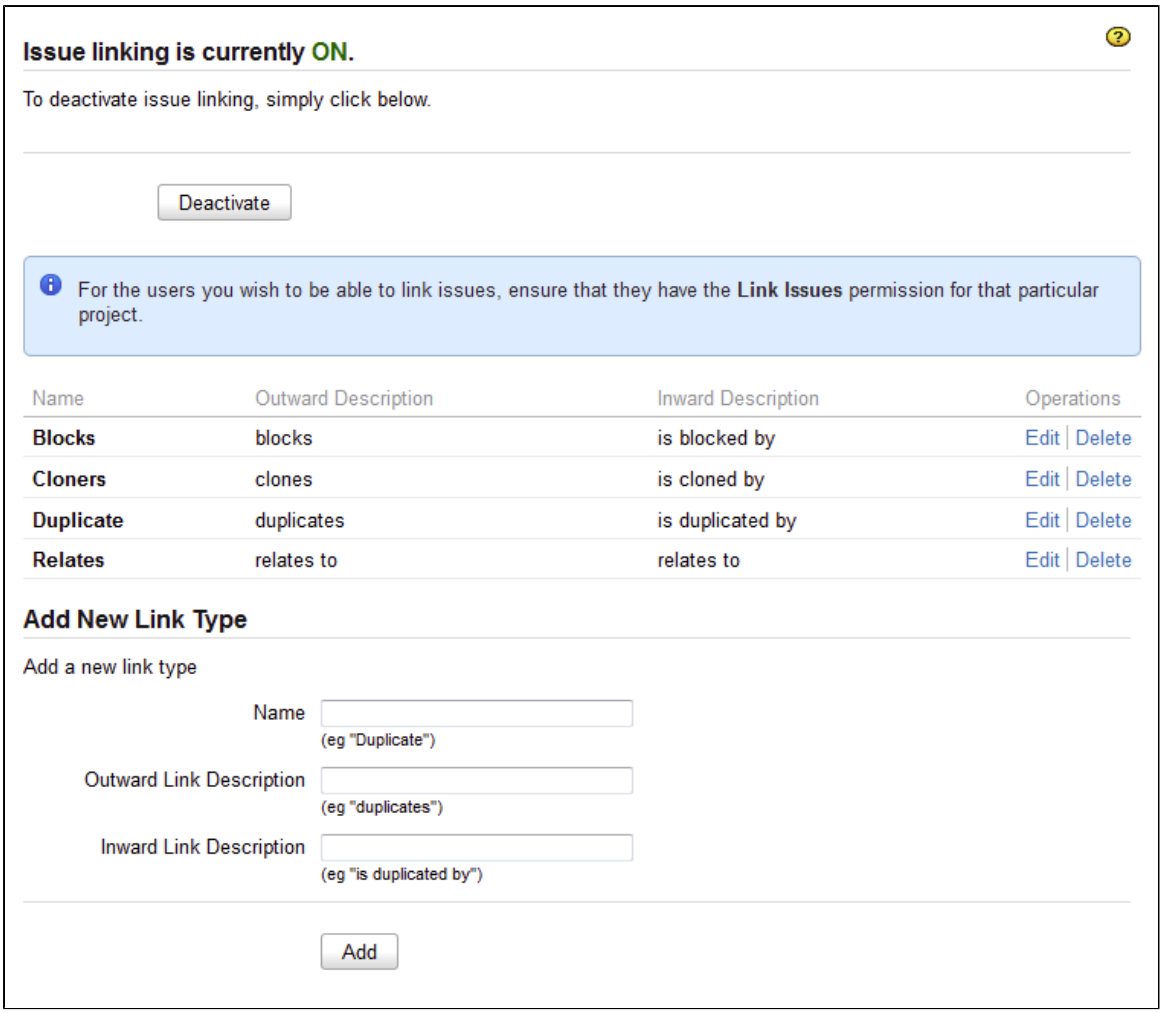
*The information on this page does not apply to **JIRA OnDemand**.*

Adding a link type

To create a new link type, e.g. 'Causes':

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Issue Features > Issue Linking** (tab) to open the 'Issue Linking' page as shown in the screenshot below.
   Keyboard shortcut: **g + g +** start typing **issue linking**
3. In the 'Add New Link Type' form at the end of the page:
   - Enter 'Causes' in the **Name** text field.
   - Enter 'causes' in the **Outward Link Description** text field.
   - Enter 'is caused by' in the **Inward Link Description** text field.
4. Click the **Add** button.
5. This returns to the **Issue Linking** page, with a new section listing the **Causes** link type.

Screenshot: the 'Issue Linking' administration page



Editing or deleting a link type

It is recommended that you do not edit or delete the **Clones** link type, as this is used to automatically link issues when they are cloned.

To edit or delete a link type:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Issue Features > Issue Linking** (tab) to open the **Issue Linking** page as shown in the screenshot above.
   Keyboard shortcut: **g + g +** start typing **issue linking**
3. Locate the link type you wish to edit or delete, and click the link type's associated **Edit**/**Delete** link in the **Operations** column.

Configuring issue linking for external applications

It is possible to create links to issues on a remote JIRA site or pages on a Confluence site. To do this, create *fully reciprocal application links* between your JIRA site to the remote JIRA or Confluence site. Fully reciprocal application links mean that:

- An application link must be configured on each server to the other.
- Each of these application links must have both incoming and outgoing authentication configured to each other's servers.

To configure fully reciprocal application links between your JIRA site and a remote JIRA or Confluence site:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Create an application link to your remote JIRA or Confluence site. (See Adding an Application Link for details.) When creating the link:
   - During step 2 of the wizard, ensure you choose the option to create a link from the remote server back to your server.
   - During step 3 of the wizard, choose the **These servers fully trust each other** option. This will ensure that incoming and outgoing authentication is configured for the application link on each server to the other server.
3. If you configured a fully reciprocal application links between your JIRA site and a Confluence site, ensure that the Confluence site's system administrator has enabled the **Remote API (XML-RPC & SOAP)** feature, since this Confluence feature is disabled by default. See Enabling the Remote API in the Confluence documentation for details.
   If you do not enable this feature, JIRA will not be able to communicate with Confluence. As a result, your users:
   - Will see **Failed to load** messages in the Confluence Wiki page links they create on JIRA issues.
   - Will not be able to search for Confluence pages using the **Find a Confluence page** dialog box.

Please Note: You can create a one-way application link from your JIRA site to a remote JIRA site or Confluence site. However, some loss of functionality will be experienced by your users when they create remote links. For instance, if your users create a link to a remote JIRA issue, they will find that the **Create reciprocal link** check box on the **Link** dialog box will not function correctly. Hence, it is recommended that you create fully reciprocal links instead.

Disabling issue linking

To disable issue linking for your entire JIRA site, for all users:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > System > Issue Features > Issue Linking** (tab) to open the **Issue Linking** page as shown in the screenshot above.
   Keyboard shortcut: **g + g +** start typing **issue linking**
3. A status message indicates whether issue linking is enabled. If issue linking is enabled, click the **Deactivate** button. The **Issue Linking** page reloads, stating that linking is disabled.

Configuring the Whitelist

What is the 'Whitelist'?

For security reasons, you as an administrator may wish to limit the URLs from which users can source content that is displayed on your JIRA site (e.g. in an External Gadget). The JIRA 'Whitelist' is a list of URLs whose content you wish to make available to users of your JIRA site.

You can add URLs (or URL patterns) to your whitelist as described below. Alternatively, if your JIRA site and users do not have access to the internet, you can choose to 'Allow all URLs' (see below).

Note that URLs for which Application Links are configured are automatically whitelisted, so you do not need to add them to this list.

> *The information on this page does not apply to **JIRA OnDemand**.*

Editing the Whitelist

You can list specific URLs (or URL patterns) from which content will be allowed onto your JIRA site.

Select 'Restrict to whitelisted URL patterns' and use the form below to list specific URLs or URL patterns that will be allowed. If you select 'Allow all URLs', content can be included from any URL, including possibly malicious content.

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Security' > 'Whitelist'** (tab) to open the 'Whitelist' page, which shows a list of URLs (or URL patterns).
   Keyboard shortcut: **'g' + 'g' +** type **'wh'**
3. On the 'Whitelist' page, you can either:
   - Select '**Restrict to whitelisted URL patterns**' and use the form below to list specific URLs or URL patterns that will be allowed.
     - Enter URL patterns to describe valid content sources. Enter one pattern per line according to the following format:
       - if the pattern starts with '=', only the exact URL following the '=' will be allowed
       - if the pattern starts with '/' then the whole pattern will be treated as a regular expression
       - otherwise, * characters in the pattern will be treated as wildcards to match 1 or more characters
     - For example, if you want to allow all requests to

       http://www.atlassian.com

       , enter the following rule:

       - http://www.atlassian.com/*
   - Select '**Allow all URLs**'. This will allow content to be included from any URL, including potentially malicious content.

Configuring Sub-tasks

Sub-Task issues are generally used to split up a parent issue into a number of tasks which can be assigned and tracked separately. (For details, see Creating a Sub-Task.)

Sub-Tasks have all the same fields as standard issues, although note that their 'issue type' must be one of the *Sub-Task issue types* (see below) rather than one of the standard issue types.

If Sub-Tasks are enabled and you have defined at least one Sub-Task issue type, your users will be able to:

- create Sub-Tasks.
- convert issues to Sub-Tasks (and vice versa).

On this page:

- Disabling Sub-Tasks
- Enabling Sub-Tasks
- Defining Sub-Task issue types
- Blocking Issue workflows by Sub-Task status
- Configuring Sub-Task fields
  - Choosing which Sub-Task fields to display on the Parent Issue
  - Choosing which fields to display in the 'Add Sub-Task' form

Disabling Sub-Tasks

Sub-tasks are enabled by default. However, this feature can be disabled from the **Sub-Tasks** administration page.

Sub-Tasks will be disabled by default if your JIRA installation was upgraded from a version prior to 4.2 that had Sub-Tasks disabled.

To disable Sub-Tasks:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Types' > 'Sub-Tasks'** (tab) to open the 'Sub-Tasks' page.
   Keyboard shortcut: **'g' + 'g' +** type **'sub'**
3. Click the '**Disable** Sub-Tasks' link. The page reloads and informs you that Sub-Tasks are now disabled.

Please Note: Sub-tasks cannot be disabled if one or more Sub-Tasks exists in the system. You must remove any existing Sub-Tasks (or convert them to standard issues) before you can disable this feature.

Enabling Sub-Tasks

Sub-tasks can be enabled from the Sub-Task administration screen.

To enable Sub-Tasks:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Issues' > 'Issue Types' > 'Sub-Tasks'** (tab) to open the 'Sub-Tasks' page.
   Keyboard shortcut: **'g' + 'g' +** type **'sub'**
3. Click the '**Enable** Sub-Tasks' link. The page will reload and inform you that the Sub-Tasks are now enabled.
   A default Sub-Task issue type is automatically available for use. You can edit it by clicking its **'Edit'** link in the **Operations** column.

Defining Sub-Task issue types

Sub-tasks must be assigned one of the *Sub-Task issue types*, which are different to standard issue types. Please note that you must define at least one Sub-Task issue type before users can create Sub-Tasks.

Sub-task issue types can be customised on the **Sub-Tasks** administration page (described above). The **Sub-Tasks** administration page also allows you to create, delete, edit, translate and choose icons for your Sub-Task issue types. For details, please see Defining Issue Types.

Blocking Issue workflows by Sub-Task status

It is possible to restrict the progression of an issue through workflow depending on the status of the issue's Sub-Tasks. For example, you might need to restrict an issue from being resolved until all of its Sub-Tasks are resolved. To achieve this, you would create a custom workflow and use the *Sub-Task Blocking Condition* on the workflow transitions that are to be restricted by the Sub-Tasks' status.

Configuring Sub-Task fields

You can choose which subtask fields are displayed in the 'Sub-Tasks' section of an issue, by customising JIRA's advanced configuration options.

Choosing which Sub-Task fields to display on the Parent Issue

- To customise the Sub-Task columns that appear on the parent issue's **View Issue** screen, edit the `jira.table.cols.subtasks` property on JIRA's 'Advanced Settings' page.

Choosing which fields to display in the 'Add Sub-Task' form

- To customise the Sub-Task 'Quick Create' form, edit the `jira.subtask.quickcreateform.fields` property of your jira-config.properties file.
  For a list of available fields to include, see Displaying Search Results in XML.

Managing Shared Filters

A **filter** is a saved issue search. JIRA users can create and manage their own filters (see Saving Searches ('Issue Filters')) and filter subscriptions (see Receiving Search Results via Email).

A **shared filter** is a filter whose creator has shared that filter with other users. When a shared filter is created by a user, that user:

- Initially 'owns' the shared filter.
- Being the owner, can edit and modify the shared filter.

JIRA administrators can change the ownership of any user's shared filter, which allows the shared filter to be edited and modified by its new owner.

Changing the Ownership of a Shared Filter

Before changing the ownership of a shared filter, ensure that you inform the shared filter's current owner of your intentions.

To change the ownership of a shared filter:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Users' > 'Shared Filters'** to open the 'Search Shared Filters' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'shared filters'**

3. Enter your search criteria into the 'Search' field and click the '**Search**' button. A list of shared filters matching your search criteria is shown below. Each shared filter indicates its:
   - Current owner — this is originally the user who created the shared filter
   - List of shares applied to the shared filter by its owner
   - Popularity — the number of users who have selected that shared filter as a 'favourite'.
4. Click the 'cog' icon to the right of the shared filter whose ownership you wish to change and select **'Change Owner'**.
5. In the 'Change Owner' dialog box, enter the username (or name) of the user who will become the new owner of the shared filter.
6. Select the appropriate user from the dropdown list and click the **'Change Owner'** button.

Please Note:

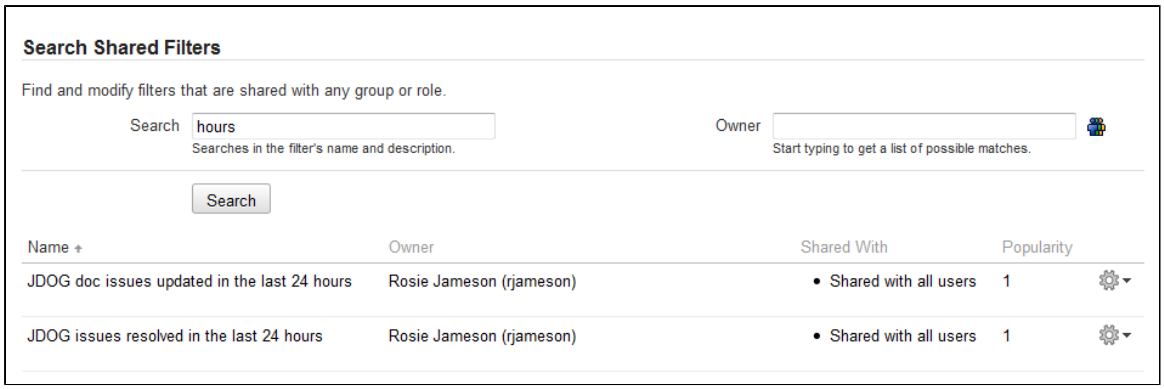
- A shared filter can only be edited by the shared filter's owner. The owner of a shared filter can only modify that filter's shares and search criteria too. See Saving Searches ('Issue Filters') for more information.
- You cannot change the ownership of a shared filter to a user who:
  - already has a shared filter with exactly the same name, or
  - does not have permission to view the shared filter.

Deleting a Shared Filter

Before deleting a shared filter, then out of common courtesy, ensure that you inform the current owner of the shared filter of your intentions.

To delete a shared filter:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Users' > 'Shared Filters'** to open the 'Search Shared Filters' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'shared filters'**



3. Enter your search criteria into the 'Search' field and click the '**Search**' button. A list of shared filters matching your search criteria is shown below. Each shared filter indicates its:
    - Current owner — this is originally the user who created the shared filter
    - List of shares applied to the shared filter by its owner
    - Popularity — the number of users who have marked that shared filter as a 'favourite'.
4. Click the 'cog' icon to the right of the shared filter you wish to delete and select **'Delete Filter'**. The 'Delete Filter' dialog box is shown.
    - The number of users who have marked the shared filter as a favourite is specified in this dialog box.
    - If any subscriptions are associated with this shared filter, a numbered link is provided leading to a page which indicates the shared filter's current subscribers.
5. If you are happy to proceed, click the **'Delete'** button to complete the action.

On this page:

- Changing the Ownership of a Shared Filter
- Deleting a Shared Filter

RELATED TOPICS

- Saving Searches ('Issue Filters')
- Receiving Search Results via Email

Managing Shared Dashboards

A **dashboard** is a customisable page that can display many different types of information, depending on your areas of interest. JIRA users can create and manage their own dashboards (see Managing Multiple Dashboard Pages).

A **shared dashboard** is a dashboard whose creator has shared that dashboard with other users. When a shared dashboard is created by a user, that user:

- Initially 'owns' the shared dashboard.
- Being the owner, can edit and modify the shared dashboard.

JIRA administrators can change the ownership of any user's shared dashboard, which allows the shared dashboard to be edited and modified by its new owner.

Changing the Ownership of a Shared Dashboard

Before changing the ownership of a shared dashboard, ensure that you inform the shared dashboard's current owner of your intentions.

To change the ownership of a shared dashboard:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Users' > 'Shared Dashboards'** to open the 'Search Shared Dashboards' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'shared dashboards'**

3. Enter your search criteria into the 'Search' field and click the '**Search**' button. A list of shared dashboards matching your search criteria is shown below. Each shared dashboard indicates its:
    - Current owner — this is originally the user who created the shared dashboard
    - List of shares applied to the shared dashboard by its owner
    - Popularity — the number of users who have selected that shared dashboard as a 'favourite'.
4. Click the 'cog' icon to the right of the shared dashboard whose ownership you wish to change and select **'Change Owner'**.
5. In the 'Change Owner' dialog box, enter the username (or name) of the user who will become the new owner of the shared dashboard.
6. Select the appropriate user from the dropdown list and click the **'Change Owner'** button.

Please Note:

- A shared dashboard can only be edited by the shared dashboard's owner. The owner of a shared dashboard can only modify that dashboard's shares and gadgets too. See Managing Multiple Dashboard Pages and Customising the Dashboard for more information.
- You cannot change the ownership of a shared dashboard to a user who:
    - already has a shared dashboard with exactly the same name, or
    - does not have permission to view the shared dashboard.

Deleting a Shared Dashboard

Before deleting a shared dashboard, ensure that you inform the shared dashboard's current owner of your intentions.

To delete a shared dashboard:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Users' > 'Shared Dashboards'** to open the 'Search Shared Dashboards' page.
   Keyboard shortcut: **'g' + 'g' +** start typing **'shared dashboard'**

3. Enter your search criteria into the 'Search' field and click the '**Search**' button. A list of shared dashboards matching your search criteria is shown below. Each shared dashboard indicates its:
    - Current owner — this is originally the user who created the shared dashboard
    - List of shares applied to the shared dashboard by its owner
    - Popularity — the number of users who have marked that shared dashboard as a 'favourite'.
4. Click the 'cog' icon to the right of the shared dashboard you wish to delete and select **'Delete Dashboard'**. The 'Delete Dashboard' confirmation message box is shown.
    - The number of users who have marked the shared dashboard as a favourite is specified in this message box.
5. If you are happy to proceed, click the **'Delete'** button to complete the action.

On this page:

- Changing the Ownership of a Shared Dashboard
- Deleting a Shared Dashboard

RELATED TOPICS

- Managing Multiple Dashboard Pages
- Customising the Dashboard

Server Administration

- Increasing JIRA Memory
- Using the Database Integrity Checker
- Precompiling JSP pages
- Database Indexing
- Logging and Profiling
- Restoring Data
- Optimising Performance
- Backing Up Data
- Search Indexing
- Using robots.txt to hide from Search Engines
- Updating your JIRA License Details
- Viewing your System Information
- Generating a Thread Dump
- Performance Testing Scripts
- Finding the JIRA Support Entitlement Number (SEN)

Increasing JIRA Memory

Java applications like JIRA and Confluence run in a "Java virtual machine" (JVM), instead of directly within an operating system. When started, the Java virtual machine is allocated a certain amount of memory, which it makes available to applications like JIRA. By default, Java virtual machines are allocated 64 MB of memory, no matter how many gigabytes of memory your server may actually have available. 64 MB is inadequate for medium to large JIRA installations, and so this needs to be increased. Seeing OutOfMemoryErrors in the logs is symptomatic of this.

This page addresses how to increase Heap Space memory. Confirm that you're not receiving Perm Gen or GC Overhead errors.

On this page:

- Step 1: Diagnosis
- Step 2: Increase Available Memory
- Step 3: Verify Your Settings

*The information on this page does not apply to **JIRA OnDemand**.*

Step 1: Diagnosis

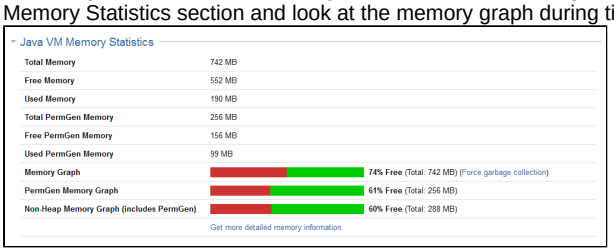
Expand to see diagnosis section

Assess Root Cause

Often, there is a root cause for OutOfMemory Errors that may be better to address than just increasing memory. See JIRA Crashes Due to 'OutOfMemoryError Java heap space' for a discussion.

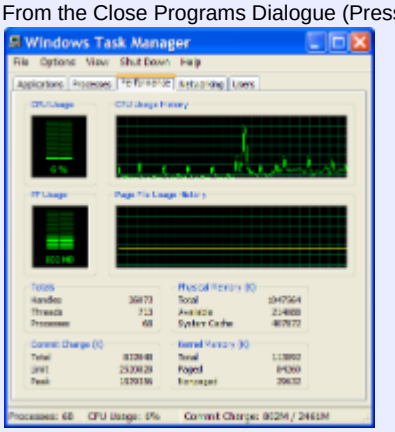
Determine JIRA usage patterns

In JIRA, go to **Administration > System > Troubleshooting and Support > System Info** and scroll down the page to view the Java VM Memory Statistics section and look at the memory graph during times of peak usage:



This server has been allocated a maximum of 768 MB and a minimum of 256 MB (typically defined in the `setenv` script which is executed by running the `start-jira` script). If you are trying to see whether your settings are being picked up by JIRA, this is where to look. Here, you can see that JIRA has reserved 742 MB, or which 190 MB is actually in use. If this JIRA instance were running out of memory, it would have reserved the maximum available (768 MB), and would be using an amount close to this.

Determine available system memory

On Windows
From the Close Programs Dialogue (Press ctrl-alt-delete), select the Performance tab:    The amount marked **Available** is the amount in kilobytes you have free to allocate to JIRA. On this server we should allocate at most 214 MB.

On Linux
Run `cat /proc/meminfo` to view the memory usage.

Setting the -Xmx above the available amount on the server runs the risk of OutOfMemoryErrors due to lack of physical memory. If that occurs the system will use swap space, which greatly decreases performance.

Guidance

As a rule of thumb, if you have fewer than 5000 issues, JIRA should run well with the default 768 MB. Granting JIRA too much memory can impact performance negatively, so it is best to start with 768 MB and make modest increases as necessary. As another data point, 40,000 works well with 768 MB to 1 GB.

Step 2: Increase Available Memory

Linux

▼ Expand to see Linux instructions

To increase heap space memory in Linux installations:
1. In your <`JIRA Installation Directory`>`/bin` (or <`Tomcat Installation Directory`>`/bin` for JIRA WAR installations), open the `setenv.sh` file. 2. Find the sections **JVM_MINIMUM_MEMORY=** and **JVM_MAXIMUM_MEMORY=** 3. See Diagnosis above and enter the appropriate values.

Windows (starting from .bat file)

▼ Expand to see Windows .bat file instructions

To Configure System Properties in Windows Installations When Starting from the .bat File:
1. In your <`JIRA Installation Directory`>`/bin` (or <`Tomcat Installation Directory`>`/bin` for JIRA WAR installations), open the `setenv.bat` file. 2. Find the section **set JVM_MINIMUM_MEMORY=** and **set JVM_MAXIMUM_MEMORY=** 3. See Diagnosis above and enter the appropriate values.

Windows Service

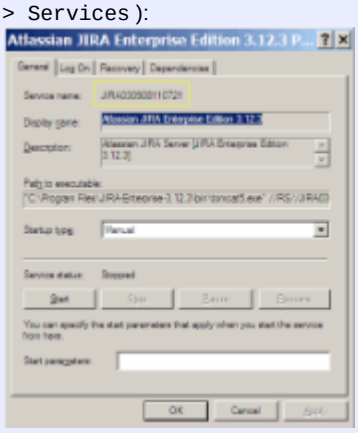
▼ Expand to see Windows Service instructions

There are two ways to configure system properties when starting Running JIRA as a Service, either via command line or in the Windows Registry.

Setting Properties for Windows Services via Command Line

Setting Properties for Windows Services via Command Line

1. Identify the name of the service that JIRA is installed as in Windows (`Control Panel > Administrative Tools > Services`):

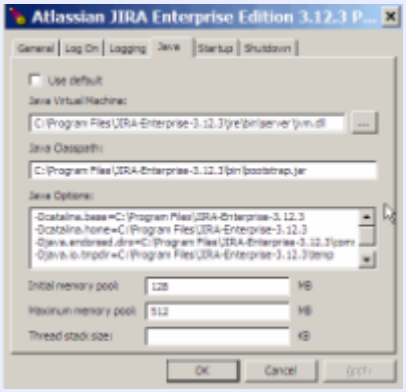


   In the above example, the **SERVICENAME** is: `JIRA030908110721`
2. Open the command window from `Start > Run > type in 'cmd' > press 'Enter'`
3. `cd` to the `bin` subdirectory of your JIRA Installation Directory (or the `bin` subdirectory of your Tomcat installation directory if your are running the JIRA WAR distribution).
4. Run:

```
tomcat6w //ES//%SERVICENAME%
```

   In the above example, it would be `tomcat6w //ES//JIRA030908110721`

5. Click on the `Java` tab to see the list of current start-up options:

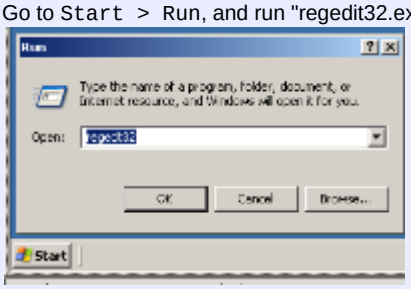


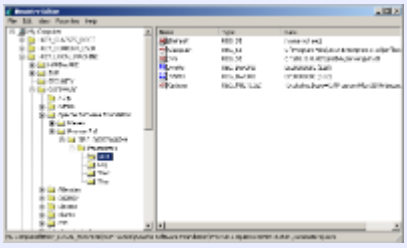
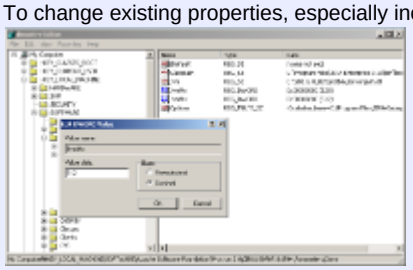
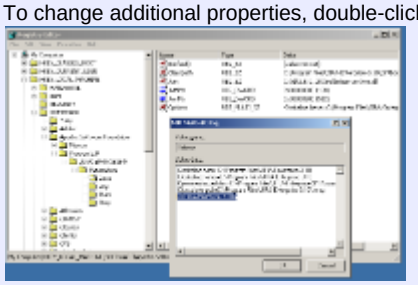
6. Set the maximum memory allocation here

Setting Properties for Windows Services via the Windows Registry

In some versions of Windows, there is no option to add Java variables to the service. In these cases, you must add the properties by viewing the option list in the registry.

To Set Properties for Windows Services via the Windows Registry,

1. Go to `Start > Run`, and run "regedit32.exe".


2. Find the Services entry:
32-bit: `HKEY_LOCAL_MACHINE > SOFTWARE > Apache Software Foundation > Procrun 2.0 > JIRA`
64-bit: `HKEY_LOCAL_MACHINE > SOFTWARE > Wow6432Node > Apache Software Foundation > Procrun 2.0 > JIRA`

3. To change existing properties, especially increasing Xmx memory, double-click the appropriate value.

4. To change additional properties, double-click options.

5. Modify the memory allocations here.

Step 3: Verify Your Settings

▼ Expand to see verification instructions

To verify what settings are in place, check the <`JIRA Home Directory`>`/logs/atlassian-jira.log` or `catalina.out` file. A section in the startup appears like this:

```
JVM Input Arguments : -Djava.util.logging.config.file=/usr/local/jira/conf/logging.properties
-XX:MaxPermSize=256m -Xms256m -Xmx384m -Djava.awt.headless=true -Datlassian.standalone=JIRA
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true -Dmail.mime.decodeparameters=true
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djava.endorsed.dirs=/usr/local/jira/endorsed -Dcatalina.base=/usr/local/jira
-Dcatalina.home=/usr/local/jira -Djava.io.tmpdir=/usr/local/jira/temp
```

Look for Xmx (maximum) and Xms (minimum) settings.

This display is also available by Viewing your System Information.

Using the Database Integrity Checker

Searching for common data inconsistencies, the **Database Integrity Checker** attempts to ensure that all JIRA data is in a consistent state.

This is useful in a number of situations, e.g.

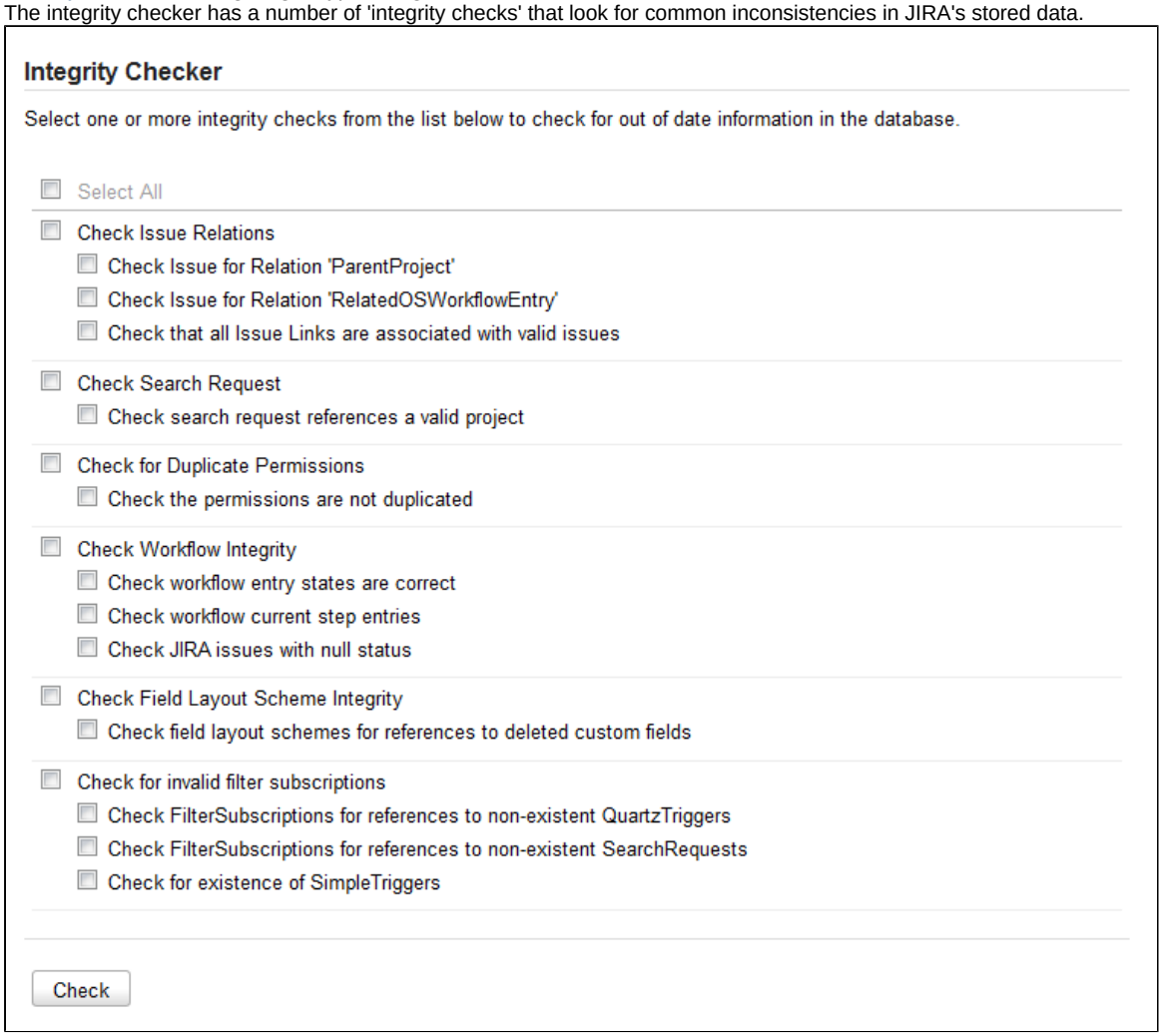
- Before migrating a project to a new workflow
- An external program is modifying JIRA's database
- Troubleshooting a server crash

If an error is encountered, most of the integrity checks provide a 'repair' option which attempts to reset the data to a stable state.

> *The information on this page does not apply to **JIRA OnDemand**.*

Using the Integrity Checker

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Troubleshooting and Support' > 'Integrity Checker'** (tab) to open the 'Integrity Checker' page.
   Keyboard shortcut: **'g' + 'g' +** type **'integ'**
   The integrity checker has a number of 'integrity checks' that look for common inconsistencies in JIRA's stored data.



3. Select one or more items whose data you would like to check the integrity of and click the **'Check'** button.
4. After the selected checks run, the preview screen will be shown.
   The screen provides details about the existing data inconsistencies. If any inconsistencies were found, the **'Fix'** button will also appear on the page. The messages in red describe inconsistencies that the check will correct if it is chosen and the **'Fix'** button is clicked. Messages that appear in yellow are warnings that the check will not correct; JIRA will auto-recover from these inconsistencies when an action is taken on an issue.
   Select any inconsistencies that you would like to correct, then click the **'Fix'** button.
   Please Note: We **strongly** recommend taking a backup of your data before correcting any data inconsistencies.
5. If any inconsistencies were found and you chose to correct them, you will be presented with a summary screen describing all the corrective actions that have taken place.

Precompiling JSP pages

If you decided to go the extra mile and extend JIRA's build process to precompile JSP pages, keep in mind that the "include" directory in the JIRA web application needs to be excluded from precompilation. The reason for this is that the JSP files in the "include" directory are not proper JSP files, but are includes that are only meant to be compiled as part of larger JSP pages.

For example, to exclude the JSP pages in the "include" directory when using Maven use the <exclude> sub-element of the <ant:jspc> task, as shown:

> *The information on this page does not apply to **JIRA OnDemand**.*

```
<ant:path id="jspc.classpath">
      <ant:pathelement location="${tomcat.home}/common/lib/jasper-runtime.jar"/>
      <ant:pathelement location="${tomcat.home}/common/lib/jasper-compiler.jar"/>
      <ant:pathelement location="${tomcat.home}/common/lib/servlet.jar"/>
      <ant:path refid="maven-classpath"/>
      <ant:path refid="maven.dependency.classpath"/>
      <ant:pathelement path="${maven.build.dest}"/>
      <ant:pathelement path="${java.home}/lib/tools.jar"/>
    </ant:path>
    <ant:jspc
      package="${pom.package}.jsp"
      destDir="${jspOutDir}"
      srcdir="${warSource}"
      uriroot="${warSource}"
      uribase="/${pom.artifactId}"
      verbose="2"
      classpathref="jspc.classpath">
      <ant:include name="**/*.jsp"/>
      <ant:exclude name="**/includes/**/*.jsp"/>
    </ant:jspc>
```

Database Indexing

JIRA creates database indices automatically when the underlying table is created in the database. Hence, if you create a new installation of JIRA 3.0 or later, you do not need to create JIRA's database indices manually.

If you are upgrading JIRA from an earlier version (e.g. JIRA 2.6) and do not wish to create the indices manually, please follow the Migrating JIRA to Another Server instructions and recreate (drop and create) JIRA's database (or remove all tables in the database) **AFTER** successfully exporting your data and before doing the import into the new version of JIRA. Removing the database will force JIRA to recreate all tables in the database and hence create all required indices.

Please be aware: If you are upgrading from JIRA 2.6.1 or earlier to JIRA 3.0 (or above), JIRA will not create indices automatically, unless the database is removed and recreated.

If you do not wish to drop and recreate JIRA's database, you can add the indices manually by running the SQL statements shown below.

The syntax for creating indices differs between databases, so consult your documentation for the your database. In addition, if you change the database tables or fields that you use in `entitymodel.xml`, you will need to change the shown SQL statements.

Below is the SQL for creating indices on PostgreSQL (you will probably need to alter this for your database):

> *The information on this page does not apply to **JIRA OnDemand**.*

```
CREATE INDEX action_issue ON jiraaction (issueid, actiontype);

      CREATE INDEX chggroup_issue ON changegroup (issueid);

      CREATE INDEX chgitem_chggrp ON changeitem (groupid);

      CREATE INDEX cf_cfoption ON customfieldoption (CUSTOMFIELD);

      CREATE INDEX cfvalue_issue ON customfieldvalue (ISSUE, CUSTOMFIELD);

      CREATE INDEX attach_issue ON fileattachment (issueid);

      CREATE INDEX subscrpt_user ON filtersubscription (FILTER_I_D, USERNAME);
      CREATE INDEX subscrptn_group ON filtersubscription (FILTER_I_D, groupname);

      CREATE INDEX issue_key ON jiraissue (pkey);

      CREATE INDEX issuelink_src ON issuelink (SOURCE);
      CREATE INDEX issuelink_dest ON issuelink (DESTINATION);
      CREATE INDEX issuelink_type ON issuelink (LINKTYPE);

      CREATE INDEX linktypename ON issuelinktype (LINKNAME);
      CREATE INDEX linktypestyle ON issuelinktype (pstyle);

      CREATE INDEX node_source ON nodeassociation (SOURCE_NODE_ID, SOURCE_NODE_ENTITY);
      CREATE INDEX node_sink ON nodeassociation (SINK_NODE_ID, SINK_NODE_ENTITY);

      CREATE INDEX ntfctn_scheme ON notification (SCHEME);

      CREATE INDEX osgroup_name ON groupbase (groupname);

      CREATE INDEX mshipbase_user ON membershipbase (USER_NAME);
      CREATE INDEX mshipbase_group ON membershipbase (GROUP_NAME);

      CREATE INDEX osproperty_all ON propertyentry (ENTITY_NAME, ENTITY_ID);

      CREATE INDEX osuser_name ON userbase (username);

      CREATE INDEX sec_scheme ON schemeissuesecurities (SCHEME);

      CREATE INDEX sec_security ON schemeissuesecurities (SECURITY);

      CREATE INDEX prmssn_scheme ON schemepermissions (SCHEME);

      CREATE INDEX sr_author ON searchrequest (authorname);
      CREATE INDEX sr_group ON searchrequest (groupname);

      CREATE INDEX user_source ON userassociation (SOURCE_NAME);
      CREATE INDEX user_sink ON userassociation (SINK_NODE_ID, SINK_NODE_ENTITY);

      CREATE INDEX workflow_scheme ON workflowschemeentity (SCHEME);
```

Once you have created the index, you may need to tell your database to recompute its indices. For PostgreSQL, the command is `vacuumdb -U username -z -v database-name`

Consult your database documentation for your database specific command.

Logging and Profiling

Logging

JIRA uses a powerful logging module called log4j for runtime logging.

Log file location

The logs are written to the `log` subdirectory of your JIRA Home Directory (or elsewhere if you have configured a different location). You can view the location of the `atlassian-jira.log` in the **'File Paths'** section of the System Information page.

- Security-related information (e.g. login, logout, session creation/destruction, security denials) is written to `atlassian-jira-security.log`.

Changing the location of the log

In the `log4j.properties` file (located in the JIRA Installation Directory):

1. Change the following line:

```
log4j.appender.filelog=com.atlassian.jira.logging.JiraHomeAppender
```

...to this:

```
log4j.appender.filelog=org.apache.log4j.RollingFileAppender
```

2. Change the following line to point to the new location of the log file:

```
log4j.appender.filelog.File=atlassian-jira.log
```

On this page:

- Logging
- Profiling

*The information on this page does not apply to **JIRA OnDemand**.*

Logging levels

There are five logging levels available in log4j: `'DEBUG', 'INFO', 'WARN', 'ERROR'` and `'FATAL'`. Each logging level provides more logging information that the level before it:

- 'DEBUG'
- 'INFO'
- 'WARN'
- 'ERROR'
- 'FATAL'

'DEBUG' provides the most verbose logging and 'FATAL' provides the least verbose logging. The default level is WARN, meaning warnings and errors are displayed. Sometimes it is useful to adjust this level to see more detail.
Please be aware: the 'DEBUG' setting may cause user passwords to be logged.

The default logging levels can be changed either

- temporarily — your change to the logging level will not persist after you next restart JIRA, or
- permanently — your change to the logging level will persist, even after you restart JIRA.

For example, when troubleshooting, you might temporarily change the logging level from 'WARNING' to 'INFO' so as to get a more detailed error message or a stack trace. If you are unsure of which logging categories to adjust, the most helpful information generally comes from the `log4j.rootLogger` category and the `log4j`**`<category>`**`.com.atlassian` categories.

Temporarily changing the logging level

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Troubleshooting and Support' > 'Logging & Profiling'** (tab). The 'Logging' page will be displayed, which lists all defined log4j categories (as package names) and their current logging levels.
   Keyboard shortcut: **'g' + 'g' +** start typing **'logging & profiling'**
3. To change logging level of a category, click linked logging level associated with the relevant package name. To turn off logging of a category, click the '**OFF**' link associated with the relevant package name.

Permanently changing the logging level

1. Edit the `log4j.properties` file (located in the JIRA Installation Directory).
2. Locate the section:

```
log4j.logger.com.atlassian = WARN, console, filelog
log4j.additivity.com.atlassian = false
```

and make your desired changes (e.g. change the `WARN` to `DEBUG`).

The `log4j.properties` file that ships with JIRA has the default logging levels specified. For more information about log4j (e.g. how to define new logging categories), and about the format of the `log4j.properties` file, please refer to the documentation on the log4j site.

3. *(Only if you are running JIRA WAR) Reduild and redeploy the web application.*
4. Restart JIRA.

Please Note: If your application server configures logging itself, you may need to remove the `log4j.properties` file. You may also need to remove the entire `log4j.jar` file to get logging to work.

Profiling

If you are experiencing performance issues with JIRA, it is often helpful to see where the slow-downs occur. To do this you can enable profiling as described below, and then analyse the performance traces that JIRA will produce for every request. An example of a profiling trace is shown below:

```
[Filter: profiling] Turning filter on [jira_profile=on]
[116ms] - /secure/Dashboard.jspa
  [5ms] - IssueManager.execute()
    [5ms] - IssueManager.execute()
      [5ms] - Searching Issues
  [29ms] - IssueManager.execute()
    [29ms] - IssueManager.execute()
      [29ms] - Searching Issues
        [28ms] - Lucene Query
          [23ms] - Lucene Search
```

Profiling can be enabled either

- temporarily — profiling will be enabled until you next restart JIRA, or
- permanently — profiling will remain enabled, even after you restart JIRA.

Temporarily enabling profiling

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Troubleshooting and Support' > 'Logging & Profiling'** (tab). The 'Logging' page will be displayed, which lists all defined log4j categories (as package names) and their current logging levels.
   Keyboard shortcut: **'g' + 'g' +** start typing **'logging & profiling'**
3. Scroll to the **'Profiling'**section at the end of the page. This section will inform you whether profiling is currently turned 'ON' or 'OFF' and will provide you with 'Disable' or 'Enable' profiling links respectively.
   - To turn Profiling 'ON', click the '**Enable** profiling' link. JIRA will start generating profiling traces in its log.
   - To turn Profiling 'OFF', click the '**Disable** profiling' link.

Permanently enabling profiling

1. In your JIRA installation directory, edit either the `atlassian-jira/WEB-INF/web.xml` file (or if you are using the JIRA WAR distribution, the `webapp/WEB-INF/web.xml` file).
2. Find the following entry:

```
<filter>
            <filter-name>profiling</filter-name>

<filter-class>com.atlassian.jira.web.filters.JIRAProfilingFilter</filter-class>
            <init-param>
                <!-- specify the which HTTP parameter to use to turn the filter on or
off -->
                <!-- if not specified - defaults to "profile.filter" -->
                <param-name>activate.param</param-name>
                <param-value>jira_profile</param-value>
            </init-param>
            <init-param>
                <!-- specify the whether to start the filter automatically -->
                <!-- if not specified - defaults to "true" -->
                <param-name>autostart</param-name>
                <param-value>false</param-value>
            </init-param>
        </filter>
```

3. Modify the `autostart` parameter to be **true** instead of **false.**That is:

```
<init-param>
                <!-- specify the whether to start the filter automatically -->
                <!-- if not specified - defaults to "true" -->
                <param-name>autostart</param-name>
                <param-value>true</param-value>
            </init-param>
```

4. Save the file. Profiling will be enabled when you restart JIRA.
   If you are running JIRA WAR, re-build and re-deploy the JIRA web application using the `build` script and the instructions for your application server (i.e. Apache Tomcat).

Logging email protocol details

To assist in resolving email issues, it can be useful to know exactly what is passing over the wire between JIRA and SMTP, POP or IMAP servers. This page describes how to enable protocol-level logging.

> *The information on this page does not apply to **JIRA OnDemand**.*

To do this

Set **-Dmail.debug=true** and restart JIRA.

Output

In the logs, you should then see JavaMail initialize the first time a mail operation is run:

```
DEBUG: JavaMail version 1.3.2
DEBUG: java.io.FileNotFoundException: /usr/local/jdk1.6.0/jre/lib/javamail.providers (No such
file or directory)
DEBUG: !anyLoaded
DEBUG: not loading resource: /META-INF/javamail.providers
DEBUG: successfully loaded resource: /META-INF/javamail.default.providers
DEBUG: Tables of loaded providers
DEBUG: Providers Listed By Class Name:
{com.sun.mail.smtp.SMTPSSLTransport=javax.mail.Provider[TRANSPORT,smtps,com.sun.mail.smtp.SMTPSSLTra
Microsystems, Inc],
com.sun.mail.smtp.SMTPTransport=javax.mail.Provider[TRANSPORT,smtp,com.sun.mail.smtp.SMTPTransport,S
Microsystems, Inc],
com.sun.mail.imap.IMAPSSLStore=javax.mail.Provider[STORE,imaps,com.sun.mail.imap.IMAPSSLStore,Sun
Microsystems, Inc],
com.sun.mail.pop3.POP3SSLStore=javax.mail.Provider[STORE,pop3s,com.sun.mail.pop3.POP3SSLStore,Sun
Microsystems, Inc],
com.sun.mail.imap.IMAPStore=javax.mail.Provider[STORE,imap,com.sun.mail.imap.IMAPStore,Sun
Microsystems, Inc],
com.sun.mail.pop3.POP3Store=javax.mail.Provider[STORE,pop3,com.sun.mail.pop3.POP3Store,Sun
Microsystems, Inc]}
DEBUG: Providers Listed By Protocol:
{imaps=javax.mail.Provider[STORE,imaps,com.sun.mail.imap.IMAPSSLStore,Sun Microsystems, Inc],
imap=javax.mail.Provider[STORE,imap,com.sun.mail.imap.IMAPStore,Sun Microsystems, Inc],
smtps=javax.mail.Provider[TRANSPORT,smtps,com.sun.mail.smtp.SMTPSSLTransport,Sun Microsystems,
Inc], pop3=javax.mail.Provider[STORE,pop3,com.sun.mail.pop3.POP3Store,Sun Microsystems, Inc],
pop3s=javax.mail.Provider[STORE,pop3s,com.sun.mail.pop3.POP3SSLStore,Sun Microsystems, Inc],
smtp=javax.mail.Provider[TRANSPORT,smtp,com.sun.mail.smtp.SMTPTransport,Sun Microsystems, Inc]}
DEBUG: successfully loaded resource: /META-INF/javamail.default.address.map
DEBUG: !anyLoaded
DEBUG: not loading resource: /META-INF/javamail.address.map
DEBUG: java.io.FileNotFoundException: /usr/local/jdk1.6.0/jre/lib/javamail.address.map (No such
file or directory)
DEBUG: getProvider() returning javax.mail.Provider[STORE,pop3,com.sun.mail.pop3.POP3Store,Sun
Microsystems, Inc]
DEBUG POP3: connecting to host "localhost", port 110, isSSL false
S: +OK Dovecot ready.
C: USER pop-test
S: +OK
C: PASS pop-test
[Filter: profiling] Using parameter [jira_profile]
[Filter: profiling] defaulting to off [autostart=false]
[Filter: profiling] Turning filter off [jira_profile=off]
S: +OK Logged in.
C: STAT
S: +OK 2 1339
C: NOOP
S: +OK
C: TOP 1 0
S: +OK
Return-path: <pop-test@atlassian.com>
Envelope-to: pop-test@localhost
Delivery-date: Wed, 28 Feb 2007 16:28:26 +1100
Received: from pop-test by teacup.atlassian.com with local (Exim 4.63)
        (envelope-from <pop-test@atlassian.com>)
        id 1HMHMY-0007gB-8O
        for pop-test@localhost; Wed, 28 Feb 2007 16:28:26 +1100
Date: Wed, 28 Feb 2007 16:28:26 +1100
From: Jeff Turner <jeff@atlassian.com>
To: pop-test@localhost
Subject: Testing to me - Wed Feb 28 16:28:23 EST 2007
Message-ID: <20070228052826.GA29514@atlassian.com>
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Disposition: inline
User-Agent: Mutt/1.5.13 (2006-08-11)
Lines: 0
```

Related pages

No content found for label(s) log, logging.

Restoring Data

This page describes how to restore JIRA data from a JIRA XML backup. This process is typically conducted towards the end of Migrating JIRA to Another Server or splitting your JIRA instance across multiple servers.

If you wish restore a single project from your backup into an existing JIRA instance, refer to these instructions on restoring a project from backup instead.

When restoring data, **all data in the existing JIRA database is deleted,** including all user accounts. Before you begin, make sure you have the password to a login in the backup file that has the 'JIRA System Administrator' global permission.

Restoring JIRA from backup is a three stage process:

1. *(Optional)* Disable email sending/receiving
2. Restore data from XML to the database
3. *(Optional)* Restore the attachments to the attachments directory (if attachments were backed up)

On this page:

- 1. Disabling email sending/receiving
- 2. Restoring XML data
- 3. Restoring attachments

*The information on this page does not apply to **JIRA OnDemand**.*

1. Disabling email sending/receiving

If you are restoring production data into a *test* JIRA instance for experimentation purposes, you probably want to disable JIRA's email interaction features before you begin:

- **Disable email notifications** — if JIRA is configured to send emails about changes to issues, and you want to make test modifications to the copy, you should start JIRA with the `-Datlassian.mail.senddisabled=true` flag.
- **Disable POP/IMAP email polling** — if JIRA is configured to poll a mailbox (to create issues from mails), you will likely want to disable this polling on your test installation. This is done by setting the `-Datlassian.mail.fetchdisabled=true` flag.

Exactly how to set these flags is dependent on your particular application server, but for JIRA, this is done by setting the `JAVA_OPTS` environment variable before starting JIRA:

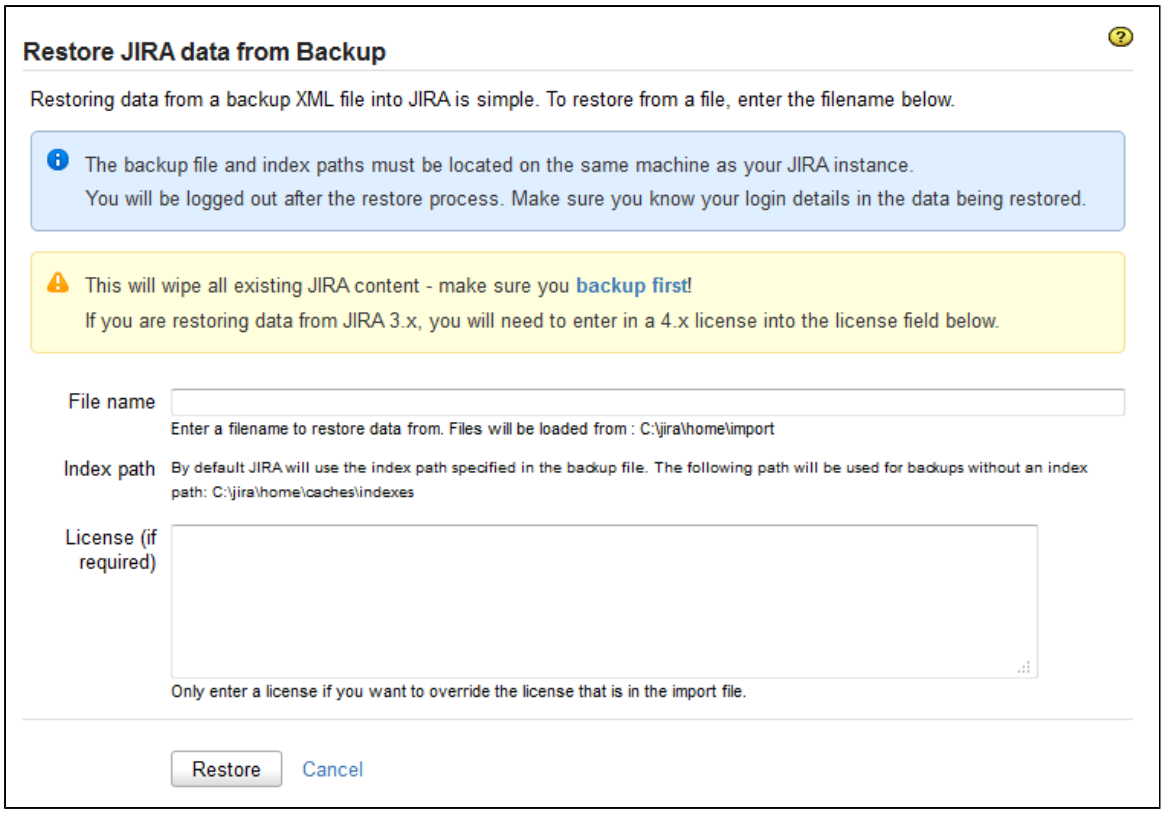
```
set JAVA_OPTS="-Datlassian.mail.senddisabled=true -Datlassian.mail.fetchdisabled=true"
cd bin
startup.bat
```

You could also try un-commenting the `JAVA_OPTS="-Datlassian.mail.senddisabled=true -Datlassian.mail.fetchdisabled=true` line from your `/bin/setenv.sh` file and then running startup.

2. Restoring XML data

Note: these instructions assume that you are restoring an XML backup. If you used native database tools to create your backup, the restore process will be tool-specific so these instructions do not apply to you.

1. Log in as a user with the '**JIRA System Administrators**' global permission.
2. Select **'Administration' > 'System' > 'Import & Export' > 'Restore System'** (tab) to open the 'Restore JIRA data from Backup' page.
   Keyboard shortcut: **'g' + 'g' +** type **'rest'**



3. In the **'File name'** field, type the file name of the zipped XML backup file generated by JIRA.
   Ensure that this backup file has been moved or copied to the location specified below this field.
4. The **Index Path** field indicates where JIRA will restore the search index data from the zipped XML backup file. This location (which cannot be modified) matches the index path specified in the zipped XML backup file. If, however, this backup file does not specify an index path, JIRA will restore the search index to the `caches/indexes` subdirectory of the JIRA Home Directory.
   Please Note:
   - The contents of the index directory may be deleted by the restore process.
   - The index directory should *only* contain JIRA index data.
5. Click the '**Restore**' button and wait while your JIRA data is restored.
   Once the data has been restored, JIRA will inform you that you have been logged out. This happens because all JIRA users which existed in JIRA prior to JIRA's data being restored will have been deleted and replaced by users stored in the JIRA export file.

It is recommended that you avoid passing through a proxy when performing an XML restore, especially if your JIRA instance is very large. Using a proxy may cause timeout errors.

3. Restoring attachments

If you created a backup of the attachments directory, you will need to restore the backup into a directory where JIRA can access it.
If you use a custom directory for storing your attachments, ensure that JIRA has read and write permissions to this directory and its subdirectories.

The process of restoring the attachments backup depends on the way it was created. Usually you can use the same tool to restore the backup as the one that was used to create it (see *Backing up attachments*).

If you are restoring the attachments into a different location (i.e. a different directory path) from where they were previously located (e.g. this will be the case when moving servers), please follow the instructions provided in *Configuring attachments* to change the location of the attachments directory so that JIRA can find the restored attachments.

Restoring a Project from Backup

JIRA's Project Import tool allows you to restore a single project from a backup file into your JIRA instance. This feature is particularly useful if you do not wish to overwrite the existing projects or configuration of your JIRA instance by importing the entire backup. Your backup file mus have been created using JIRA's backup tool. You cannot import a project from a backup using your native database tools.

Please note, if you wish to restore a project from a backup file into a **new empty JIRA instance,** we highly recommend that you **do not use the Project Import tool.** Restoring the entire backup file into the new instance and then deleting unwanted projects is much simpler in this scenario, as you will retain the configuration settings from your backup. Instructions on moving a project to a new instance are available on the splitting a JIRA instance page. Projects can be deleted via the 'Projects' page in JIRA, which is accessed from the '*Administration' menu

On this page:

- Before you begin
  - Project Import Restrictions
- Restoring your project
  - Preparing your target JIRA instance
  - Project Import
  - Need Help?

*The information on this page does not apply to **JIRA OnDemand**.*

Before you begin

Restoring a project from a backup is not a trivial task. You may be required to change the configuration of your target JIRA instance to accommodate the project import. Additionally, the Project Import data mapping can be resource intensive on your hardware and may take a long time to complete, if you are importing a large project. Note, the Project Import tool will lock out your instance of JIRA during the actual data import (not during the validations), so please ensure that your instance does not need to be accessible during this time.

We strongly recommend that you perform a full backup of your target JIRA instance before attempting to restore a project into it.

Project Import Restrictions

The Project Import tool will only import a project between identical instances of JIRA. That is;

- The **version** of JIRA in which your backup was created must be identical to the version of your target JIRA instance, e.g. if your backup file was created in JIRA 4.0, then your target instance of JIRA must be version 4.0.
- If your instance of JIRA had a **custom fields plugin** (e.g. JIRA Toolkit) installed when the backup file was created and the custom field was used in your project, then your target instance of JIRA must have the same version of the plugin installed for the Project Import tool to automatically work.

If any of these restrictions apply and you still wish to restore your project from backup, you will need to create a compatible backup file before importing your project by following the appropriate instructions below.

JIRA versions do not match

- If your backup file was created in an earlier version of JIRA than your target instance of JIRA:
  1. Set up a test JIRA instance, which is the same version as your target instance of JIRA. Make sure that the test JIRA instance uses a separate database and index from your target JIRA instance.
  2. Import the backup file into a test JIRA instance. (This will completely overwrite the test instance.)
  3. Create a new backup file from your test JIRA instance. You can now use this backup to import a specific project into your target production instance.
- If your backup file is from a later version of JIRA than your target instance of JIRA:
  1. Upgrade the version of your target instance of JIRA to match the version of JIRA in which the backup was created.

Custom fields plugin versions do not match

- If the custom fields plugin from your backup is an earlier version than the custom fields plugin in your target instance of JIRA:
  1. Import the backup file into a test JIRA instance. Make sure that the test JIRA instance uses a separate database and index from your target JIRA instance, as the import will overwrite all data in the database.
  2. In your test JIRA instance, upgrade your version of your custom fields plugin to match the version of the plugin in your target instance of JIRA.
  3. Create a new backup file from your test JIRA instance.
- If the custom fields plugin from your backup is a later version than the custom fields plugin in your target instance of JIRA:
  1. Upgrade the custom fields plugin version of your target instance of JIRA to match the version of JIRA in which the backup was created.

Restoring your project

The Project Import tool will attempt to map the data in your backup file into your target JIRA instance. If the project you are restoring does not exist in your target JIRA instance, it will create and populate the project with data from your backup. If the project already exists and is empty, it will attempt to populate the data from your backup into the project.

Why should I create an empty project in my target JIRA instance?

It is important to note that the primary task of the Project Import tool is to restore the data from your backup project into your target JIRA instance. While the Project Import tool can create a project if one does not exist in your target JIRA instance, it does not recreate any configuration settings that affect the data (e.g. screen schemes). If you wish to retain any configuration settings from your original project, we recommend that you create an empty project in your target instance with the necessary configuration settings before importing the data from your backup project.

You may wish to carry out the following setup tasks to ensure that your target JIRA instance is prepared to receive a project import beforehand. This can improve the time taken to validate the data mappings to your target JIRA instance.

If you are confident that your JIRA instance is set up appropriately, you can skip straight to the Project Import tool instructions. If there are any problems mapping the data from your backup file to your target JIRA instance, the Project Import tool will present validation errors for you to address.

Preparing your target JIRA instance

The Project Import tool does not automatically add missing project entities (e.g. user groups, issue priorities, custom field types) or fix incorrect associations (e.g. issue types in workflow schemes), so some manual work is required to set up your target JIRA instance so that your project can be restored. If the Project Import wizard cannot find a valid target location for any of the backup project data, it will not be able to restore the project. The instructions below describe the setup activities that address the most common data mapping problems that occur when restoring a project from a backup.

We recommend that you perform as much of the configuration of your target JIRA instance as possible, prior to starting the project import. However, if you do not have the information available to complete these setup activities beforehand, the Project Import wizard will inform you of any problems that need your attention. Alternatively, you can import the backup file into a test JIRA instance to check the configuration.

1. Setting up the project

If you have a project in your target JIRA instance that you wish to restore data into, you will need to ensure that the project is empty, i.e.

- no issues — read the Quick Search page to find out how to find all issues in a project
- no components — read the Component Management page to find out how to view a summary of a project's components
- no versions — read the Version Management page to find out how to view a summary of a project's versions

2. Setting up users and groups

The following types of users are considered mandatory for a project to be imported:

- reporter, assignee, component lead or project lead.

The following users are considered to be optional for a project to be imported:

- comment author/editor, work log author/editor, a user in a custom field (user picker), voter, watcher, change group author (i.e. someone who has changed an issue), attachment author, user in a project role.

The Project Import will attempt to create missing users if they are associated with the project. However, if the Project Import tool cannot create missing mandatory users in your target JIRA instance, then you will not be permitted to import the project. This may occur if you have External User Management enabled in your target JIRA instance — you will need to disable External User Management or create the missing users manually in your external user repository before commencing the import.

Please note that if you do not have enough information about the users in your backup file, the Project Import wizard will provide a link to a table of the missing users on a new page as well as a link to an XML file containing the missing users (on the new page). The table of users will display a maximum of 100 users, but the XML file will always be available.

3. Setting up custom fields

As described previously, the versions of your custom field plugins must match between your backup and your target instance of JIRA for your project to be imported. You need to ensure that you have set up your custom fields correctly in your target JIRA instance, as follows:

- **Custom Field Type** — If you do not have a particular custom field type (e.g. cascading select) installed on your target JIRA, then all custom field data in your backup project that uses that custom field type will not be restored. However, your project can still be restored.
  For example, say you have a custom field, 'Title', which is a 'Cascading Select' field type and was used in your backup project (i.e. there is saved data for this field). If you do not have the 'Cascading Select' custom field type installed on your target JIRA, then all data for custom field 'Title' (and all other cascading select custom fields) will not be restored.
- **Custom Field Configuration** — If you do have a particular custom field type (e.g. multi select) installed on your target JIRA, then you must configure all of the custom fields (of that custom type) in your target JIRA to match the equivalent custom fields in your

backup project. Additionally, if your custom field has selectable options, then any options used (i.e. there is saved data for these options) in your backup project must exist as options for the custom field in your target JIRA.
For example, say you have a custom multi select field named, 'Preferred Contact Method', in your backup project with options, 'Phone', 'Email', 'Fax'. Only the 'Phone' and 'Email' were actually used in your backup project. In this scenario, you need to set up your target JIRA instance as follows:

- There must be a field named, 'Preferred Contact Method', in your target JIRA instance.
- 'Preferred Contact Method' must be a multi select custom field type.
- 'Preferred Contact Method' must have the options, 'Phone' and 'Email' at a minimum, since they were used in your backup project. Please note, 'Preferred Contact Method' in your target JIRA could also have additional options like 'Fax', 'Post', 'Mobile', etc, if you choose.
  If you have not configured your existing custom field correctly, you will not be permitted to import your backup project until you correct the configuration errors in your target JIRA.
  You may wish to refer to the custom fields documentation for more information on the custom field types and custom field configuration.

- **Compatibility with the Project Import tool** — Custom fields also need to be compatible with the Project Import tool for the custom field data to be imported. Custom fields created prior to JIRA v4.0 cannot be imported by the Project Import tool. The custom field developer will need to make additional code changes to allow the Project Import tool to restore the custom field data. If any of the custom fields used in your backup file are not compatible with the Project Import tool, the Project Import wizard will warn you and the related custom field data will not be imported. All the target JIRA system custom fields and the custom fields included in JIRA plugins supported by Atlassian (e.g. JIRA Toolkit, Charting Plugin, Labels Plugin, Perforce Plugin) are compatible with the Project Import tool.

4. Setting up workflows, system fields, groups and roles

In addition to custom fields, you need to correctly configure the project workflow, issue attributes (e.g. issue types) and groups/roles in your target JIRA instance for your project to be restored successfully. Please ensure that you have reviewed the constraints on each of the following:

Workflows and Workflow Schemes:

- The project import process does not import workflows or workflow schemes. If you wish to retain a customised workflow from your backup, you will need to create a new workflow in your target JIRA instance and manually edit the new workflow (e.g. create steps and transitions) to reflect your old workflow (note, the default JIRA workflow is not editable). You will then have to add this workflow to a workflow scheme to activate it.
  Read more about creating and editing workflows in the JIRA Workflow and Activating Workflows documents. Please note that you may be required to create and edit a new workflow and workflow scheme to satisfy constraints on workflow entities from your backup, as described in the sections below, even if you do not wish to recreate the exact same workflow.

Do not use the JIRA functionality for exporting and importing workflow XML definitions, to copy your backup workflow to your target JIRA instance. The workflow import/export tools do not include workflow screens in the process. Hence, you will be required to manually edit the workflow definitions post-import to match up new screens to the workflow, which is more work than it is worth.

Issue Types:

- If an issue type has been used in your backup project (i.e. there are issues of this issue type), you must set up the same issue type in your target JIRA project. You may want to consider setting up Issue Types for the project instead of globally.
- Workflow schemes — If you have associated an issue type with a particular workflow scheme in your backup project, you must ensure that the same association exists in your target JIRA. See the above section on **'Workflow and Workflow Schemes'** for further information on how to set up a workflow in your target JIRA instance.
- Custom field configuration schemes — custom field configuration schemes can be used to apply a custom field configuration to specific issue types. If you have configured a custom field differently for different issue types in your backup project, you may wish to set up a custom field configuration scheme to apply the same custom field configuration to the same issue types in your target JIRA instance. This will help ensure that you do not have a custom field for an issue type that is configured incorrectly (e.g. missing an option, if it has multiple selectable options), as described in the 'Setting up custom fields' section above.

Statuses:

- If an issue status has been used in your backup project (i.e. there are issues with the status), you must set up the same status in your target JIRA project.
- Workflow schemes — If you have linked a status into a particular workflow scheme in your backup project, you must ensure that the same association exists in your target JIRA. See the above section on 'Workflow and Workflow Schemes' for further information on how to set up a workflow in your target JIRA instance.

Make sure to match the **Linked Status** name, not the **Step Name**, when inspecting your workflow.

Security Levels:

- If an issue security level has been used in your backup project (i.e. there are issues with this security level), it must be set up in your target instance of JIRA. If you did not create an existing empty project, we recommend that you do so and set up the appropriate security levels for the project (via an issue security scheme).
- Issue Security schemes — Not applicable. It does not matter which users, groups or project roles are assigned to which security levels, as long as the appropriate security levels exist (please see the constraints on security levels in the 'Setting up entities and

types' section).

Priority:

- If an issue priority has been used in your backup project (i.e. there are issues with this priority), it must be set up in your target instance of JIRA.

Resolution:

- If an issue resolution has been used in your backup project (i.e. there are issues with this resolution), it must be set up in your target instance of JIRA.

Issue Link Type:

- If an issue link type has been used in your backup project (i.e. there are issues associated by this link type), it must be set up in your target instance of JIRA.

Project Role:

- If a project role has been used in your backup project (i.e. there are users/groups assigned to this project role), it must be set up in your target instance of JIRA.
  (Note: The Project Import tool will copy across the project role membership from your backup project to your target JIRA instance, if you choose. See the Project Import section for further details).

Group:

- If a user group has been used in your backup project (i.e. there are users in this group), it must be set up in your target instance of JIRA.

A note about schemes
The project import process does not directly affect schemes, although entities and types associated with schemes may be affected as described above. Please note that the following schemes are not affected at all by the project import:

- Permission schemes — Not applicable. Permissions schemes do not need to match between the backup and target instance of JIRA.
- Notification schemes — Not applicable. Notification schemes do not need to match between the backup and target instance of JIRA.
- Screen schemes — Not applicable. Screen schemes do not need to match between the backup and target instance of JIRA.
- Issue type screen schemes — Not applicable. Issue type screen schemes do not need to match between the backup and target instance of JIRA.
- Field Configuration schemes — Not applicable. Please note that if a field was configured as optional in your backup project and is configured as a required field in your target JIRA instance, then the project will still be imported even if the field is empty. However, this field will be enforced as mandatory the next time a user edits an issue containing the field.

5. Setting up links

The Project Import tool will automatically create all issue links between issues within your backed up project. It will also try to create links between the backup project and another project, as long as the other project already exists in your target JIRA instance with the relevant issue keys. If the source/target of a link cannot be found (i.e. the entire project or the particular issue may be missing), the link will not be created although the project will still be imported.

Note that the Project Import tool will create issue links between projects in either direction (source to target, or target to source). This means that if you import two projects from the same backup file, the second project import will create all of the links between the two projects that were missing from the first project import.

Once you have completed as many of the setup tasks as you are able to, run the Project Import tool.

Project Import

Restoring your project is a four step process:

1. Specify the backup file
2. Select a project
3. Review data mapping validations
4. Verify the restored project

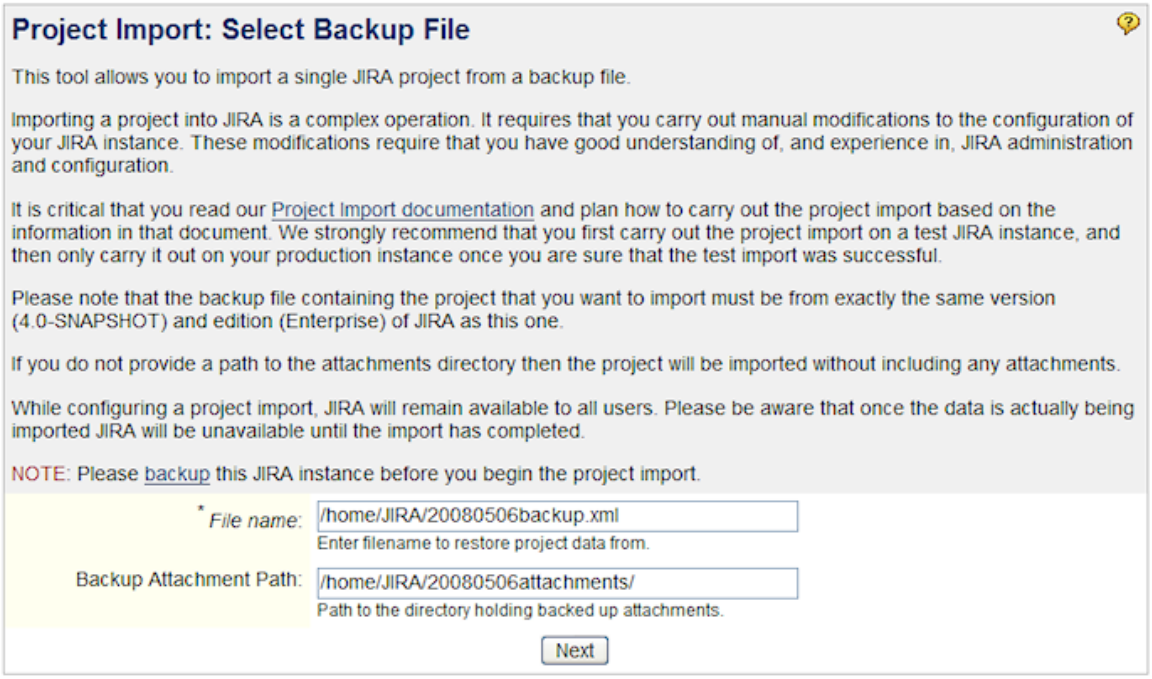
If you start the Project Import tool, we strongly recommend that you complete all steps of the wizard before performing any other activities in JIRA. Please be aware that it can take some time to validate the data mappings and then import the project.

You will most likely need to navigate away from the Project Import wizard to correct your JIRA configuration, as advised by validation errors in the wizard. If you have to navigate to other pages in JIRA to correct your JIRA configuration or for other activities, you should:

- ***(recommended)*** open a separate session of JIRA in a new browser window/tab. When you return to the Project Import wizard in the

original browser window/tab, you can use the **'Refresh validations'** button on the validation screen to re-validate the data mappings; or,
- wait until the progress bar completes for the step you are currently in, before navigating elsewhere in JIRA. The state of the Project Import wizard will be saved until you log out of JIRA, your user session expires or you commence a different project import. You can resume your project import by returning to the Project Import page (via the main Administration menu) and selecting the 'resume' link on the first page of the wizard.

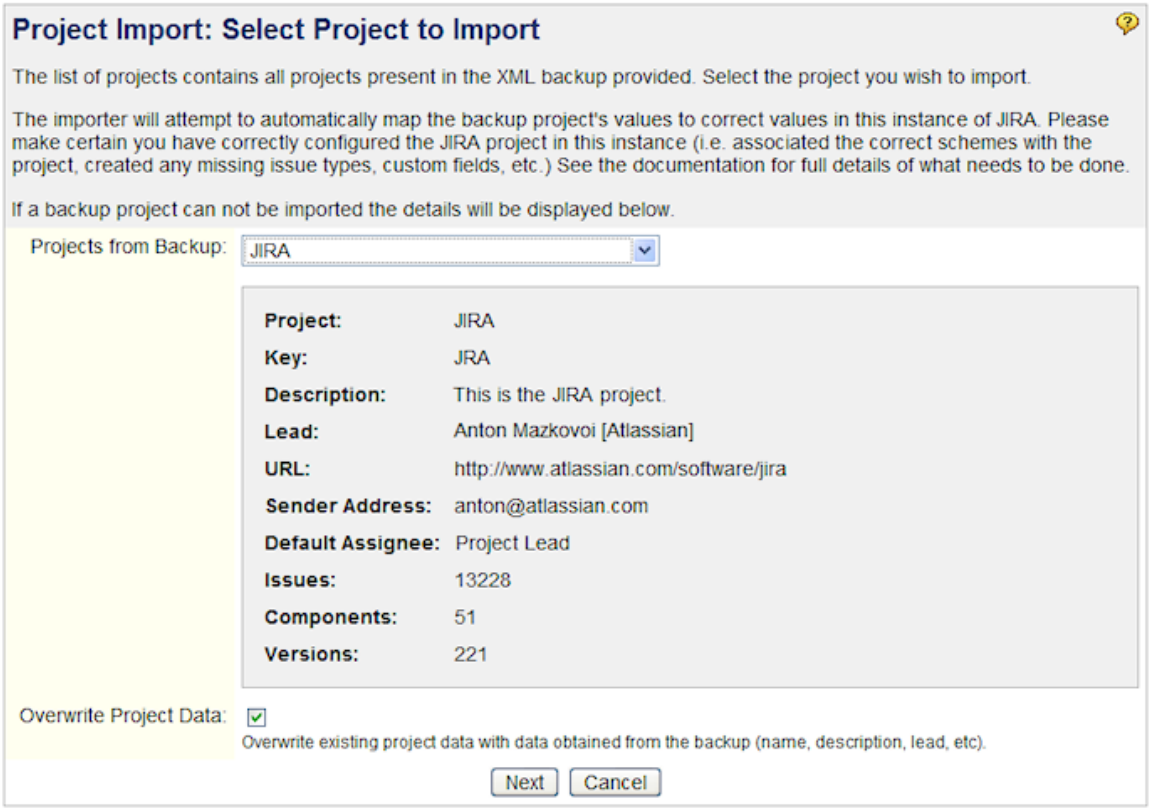
1. Specify the backup file



To start the Project Import tool,

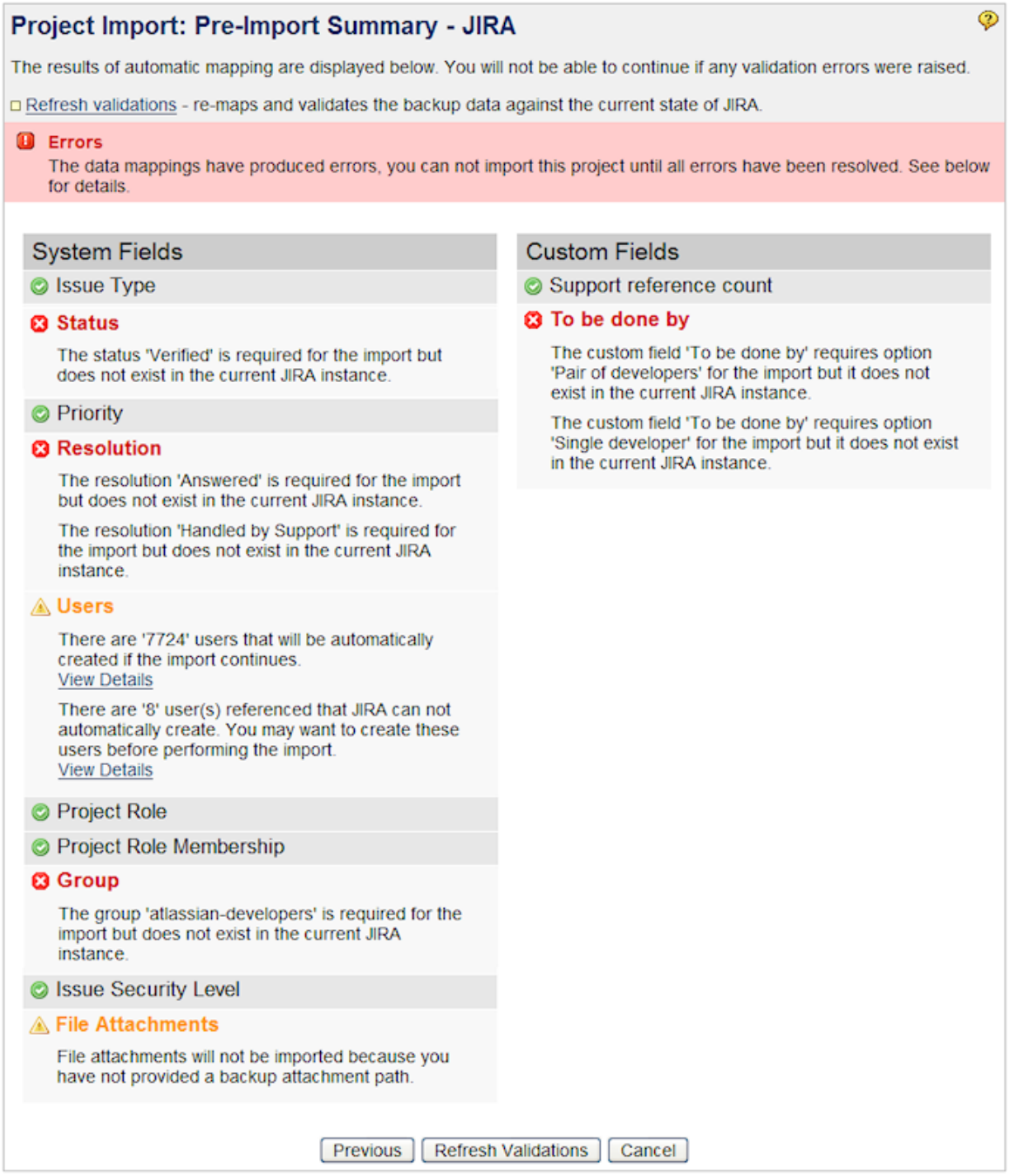
1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Bring up the administration page by clicking either the **'Administration'** link on the top bar or the title of the Administration box on the dashboard.
3. Click the **'Project Import'** link in the left hand menu. The first step of the Project Import wizard will display, ' Project Import: Select Backup File'.
4. Specify the path and name of your backup file in the 'File name' field. Your backup file must be an XML or ZIP file (as exported by JIRA).
5. Specify the path where you have backed up the attachments (add anchor to backup attachments section) for your project in the **'Backup Attachment Path'** field. Do not specify the attachment path for your target instance of JIRA as the backup attachment path, as the Project Import tool will overwrite attachments in that directory. Please also ensure that you have enabled file attachments in your target JIRA instance. You will not be allowed to proceed with the import if you have specified a backup attachment path and do not enable file attachments in your target JIRA instance.
   Note: You can choose to not specify a backup attachment path. If so, you will be able to restore your project from backup, however it will have no attachments associated with it. Please note, you cannot restore your attachments separately if you do not restore them as part of the project import, as the database entries for the attachments will be missing.

2. Select a project to restore



1. Select a project to restore from the **'Projects from Backup'** dropdown. This dropdown will list all of the projects contained in your backup file.
2. If you have a valid project to restore from your backup, and your target JIRA instance has an existing empty project, then the **'Overwrite Project Details'** option will display. Select the **'Overwrite Project Details'** option if you want to overwrite the project details of the existing empty project with the project details from your backup. The project details are the Name, URL, Project Lead, Default Assignee and Description of the project, as well as any project role members set up on your project. If there is no existing empty project in your target instance of JIRA, this option will be checked and disabled as the Project Import will create the project with project details from your backup file.

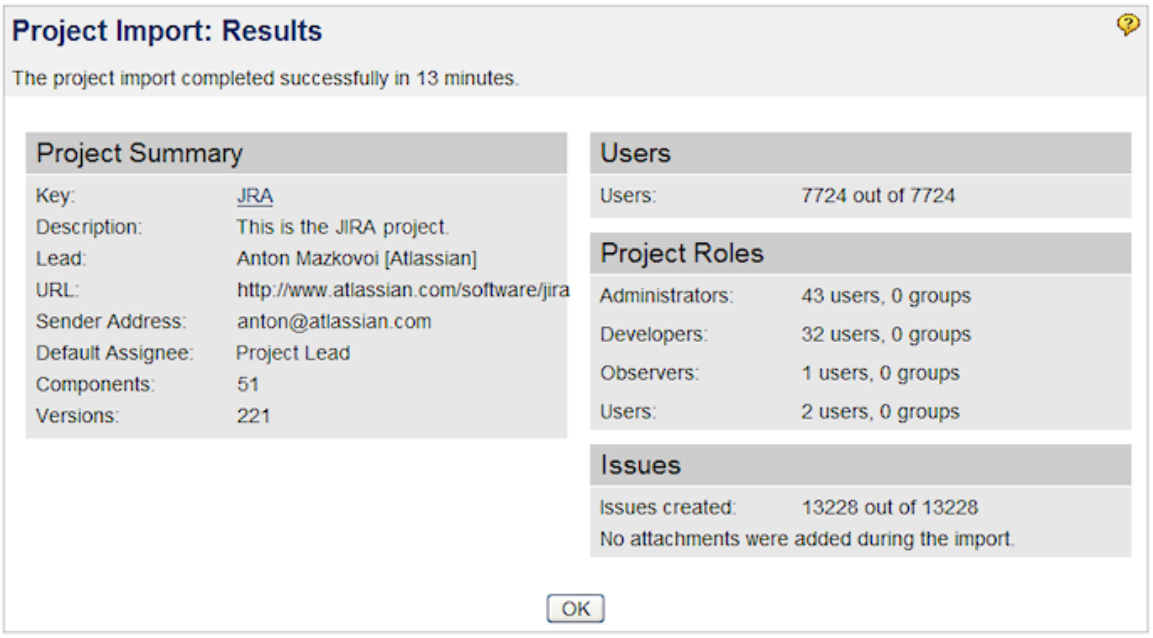
3. Review data mapping validations



1. The Project Import wizard will attempt to validate the data mappings required to import your project from the backup file. You can review the validations at this step of the wizard and modify your target JIRA instance as required.
    - A tick symbol ( ) means that there are no problems with mapping these entities.
    - An exclamation mark symbol ( ) means that there are problems with the data mapping that you should review before importing the project, but the project can still be imported. For example, a missing optional user that cannot be created automatically by the Project Import tool.
    - A cross symbol ( ) means that there are problems with the data mapping that must be fixed before you can import the project. For example, an Issue Type that is used in the backed up project is missing in your target JIRA instance.
2. The 'Preparing your target JIRA instance' section on this page lists the common data mapping errors.
3. Once you have resolved the data validation errors as required, click **'Import'** to commence the import of data from your backup file.

> The Project Import tool will lock out your instance of JIRA during the actual data import (not during the validations), so please ensure that your instance does not need to be accessible during this time.

4. Verify the restored project



1. Once the Project Tool has finished running, click **'OK'** to navigate to the restored project. You should verify that the issues, components and versions have been restored correctly. You should also check that any custom field data and links have been restored correctly.
2. Check that your attachments were correctly restored from your attachments backup directory.

> The Project Import tool will add an entry to every imported issue's Change History, showing when the issue was imported. Note that old entries in the Change History, from before the import, are retained for historical purposes only. Old entries may contain inconsistent data, since the configuration of the old and new JIRA systems may be different.

What if something went wrong?

- If your project import **did not complete,** you can refer to the JIRA log file. The Project Import tool will log details of the operation to this file, including any unexpected errors and exceptions. e.g. database locked out, disk full... etc.
- If your project import completed but **did not restore your project as expected,** you may wish to attempt to fix the problem manually in your target JIRA instance. You may also wish to try deleting the project in your target JIRA instance and re-importing it from backup, paying special note to any warning validations (e.g. users that will not be added automatically).

If you cannot resolve the problem yourself, you can contact us for assistance. Please see the **'Need help'** section below for details.

Need Help?

Need further help? You can raise a support request in the JIRA project at https://support.atlassian.com for assistance from our support team. Please attach to the support case:

- the backup file you are trying to import projects from, and
- the following information from your target JIRA instance:
  - your log file
  - an XML backup of your target JIRA instance
  - a copy and paste of the **entire contents** of the **System Info** page (accessed via the **Administration** tab), so that we know the details of your JIRA configuration.

You can anonymise the XML backups, if your data contains sensitive information.

Optimising Performance

For more information about optimising performance in JIRA, please refer to our Crashes and Performance Issues Troubleshooting knowledge base article.

> *The information on this page does not apply to **JIRA OnDemand**.*

Backing Up Data

This page describes how to back up your JIRA data, and establish processes for maintaining continual backups. Backing up your JIRA data is the first step in upgrading your server to a new JIRA revision, or splitting your JIRA instance across multiple servers. See also *Restoring JIRA data* and *Restoring a Project from Backup*.

Creating a complete backup of JIRA consists of two stages:

- 1. Backing up database contents
  - Using native database backup tools
  - Using JIRA's XML backup utility
- 2. Backing up the `data` directory

> *The information on this page does not apply to **JIRA OnDemand**.*

1. Backing up database contents

There are two possibilities: native database backup tools, or JIRA's XML backup utility.

> For production use, it is **strongly recommended** that for regular backups, you use native database backup tools instead of JIRA's XML backup service.
>
> When JIRA is in use, XML backups are not guaranteed to be consistent as the database may be updated during the backup process. JIRA does not report any warnings or error messages when an XML backup is generated with inconsistencies and such XML backups will fail during the restore process. Native database backup tools offer a much more consistent and reliable means of storing (and restoring) data.

Using native database backup tools

All serious databases come with tools to back up and restore databases (the 'MS' in RDBMS). We strongly recommend these tools in preference to the XML backup option described below, as they:

- ensure integrity of the database by taking the backup at a single point in time
- are much faster and less resource-intensive than JIRA's XML backup.
- integrate with existing backup strategies (e.g. allowing one backup run for all database-using apps).
- may allow for incremental (as opposed to 'full') backups, saving disk space.
- avoid character encoding and format issues relating to JIRA's use of XML as a backup format.

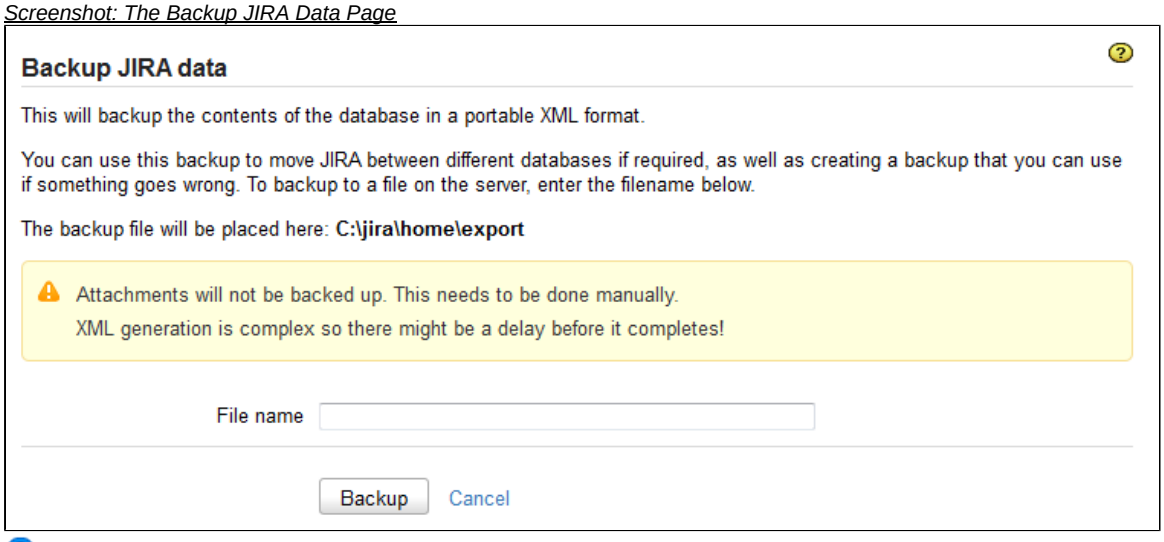
See the documentation for your database on how to set up periodic backups. This typically involves a cron job or Windows scheduled task invoking a command-line tool like mysqldump or pg_dump .

Using JIRA's XML backup utility

To perform a once-off backup, e.g. before an upgrade, follow the steps below.

You can also configure scheduled XML backups, as described in *Automating JIRA Backups*.

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Import & Export' > 'Backup System'** (tab) to open the 'Backup JIRA data' page.
   Keyboard shortcut: **'g' + 'g' +** type **'backu'**
   Screenshot: The Backup JIRA Data Page



   As shown in the screenshot above, the backup will be stored within the `export` subdirectory of the JIRA Home Directory.
3. In **'File name'** field, type the name of the backup file.
   Ensure that JIRA has the necessary file system permissions to write to this location. See the relevant procedures in the JIRA Installation and Upgrade Guide for details on creating a dedicated operating system account to run JIRA.

4. Click the **'Backup'** button and wait while your JIRA data is backed up.
   JIRA will save your XML backup as a zipped archive file.
5. When the backup is complete, a message will be displayed, confirming that JIRA has written its data to the file you specified.

2. Backing up the `data` directory

The `data` directory is a sub-directory of your JIRA Home Directory. It contains application data for JIRA, e.g. if you have attachments enabled, all files attached to JIRA issues are stored in the `data\attachments` directory (not in the database).

To back up the `data` directory, you need to create a snapshot of the `data` directory (including all files and subdirectories), then back up the snapshot. Note that the directory structure under the `data` directory **must** be preserved in the snapshot.

Creating this snapshot is an operating system-specific task, e.g.:

- On MS Windows, a batch script copying the directory can be written and scheduled periodically (Programs > Accessories > System Tools > Scheduled Tasks).
- On Linux/Solaris, it is best to write a small shell script, placed in `/etc/cron.daily` , backing up files to a directory like `/var/backup/jira` . It is best to copy an existing script in `/etc/cron.daily` to ensure local conventions (file locations, lockfiles, permissions) are adhered to.

Your "attachments" directory may be located elsewhere

If you have put your `attachments` directory in a custom location (see Configuring File Attachments) rather than inside the `data` directory, you will also need to back up your `attachments` directory using the snapshot method described above.

Automating JIRA Backups

JIRA can be configured to automatically create an XML backup of JIRA data on a routine basis.

Please Note:

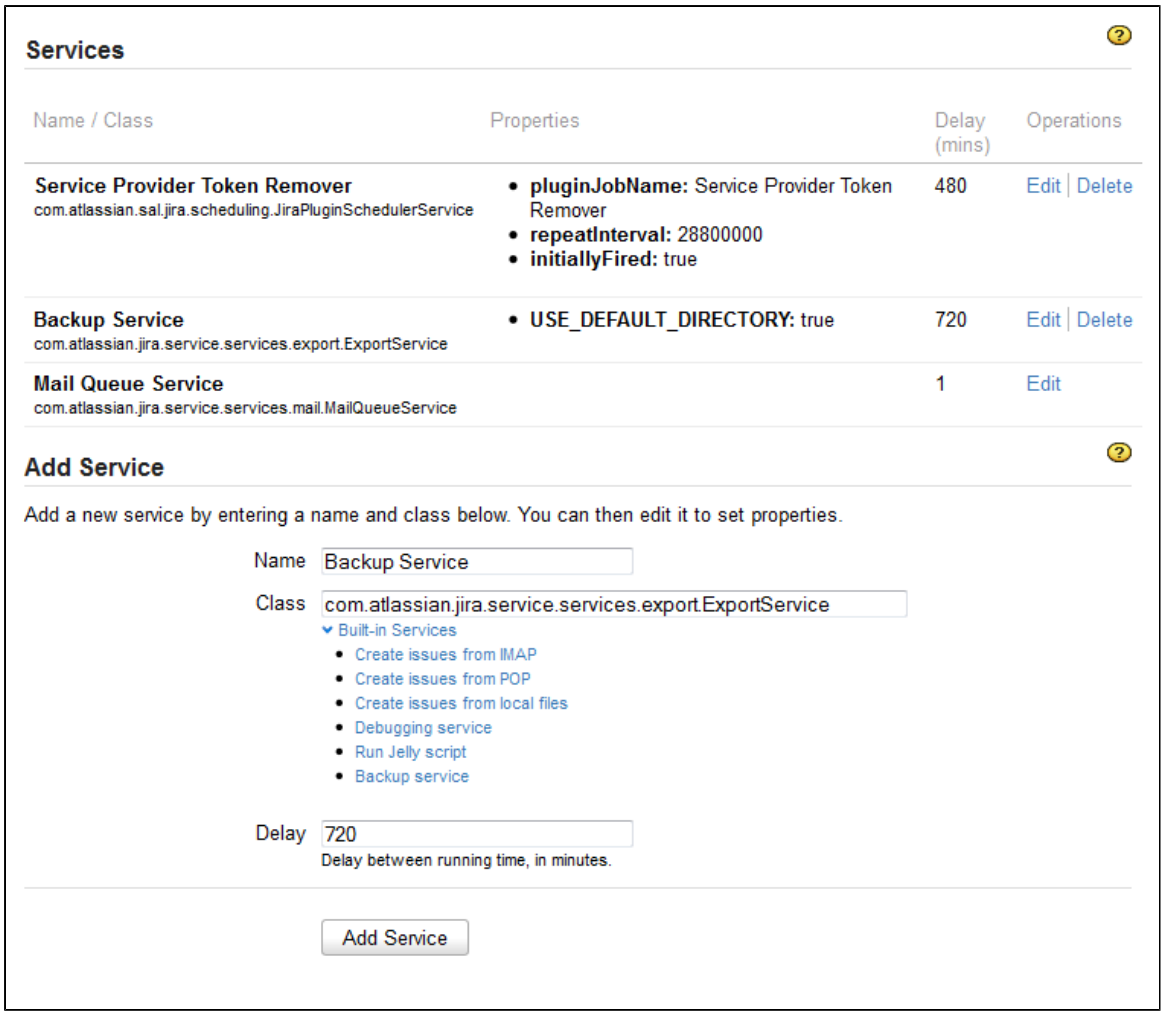
- The XML backup includes all data in the database. However, it **does not include** your attachments directory, JIRA Home Directory or JIRA Installation Directory, which are stored on the filesystem.
- You can also perform XML backups manually. See Backing Up Data for details.
- Be aware that after installing JIRA and running the setup wizard, a backup service will automatically be configured to run every 12 hours.

For production use or large JIRA installations, it is **strongly recommended** that you use native database-specific tools instead of the XML backup service. XML backups are not guaranteed to be consistent, as the database may be updated during the backup process. Inconsistent backups are created successfully without any warnings or error messages, but fail during the restore process. Database-native tools offer a much more consistent and reliable means of storing data.

*The information on this page does not apply to **JIRA OnDemand**.*

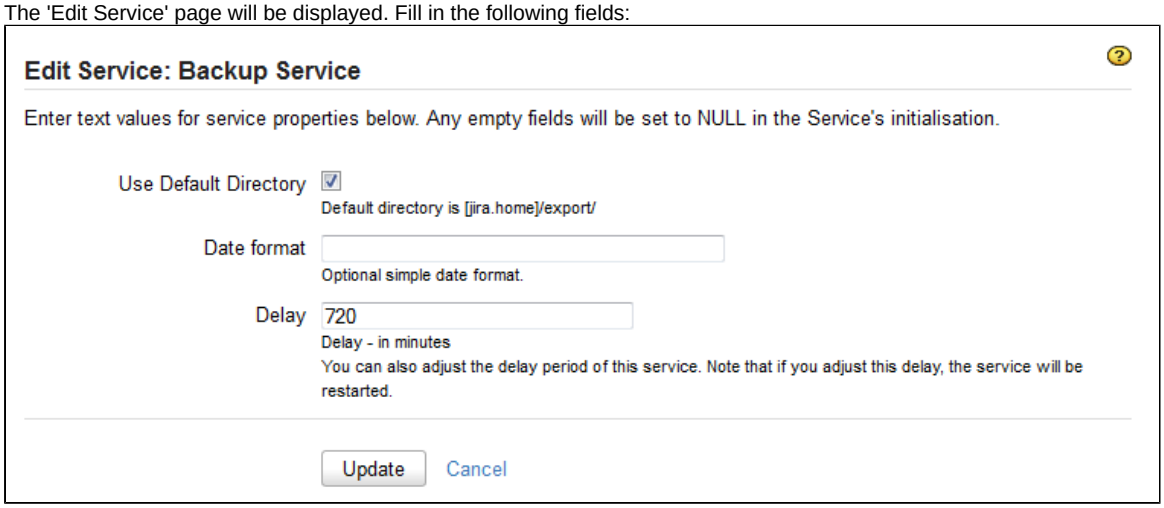
To configure automated JIRA backups:

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Advanced' > 'Services'** (tab) to open the 'Services' page, which lists the current services running on this system. By default, there should be at least one 'Mail Queue Service' running, which cannot be deleted.
   Keyboard shortcut: **'g' + 'g' +** type **'servi'**



3. In the 'Add Service' form at the bottom of the page, complete the following fields:
    - 'Name' — a descriptive name for the backup service, such as `Backup Service`.
    - 'Class' — the appropriate fully-qualified class name for the 'Backup service' using either of the following methods:
        - Select the 'Backup service' from the list of JIRA's 'built-in services'. To do this:
            a. Click the '**Built-in Services**' link below the 'Class' field to expand the list of JIRA's built-in service classes.
            b. Click the '**Backup service**' link. The 'Class' field will automatically be populated with the following class text string '`com.atlassian.jira.service.services.export.ExportService`'
        - Type the fully-qualified class name '`com.atlassian.jira.service.services.export.ExportService`' into the 'Class' field.
    - 'Delay' — enter the number of minutes between backups. A good default for this would be 720 minutes (12 hours).

      Please Note: The interval specified in the Backup Service Delay (mins) is the time *when the next backup job will run since the last server restart*. Backup services cannot be scheduled to run at a specific time of day - please see JRA-1865 for more on this.
4. Click the **'Add Service'** button.
   The 'Edit Service' page will be displayed. Fill in the following fields:



    - Select the '**Use Default Directory**' check box.
    - For the 'Date format' field, specify the format which JIRA will use to name the individual backup files. This format can be anything that SimpleDateFormat can parse. A good default is 'yyyy-MMM-dd-HHmm', which would generate files named like this: '2007-Mar-05-1322'.

- For the 'Delay' field, modify the number of minutes between backups if necessary.

5. Click the '**Update**' button. Your backup service is now configured. XML backups will be performed according to the schedule you specified in the **Delay**field.
    - For every successful backup, a zipped file of your XML backup will be saved in the backup directory.
    - If a scheduled backup fails for any reason, the zipped XML backup file will be saved into the 'corrupted' directory, which is directly under your nominated backup directory. A file explaining the reason for the failure will be written to the 'corrupted' directory. This file will have the same name as the backup file, but with the extension '.failure.txt'.

      JIRA will create the 'corrupted' directory if required - you do not need to create it.

Choosing a custom Backup Directory:

- If you upgraded JIRA with an XML backup from a JIRA version prior to 4.2 and are editing an existing backup service which used a custom directory, you can choose between using that custom directory (which cannot be edited) or the default directory for this backup service. However, once you switch to using the default directory, you can no longer choose the custom directory option.
- The default directory location is the `export` subdirectory of the JIRA Home Directory.

Preventing users from accessing JIRA during backups

For production use, it is **strongly recommended** that for regular backups, you use native database backup tools instead of JIRA's XML backup service.

When JIRA is in use, XML backups are not guaranteed to be consistent as the database may be updated during the backup process. JIRA does not report any warnings or error messages when an XML backup is generated with inconsistencies and such XML backups will fail during the restore process. Native database backup tools offer a much more consistent and reliable means of storing (and restoring) data.

*The information on this page does not apply to **JIRA OnDemand**.*

If you perform an XML backup (e.g. when upgrading JIRA via a test environment or migrating JIRA to another server), you can follow one of these methods to prevent users from accessing JIRA and minimise inconsistencies in the backup file:

- **Recommended method:**
    - If you have an Apache or other web/proxy server sitting in front of JIRA, then you can stop Apache from proxying to JIRA, and serve a static HTML page with a nice message along the lines of "JIRA is undergoing maintenance". Note:
        - The administrator must be able to access JIRA directly (not through Apache) to perform the XML backup.
        - This method does not require JIRA to be restarted.

- *Alternative method 1:*
    1. Shut down JIRA, configure it to listen on a different port and restart. Do this by editing the `server.xml` file (or the `jira.xml` file in your Apache Tomcat installation running JIRA WAR). Change the following section:

```
<Connector port="8080"
           maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25"
maxSpareThreads="75" useBodyEncodingForURI="true"
           enableLookups="false" redirectPort="8443" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true" />
```

        - Note: If you have enabled HTTPS, then you would need to edit the `HTTPS Connector` section as well.
    2. Restart JIRA and do the XML backup.
    3. Shut down JIRA, change all the settings back, then re-start JIRA.

- *Alternative method 2:*
    - If you have a firewall in front of JIRA, you could stop requests from getting through or change the port number that it uses. Note:
        - The administrator will need to log in to JIRA on the temporary port number (or access it from behind the firewall), to perform the XML backup.
        - This method does not require JIRA to be restarted.

- *Alternative method 3:*
    - In your database server, remove **'write'** permissions from the database user (e.g. `jirauser`) that JIRA uses.
        - Users will be able to read but not update issues during the backup.
        - This method does not require JIRA to be restarted.

Before you start:

Whichever method you choose, we recommend setting an Announcement Banner to warn your users that JIRA will be unavailable for a period of time.

Search Indexing

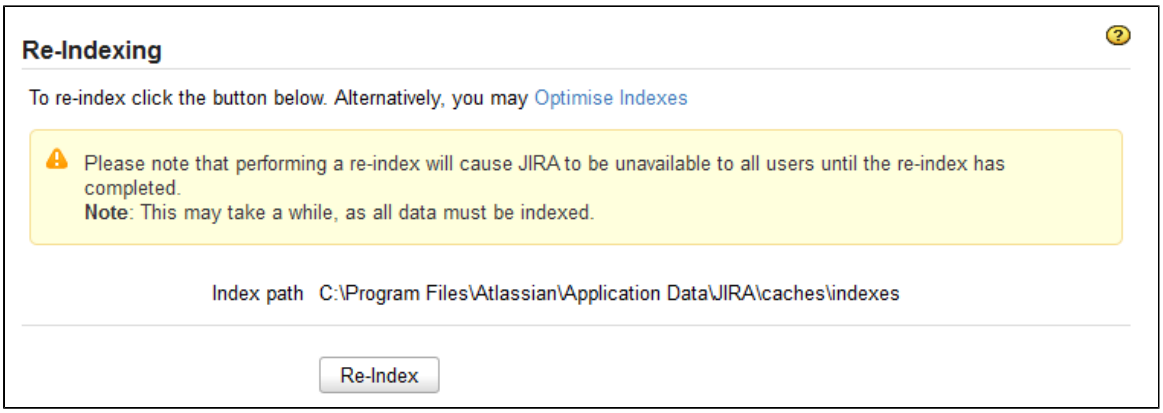
In order to provide fast searching, JIRA creates an index of the text entered into issue fields. This index is stored on the file system, and updated whenever issue text is added or modified. It is sometimes necessary to regenerate this index manually; for instance if issues have been manually entered into the database, or the index has been lost or corrupted.

*Some functionality described on this page is restricted in **JIRA OnDemand**.*

Indexing Administration

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Advanced' > 'Indexing'** (tab) to open the 'Re-Indexing' page.
   Keyboard shortcut: **'g' + 'g' +** type **'ind'**
3. This page allows you to:
   - **re-index your data**— whenever you re-index data, JIRA will clear any existing indexes and re-index all the current data from scratch. This may take a few minutes (depending on how many issues you have) and users will be unable to access JIRA during this time.
     - See also Re-Indexing after Major Configuration Changes for more information.
   - **optimise your indexes** — click the '**Optimise Indexes**' link to access this function.
     - Note that JIRA schedules an optimisation of your indexes every night at midnight.

Screenshot: Indexing JIRA



Choosing a custom Index Path:

- If you upgraded JIRA with an XML backup from a JIRA version prior to 4.2 and used a custom directory for your index path, you can choose between using this custom directory (which cannot be edited) or the default directory for your index path location. However, once you switch to using the default directory, you can no longer choose the custom directory option.
- The default directory location is the `caches/indexes` subdirectory of the JIRA Home Directory.

Re-Indexing after Major Configuration Changes

Once issues have been created, modifying the configuration of your JIRA instance can result in the search index becoming out-of-sync with JIRA's configuration. Configuration details such as the following can affect the search index:

- Field Configuration Schemes
- Custom Fields
- Plugins
- Time Tracking

If you make changes to any of these areas of configuration, you might see the following message in your Administration view:

```
USERFULLNAME made configuration changes to 'SECTION' at TIME. It is recommended that you
perform a re-index. It is recommended that you perform a re-index. For more information, please
click the Help icon.
To perform the re-index now, please go to the 'Indexing' section.
Note: So that you only have to re-index once, you may wish to  complete any other configuration
changes before performing the re-index.
```

All users that have access to the Administration Tab will see this message (JIRA Administrators, System Administrators, Project Administrators). The above message means that configuration changes have been made to JIRA, but have not yet been reflected in the search index. Until JIRA's search index has been rebuilt, it is possible that some search queries from JIRA will return incorrect results. For example:

- If a plugin containing a custom field is enabled after being disabled, search queries which specify that the custom field should be empty will return *no issues* instead of *all issues*.
- If a Field Configuration is modified by altering the visibility of a particular field, search queries which specify that field may also return erroneous results (depending on which field is being modified and what query is being executed).
- etc

The way to resolve the discrepancy is to rebuild JIRA's search index . This can take anywhere from seconds to hours, depending on the number of issues and comments in your JIRA instance. While re-indexing is taking place, your instance will be unavailable to all users. For these reasons, it is recommended that you:

- Make all your necessary configuration changes in one go before starting the re-index process; and
- Start the re-index process in a time period of low activity for your instance.

Using robots.txt to hide from Search Engines

The robots.txt protocol is used to tell search engines (Google, MSN, etc) which parts of a website should not be crawled.

For JIRA instances where non-logged-in users are able to view issues, a robots.txt file is useful for preventing unnecessary crawling of the Issue Navigator views (and unnecessary load on your JIRA server).

> *The information on this page does not apply to **JIRA OnDemand**.*

Editing robots.txt

JIRA (version 3.7 and later) installs the following `robots.txt` file at the root of the JIRA webapp:

```
# robots.txt for JIRA
# You may specify URLs in this file that will not be crawled by search engines (Google, MSN,
etc)
#
# By default, all SearchRequestViews in the IssueNavigator (e.g.: Word, XML, RSS, etc) and all
IssueViews
# (XML, Printable and Word) are excluded by the /sr/ and /si/ directives below.

User-agent: *
Disallow: /sr/
Disallow: /si/
```

Alternatively, if you already have a `robots.txt` file, simply edit it and add **`Disallow: /sr/`** and **`Disallow: /si/`**.

Publishing robots.txt

The `robots.txt` file needs to be published at the root of your JIRA internet domain, e.g. **`jira.mycompany.com/robots.txt`**.

> If your JIRA instance is published at **`jira.mycompany.com/jira`**, change the contents of the file to `Disallow: /jira/sr/` and `Disallow: /jira/sr/`. However, you still need to put `robots.txt` file in the root directory, i.e. `jira.mycompany.com/robots.txt` (not `jira.mycompany.com/jira/robots.txt`).

Updating your JIRA License Details

When you upgrade or renew your JIRA license, you will receive a new license key. You will need to update your JIRA server with the new

license key.

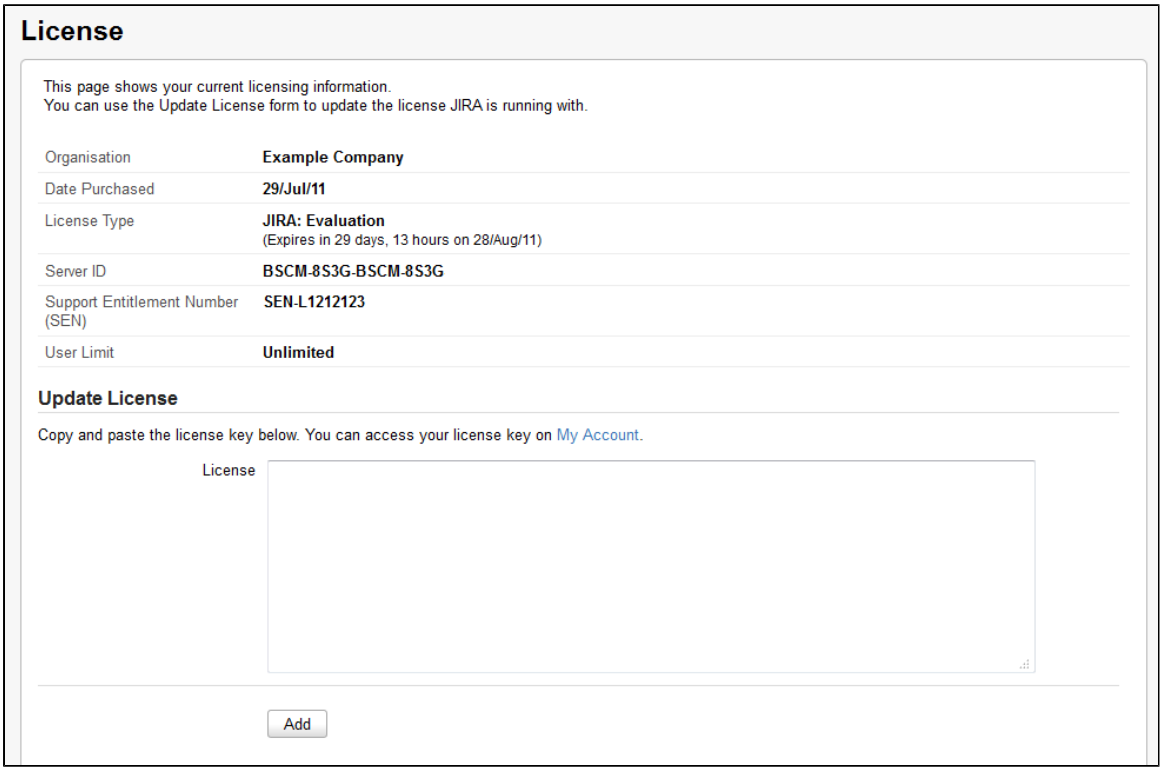
You can access your license key via http://my.atlassian.com

To update your JIRA license key:

1. Log in to JIRA as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'License'** to open the 'License' page. This page displays your existing JIRA license details with an 'Update License' section below that.
   Keyboard shortcut: **'g' + 'g' +** start typing **'license'**
3. Paste your new license into this box.
   You can retrieve existing licenses or generate an evaluation one by clicking the '**My Account**' link.
4. Click the '**Add**' button to update the JIRA installation with the new license.

> *The information on this page does not apply to **JIRA OnDemand**.*

Screenshot : License Details



Do you have a user-limited license?

If you have a user-limited license (such as a starter license), then the user limit of your license will be displayed on the '**License Details**' page. This field is periodically refreshed, but you can retrieve the latest user limit by clicking the '**Refresh**' button.

Need more information about licensing or want to find out more about starter licenses? Please see the Licensing FAQ and Starter Licenses page.

Reducing your user count

You may want to reduce your user count in JIRA if you have *exceeded your user count* or if you want to *change to a lower-tier license* to reduce costs. For instructions on how to do this, see How do I reduce my user count in JIRA.

Note, if you exceed the user count allowed by your JIRA license, your users will not be able to create issues.

Viewing your System Information

JIRA provides you with detailed information about your system configuration, as described in the table below. This information can be useful when modifying, troubleshooting or upgrading your system.

Viewing your JIRA System Information

1. Log in as a user with the '**JIRA Administrators**' global permission.
2. Select **'Administration' > 'System' > 'Troubleshooting and Support' > 'System Info'** (tab) to open the 'System Info' page.
   Keyboard shortcut: **'g' + 'g' +** type **'system i'**
   The following categories of information is shown on the 'System Info' page:
   - Warnings
   - System Info
   - Java VM Memory Statistics
   - JIRA Info
   - License Info
   - Configuration Info
   - Database Statistics
   - File Paths
   - Listeners
   - Services
   - Plugins
   - System Properties
   - Trusted Applications

Warnings

Any warnings about known issues with your configuration will be displayed here.

System Info

Setting	Description
Base URL	The base URL of this JIRA installation. It is used in outgoing email notifications as the prefix for links to JIRA issues. It can be changed as described in Configuring JIRA Options.
System Date	The JIRA server's system date.
System Time	The JIRA server's system time.
Current Working Directory	For a description of the JIRA Working Directory, please see Important Directories and Files.
Java Version	The JIRA server's Java version.
Java Vendor	The JIRA server's Java vendor.
JVM Version	The JIRA server's JVM version.
JVM Vendor	The JIRA server's JVM version.
JVM Implementation Version	The JIRA server's JVM implementation version.
Java Runtime	The JIRA server's Java runtime environment.
Java VM	The JIRA server's Java Virtual Machine.
User Name	The operating system login name which JIRA runs under.
User Timezone	The JIRA server's timezone.
User Locale	The JIRA server's locale. Unless the default language is modified in JIRA's General Configuration, the User Locale will dictate the default language.

System Encoding	The JIRA server's system encoding.
Operating System	The JIRA server's operating system.
OS Architecture	The JIRA server's operating system architecture (e.g. i386).
Application Server Container	The application server in which your JIRA instance is running (see Supported Platforms for a list of supported application servers).
Database type	The type of database to which your JIRA instance is connected (see Supported Platforms for a list of supported databases).
Database JNDI address	The JNDI address of the database to which your JIRA instance is connected. (For more details, see Connecting JIRA to a Database.)
Database URL	The URL of the database to which your JIRA instance is connected. (For more details, see Connecting JIRA to a Database.)
Database version	The version of the database to which your JIRA instance is connected (see Supported Platforms for a list of supported database versions).
Database driver	The driver which your JIRA instance is using to connect to its database. (For more details, see Connecting JIRA to a Database.)
External user management	'**ON**' / '**OFF**' indicates whether JIRA's users are being managed externally or internally to JIRA (e.g. via Crowd).
Crowd integration	'**YES**' / '**NO**' indicates whether Atlassian's Crowd identity management system has been integrated with this instance of JIRA. For more information please see the chapter titled 'Integrating JIRA with Crowd' in the Crowd documentation.
JVM Input Arguments	A list of any variables that are being passed to your application server when it starts up. For more information, see Setting Properties and Options on Startup.
Modified Files	A list of any files in your JIRA installation that have been modified as part installation or customisation of JIRA.
Removed Files	A list of any files that have beeen removed from your JIRA installation.

Java VM Memory Statistics

Java applications, such as JIRA, run in a "Java virtual machine" (JVM) instead of directly within an operating system. When started, the Java virtual machine is allocated a certain amount of memory, which it makes available to applications like JIRA. The following table shows the JVM memory data for your JIRA instance.

Setting	Description
Total Memory	The total amount of memory allocated to the JVM that is available to this instance of JIRA. For more details, see Increasing JIRA Memory.
Free Memory	The amount of free JVM memory currently available to this instance of JIRA.
Used Memory	The amount of JVM memory currently being used by this instance of JIRA.
Total PermGen Memory	The total amount of PermGen (Permanent Generation) memory available to this instance of JIRA.

Free PermGen Memory	The amount of free PermGen (Permanent Generation) memory currently available to this instance of JIRA.
Used PermGen Memory	The amount of PermGen (Permanent Generation) memory currently being used by this instance of JIRA.
Memory Graph	A bar graph showing the available versus free JVM memory. You can click the '**Force garbage collection**' link to start a clean-up. Note that **this is generally not needed (even if the graph shows 100% utilisation)** unless you want to examine JIRA's baseline heap usage.
PermGen Memory Graph	A bar graph showing the available versus free PermGen (Permanent Generation) memory.
Non-Heap Memory Graph (includes PermGen)	A bar graph showing the available versus free non-heap memory (including PermGen memory).

You can click the '**More Information...**' link at the bottom of this table to view an additional section titled '**Memory Pool Info**' (which lists detailed information about the various parts of memory that the Java virtual machine uses to store its data, and is generally only useful to Atlassian's support engineers.)

JIRA Info

Setting	Description
Uptime	The period of time since your JIRA instance was last started.
Edition	The 'edition' of JIRA you are running. (Note: from JIRA 4.0 onwards, only '**Enterprise**' edition is available. Prior to JIRA 4.0, there were three editions: '**Standard**', '**Professional**' and '**Enterprise**'.)
Version	The version of JIRA you are running. (Note: for the latest version, please see JIRA Releases.)
Build Number	The build number of your JIRA version. This is generally only useful to Atlassian's support engineers.
Build Date	The date on which your JIRA version was built. This is generally only useful to Atlassian's support engineers.
Atlassian Partner	Indicates whether your distribution of JIRA was built by an Atlassian partner company. Blank indicates that it was built directly by Atlassian.
Installation Type	Indicates whether JIRA has been installed as a 'recommended' distribution or as a 'WAR' distribution.
Server ID	This number is calculated automatically by JIRA, based on your license number.
Last Upgrade	The time at which your JIRA installation was last upgraded, and from which version it was upgraded from (if applicable). Click the '**More Information...**' link to see a list of all upgrades that have been performed on your JIRA system from version 4.1 onwards.
Installed Languages	A list of all language packs available within the JIRA system. (Note: to install additional languages, see Translating JIRA.)
Default Language	The language used throughout the JIRA interface. To change the default language, see Configuring JIRA Options. Note that users can override the default language by using the **Language** setting in their user profile.

License Info

To edit your license details, please see Updating your JIRA License Details. Note that you will require the **'JIRA System Administrators'** global permission.

Setting	Description
Date Purchased	The date on which this system's JIRA license was originally purchased. Note: you can verify this information by visiting http://my.atlassian.com
License Type	For information about the different types of JIRA licences, please see http://www.atlassian.com/software/jira/licensing.jsp
Maintenance Period End Date	For information about JIRA support and maintenance, please see http://www.atlassian.com/software/jira/licensing.jsp
Maintenance Status	For information about JIRA support and maintenance, please see http://www.atlassian.com/software/jira/licensing.jsp
Support Entitlement Number (SEN)	For information about JIRA support and maintenance, please see http://www.atlassian.com/software/jira/licensing.jsp

Configuration Info

Setting	Description
Attachments Enabled	'**true**' / '**false**' indicates whether or not users can attach files and screenshots to issues in this JIRA system (subject to project permissions). For more information, see Configuring File Attachments.
Issue Voting Enabled	'**true**' / '**false**' indicates whether or not users can vote on issues in this JIRA system (subject to project permissions). For more information, see Configuring JIRA Options.
Issue Watching Enabled	'**true**' / '**false**' indicates whether or not users can watch issues in this JIRA system (subject to project permissions). For more information, see Configuring JIRA Options.
Unassigned Issues Enabled	'**true**' / '**false**' indicates whether or not issues can be 'unassigned' (i.e. assigned to noone) in this JIRA system. For more information, see Configuring JIRA Options.
Sub-Tasks Enabled	'**true**' / '**false**' indicates whether or not 'sub-task' issues can be created in this JIRA system. For more information, see Configuring Sub-tasks.
Issue Linking Enabled	'**true**' / '**false**' indicates whether or not issues can be linked to each other within this JIRA system. For more information, see Configuring Issue Linking.
Time Tracking Enabled	'**true**' / '**false**' indicates whether or not time (work) can be logged on issues in this JIRA system. For more information, see Configuring Time Tracking.
Time Tracking Hours Per Day	The number of hours per working day for which work that can be logged on issues in this JIRA system. For more information, see Configuring Time Tracking.
Time Tracking Days Per Week	The number of days per week for which work that can be logged on issues in this JIRA system. For more information, see Configuring Time Tracking.

Database Statistics

The information in this section can help determine how much resource (e.g. memory) your JIRA system requires.

Setting	Description
Issues	The number of issues that have been created in this JIRA system.
Projects	The number of projects that have been created in this JIRA system.
Custom Fields	The number of custom fields that have been created in this JIRA system.
Workflows	The number of workflows that have been created in this JIRA system.
Users	The number of user IDs that have been created in this JIRA system.
Groups	The number of groups that have been created in this JIRA system.

File Paths

Setting	Description
Location of JIRA Home	The path to your JIRA Home Directory. For information about changing the location, see Setting your JIRA Home Directory.]
Location of entityengine.xml	The path to your Entity Engine. For database-specific information about configuring your `entityengine.xml` file, see Connecting JIRA to a Database.
Location of atlassian-jira.log	The path to the JIRA log file. (Note that, if you are requesting support, the support engineers will generally need your application server log file as well as your JIRA log file.) For information about changing the logging level, see Logging and Profiling; note that you will require the **'JIRA System Administrators'** global permission.
Location of indexes	The path to your JIRA search indexes (not your database indexes). For information about moving the indexes, please see Search Indexing]; note that you will require the **'JIRA System Administrators'** global permission.

Listeners

This section lists all the listeners that are installed in this JIRA system. For more information, please see Listeners. Note that you will require the '**JIRA System Administrators**' global permission in order to register a listener.

Services

This section lists all the services that are installed in this JIRA system. For more information, please see Services. Note that you will require the '**JIRA System Administrators**' global permission in order to register a service.

Plugins

This section lists all plugins that are installed in this JIRA system. For more information, please see Managing JIRA's Plugins.

System Properties

The information in this section is specific to the application server and Java version you are using, and is generally only useful to Atlassian's support engineers.

Trusted Applications

This section lists all 'trusted application' (i.e. applications that JIRA will allow to access specified functions on behalf of any user — without the user logging in to JIRA). To edit the trusted applications for this JIRA system, please see Configuring Trusted Applications. Note that you will require the '**JIRA System Administrators**' global permission.

Generating a Thread Dump

Occasionally, JIRA may appear to 'freeze' during execution of an operation. During these times, it is helpful to retrieve a **thread dump** — a

log containing information about currently running threads and processes within the Java Virtual Machine. Taking thread-dumps is a non-destructive process that can be run on live systems. This document describes the steps necessary to retrieve a **thread dump.**

The steps necessary to retrieve the **thread dump** are dependant on the operating system JIRA is running in — please follow the appropriate steps below.

On this page:

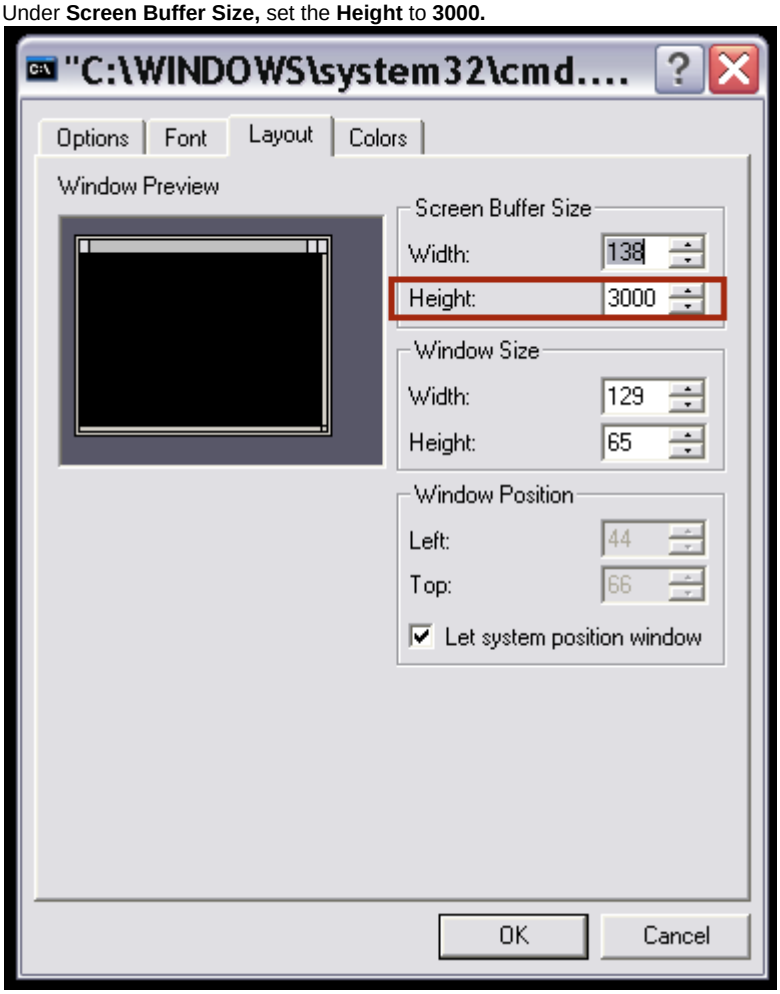
- Windows Environment
- Linux/Unix/OS X Environment
- Analysis Tools

*The information on this page does not apply to **JIRA OnDemand**.*

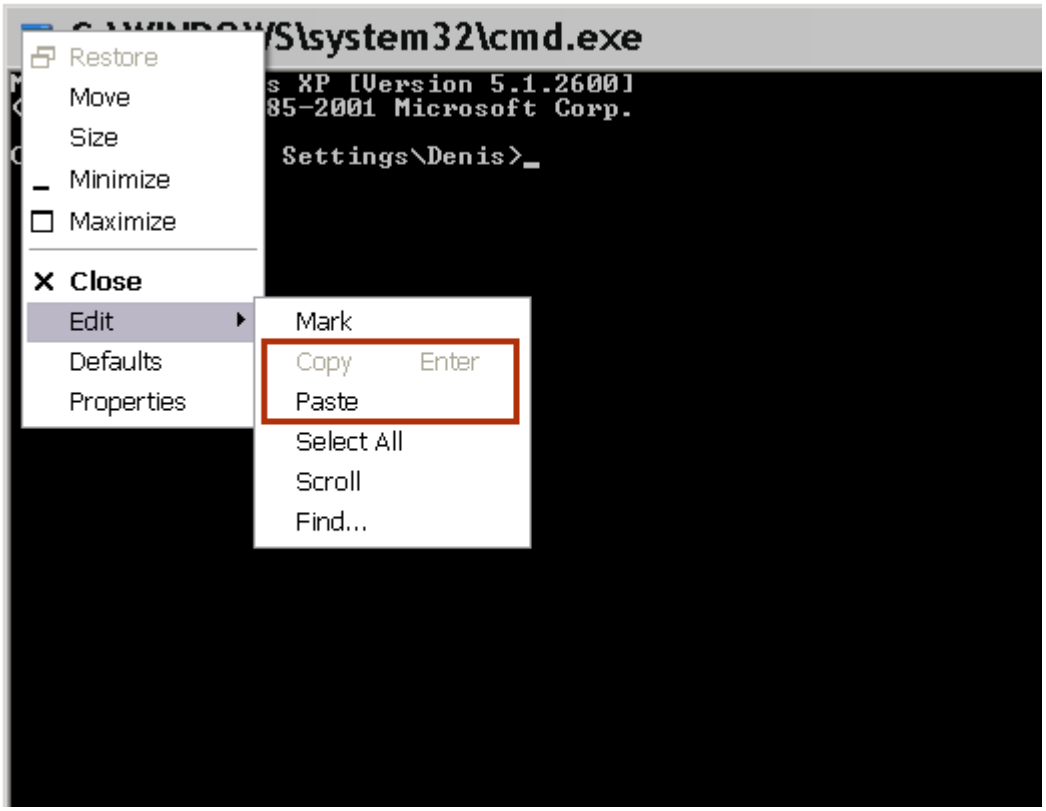
Windows Environment

JIRA running from startup.bat

1. In the **Command Console** window where JIRA is running, open the properties dialog box by right clicking on the title bar and select **"Properties".**
2. Select the **Layout** tab.
3. Under **Screen Buffer Size,** set the **Height** to **3000.**



4. Click **OK.**
5. With the same command console in focus, press **CTRL-BREAK.** This will output the thread dump to the command console.
6. Scroll back in the command console until you reach the line containing "Full thread dump".
7. Right click the title bar and select **Edit -> Mark.** Highlight the entire text of the thread dump.
8. Right click the title bar and select **Edit -> Copy.** The thread dump can then be pasted into a text file.



JIRA running as a Windows Service

Using jstack

The JDK ships with a tool named jstack for generating thread dumps.

1. Identify the process. Launch the task manager by, pressing `Ctrl + Shift + Esc` and find the Process ID of the Java (JIRA) process. You may need to add the PID column using `View` -> `Select Columns ...`
2. Run jstack <pid> to Capture a Single Thread Dump. This command will take one thread dump of the process id <pid>, in this case the pid is 22668:

```
adam@track:~$ jstack \-l 22668 > threaddump.txt
```

This will output a file called threaddump.txt to your current directory.

Common issues with jstack:

- You must run jstack as the same user that is running JIRA
- If the jstack executable is not in your $PATH, then please look for it in your <JDK_HOME>/bin directory
- If you receive `java.lang.NoClassDefFoundError: sun/tools/jstack/JStack` check that tools.jar is present in your JDK's lib directory. If it is not, download a full version of the JDK.

Linux/Unix/OS X Environment

Linux/Unix Command Line

1. Identify the **java**process that JIRA is running in. This can be achieved by running a command similar to:

```
ps -ef | grep java
```

The process will appear similarly as follows:

```
keithb     910   873  1 17:01 pts/3    00:00:18 /usr/java/jdk/bin/java -Xms128m -Xmx256m
-Xms128m -Xmx256m -Djava.awt.headless=true
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djava.awt.headless=true
-Djava.endorsed.dirs=/tmp/atlassian-jira-enterprise-3.6-standalone/common/endorsed
-classpath :
```

2. In order to retrieve the thread dump, execute the command

```
kill -3 <pid>
```

   where **pid** is the process id — in this case, 910.

3. The thread dump is logged to the console in which JIRA was started.

Linux/Unix Alternative: Generating Thread Dumps using jstack

If you have trouble using kill -3 <pid> to obtain a thread dump, try using jstack a java utility that will output stack traces of Java threads for a given process.

1. Identify the **java**process that JIRA is running in. This can be achieved by running a command similar to:

```
ps -ef | grep java
```

2. The process will appear similarly as follows:

```
adam 22668 0.3 14.9 1691788 903928 ? Sl Jan27 9:36
/usr/lib/jvm/java-6-sun-1.6.0.14/bin/java
-Djava.util.logging.config.file=/home/adam/Products/installs/atlassian-jira-enterprise-4.0.1-
-XX:MaxPermSize=256m -Xms128m -Xmx1048m -Djava.awt.headless=true
-Datlassian.standalone=JIRA -Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true
-Dmail.mime.decodeparameters=true -Datlassian.mail.senddisabled=false
-Datlassian.mail.fetchdisabled=false
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djava.endorsed.dirs=/home/adam/Products/installs/atlassian-jira-enterprise-4.0.1-standalone/
-classpath
/home/adam/Products/installs/atlassian-jira-enterprise-4.0.1-standalone/bin/bootstrap.jar
-Dcatalina.base=/home/adam/Products/installs/atlassian-jira-enterprise-4.0.1-standalone
-Dcatalina.home=/home/adam/Products/installs/atlassian-jira-enterprise-4.0.1-standalone
-Djava.io.tmpdir=/home/adam/Products/installs/atlassian-jira-enterprise-4.0.1-standalone/temp
org.apache.catalina.startup.Bootstrap start
```

3. Run jstack <pid> to Capture a Single Thread Dump
   This command will take one thread dump of the process id <pid>, in this case the pid is 22668, and log output to the file JIRAthreaddump.txt

```
adam@jiratrack:~$ jstack 22668 > JIRAthreaddump.txt
```

4. Take Multiple Thread Dumps
   Typically you'll want to take several dumps about 10 seconds apart, in which case you can generate several dumps and output the stack traces to a single file as follows:

```
adam@jiratrack:~$ jstack 22668 >> JIRAthreaddump.txt
adam@jiratrack:~$ jstack 22668 >> JIRAthreaddump.txt
adam@jiratrack:~$ jstack 22668 >> JIRAthreaddump.txt
```

Analysis Tools

Try TDA or Samurai to inspect your thread dump.

TDA

1. Download TDA
2. CD to the directory where the JAR exists
3. Run:

```
java -jar -Xmx512M ~/tda-bin-1.6/tda.jar
```

4. Open your catalina.out file, containing the thread dump

Check the known thread dump knowledge base articles:

No content found for label(s) thread_dump.

Performance Testing Scripts

Please be aware that the content on this page is not actively maintained and Atlassian **can not guarantee providing any support for it**. This page is provided for your information only and using it is done so at your own risk.

This page contains scripts and hints for testing usage load on your JIRA installation.

When setting up a new JIRA installation, it is useful to understand how it will perform under your anticipated load before users begin accessing it. Scripts that generate 'request' (or usage) load are provided in our public Maven repository (link below). Using these scripts, you can find out where you may need to consider improving your configuration to remove bottlenecks.

While this kind of testing is not an exact science, the tools and processes described here are intended to be straightforward and configurable and provide you with an extensible way to assess load testing.

The performance tests described on this page utilise JMeter. While it is not necessary to know JMeter, briefly reading through the JMeter documentation is recommended as it may help you resolve any JMeter-specific issues.

It is rarely the case that these scripts will perform representative testing for you 'out of the box'. However, it should be possible to build an appropriate load test by configuring or extending these scripts.

Load testing scripts should not be used on a production JIRA installation!

While we recommend using a copy of your production data for testing usage load, the load testing scripts below *will modify data within the targeted JIRA installation!* Hence, these scripts *should not be used on a production JIRA installation*. Instead, use a copy of your production JIRA data on a *test* JIRA installation.

If you do run these test scripts against your production JIRA installation, you will be responsible for any data loss and backup recovery!

Likewise, when making changes to your JIRA installation to remove performance bottlenecks, it is useful to assess the impact of these changes in a test JIRA installation before implementing them in production.

On this page:

- Prerequisites
- Quick, just tell me how to run these tests!
- Performance Tests

*The information on this page does not apply to **JIRA OnDemand**.*

Prerequisites

You will need the following:

- A JIRA installation, set up and running with an administrator user. The scripts assume that the username/password combination of this user is 'admin'/'admin'.

- It is recommended that you test with a production quality database, such as one listed on the Supported Platforms page. Do not use HSQLDB.
- Apache JMeter (currently version 2.3.4). If you intend to do high load testing, please use our modified version of JMeter instead (which requires Java 1.6).
- The load testing scripts and resources which are available in our public Maven repository — Please choose the version that most closely matches your JIRA version and download the ZIP or Gzip file in that directory. If in doubt, download the ZIP file archive.

Users have reported problems using the Windows built-in unzip utility to extract these archives. If you encounter such a problem, please use a third party file archiving and extraction program (for example, 7-Zip) to extract these performance tests.

Quick, just tell me how to run these tests!

If you do not want to read the rest of this document, here are the main points:

1. Create the **setup test**data:

```
<jmeter location>/bin/jmeter -n -t jmeter-test-setup.jmx -Jadmin.user=<username>
-Jadmin.pass=<password>
```

2. Run the **fixed load test**:

```
<jmeter location>/bin/jmeter -n -t jmeter-test-fixedload.jmx
```

The remainder of this document is just an elaboration of those two steps.

For information on how to use JMeter please refer to the JMeter documentation.

Performance Tests

JIRA performance tests are made up of two parts:

- **Setup test** — runs first and prepares the JIRA installation for a subsequent **fixed load test**
- **Fixed load test** — simulates a number of users accessing the JIRA installation.

Setup Test

The **setup test** is responsible for:

- Creating projects
- Creating users
- Creating and commenting on (and optionally resolving) issues.

Running the setup test:

After extracting the performance test zip file, change into the `performanceTest` directory. From this directory, run the performance setup test:

```
<jmeter location>/bin/jmeter -n -t jmeter-test-setup.jmx -Jadmin.user=<username>
-Jadmin.pass=<password>
```

where `<jmeter.location>` is the base directory of JMeter

If you omit the `-n` switch, JMeter will run as a GUI. You may then start the test from within the GUI.

As seen above with the `admin.user` and `admin.pass` parameters, JMeter supports `-Jparameter=value` command arguments in order to control execution. The following parameters control the setup test execution:

Configuration Control

Parameter	Default	Explanation
jira.host	localhost	The hostname or address of the JIRA installation.
jira.port	8000	The network port that the JIRA installation is running on.
jira.context	/	JIRA webapp context.
admin.user	admin	Administrator username.
admin.pass	admin	Administrator password.
script.base	.	The location of the performance tests. This should only be set if you run the tests from outside the `scripts` directory.
remove.data	false	Running the script with this enabled will remove the users and projects created by the test.

User Control

Parameter	Default	Explanation
create.users.enable	true	Create users in the target JIRA installation. Use **false** if you already have the users created elsewhere.
browseissue.max	250	The number of users to be created for browsing the JIRA installation (aka "browseissue" users).
createissue.max	250	The number of users to be created for creating issues (aka "createissue" users).
editissue.max	250	The number of users to be created for editing issues (aka "editissue" users).
search.max	250	The number of users to be created for searching issues (aka "search" users).
useraction.max	250	The number of users to be created for browsing user information (aka "useraction" users).
browseissue.groupname	none	The group to which "browseissue" users will be placed. Use **none** for no group.
createissue.groupname	jira-developers	The group to which "createissue" users will be placed. Use **none** for no group.
editissue.groupname	jira-developers	The group to which "editissue" users will be placed. Use **none** for no group.
search.groupname	none	The group to which "search" users will be placed. Use **none** for no group.
useraction.groupname	jira-developers	The group to which "useraction" users will be placed. Use **none** for no group.

Project Control

Parameter	Default	Explanation
create.projects.enable	true	Create projects. Use **false** if you want to use existing projects (in existing data).
project.max	20	The number of projects to create in the system.

Issue Control

Parameter	Default	Explanation
create.issues.enable	true	Creates issues in the target JIRA installation. Use **false** if you do not want the test to create sample issues.
issue.max	3000	The number of issues to be created.
issue.comment.enable	true	Controls whether or not comments are added to issues.
issue.comment.max	10	If `issue.comment.enable` is true, then the number of actual comments created on an issue is chosen randomly between 0 and this value.
issue.close	true	Controls whether or not issues will be closed automatically after being created.
issue.close.percentage	60	If `issue.close` is enabled, then this value defines the percentage of issues closed.
issue.setupload.threads	10	The number of threads used for creating the issues.
issue.setupload.pause	50	The amount of time (in milliseconds) for which a simulated user will 'sleep' between each request during issue creation.
resource.dir	resources	The directory which contains the CSV data resources.

Test Output

Once you have chosen your target settings, run JMeter and you should get output similar to the following:

```
jmeter -n -t jmeter-test-setup.jmx
Created the tree successfully using jmeter-test-setup.jmx
Starting the test @ Mon Oct 26 23:53:28 CDT 2009 (1256619208435)
Generate Summary Results +   931 in  31.3s =   29.7/s Avg:    26 Min:    13 Max:  3256 Err:
0 (0.00%)
Generate Summary Results +  2948 in 180.0s =   16.4/s Avg:    31 Min:     8 Max:  1162 Err:
0 (0.00%)
Generate Summary Results =  3879 in 211.4s =   18.3/s Avg:    29 Min:     8 Max:  3256 Err:
0 (0.00%)
Generate Summary Results +  5048 in 179.9s =   28.1/s Avg:    44 Min:     7 Max:   936 Err:
0 (0.00%)
Generate Summary Results =  8927 in 391.4s =   22.8/s Avg:    37 Min:     7 Max:  3256 Err:
0 (0.00%)
Generate Summary Results +  3114 in 180.1s =   17.3/s Avg:    41 Min:     7 Max:   805 Err:
0 (0.00%)
Generate Summary Results = 12041 in 571.3s =   21.1/s Avg:    38 Min:     7 Max:  3256 Err:
0 (0.00%)
Generate Summary Results +  4956 in 179.8s =   27.6/s Avg:    45 Min:     7 Max:  1844 Err:
0 (0.00%)
Generate Summary Results = 16997 in 751.4s =   22.6/s Avg:    40 Min:     7 Max:  3256 Err:
0 (0.00%)
Generate Summary Results +   313 in  17.1s =   18.3/s Avg:    37 Min:     7 Max:   165 Err:
0 (0.00%)
Generate Summary Results = 17310 in 768.5s =   22.5/s Avg:    40 Min:     7 Max:  3256 Err:
0 (0.00%)
Tidying up ...    @ Tue Oct 27 00:06:17 CDT 2009 (1256619977181)
... end of run
```

This output will be updated every 3 minutes, showing the most recent activity as well as a summary for the whole test.

Result Logs

In addition to this summary data, which is output on the command line, log files are created for both the successful (`jmeter-results-setup.jtl`) and unsuccessful (`jmeter-assertions-setup.jtl`) results. These log files are saved in JTL format (which is based on XML). There are a number of parsers which will generate graphs from these log files. For more information, see the JMeter wiki page on Log Analysis.

Fixed Load Test

Once the **setup test** has successfully run, the **fixed load test** can be run. This test will simulate a number of users accessing the JIRA installation.

This test has a number of parameters for tweaking the behaviour if the test. By default, the test has the following behaviour and strategy:

- Several groups of users, all running concurrently for a fixed amount of time, each with a small delay between requests.
  - 'Edit Issue' (editissue) users browse a project and then attempt to find an issue. They will then comment, edit or change the workflow of that issue.
  - 'User Action' (useraction) users create filters, view watches and votes.
  - 'Browse Issue' (browseissue) users browse projects and issues.
  - 'RSS' users browse project and then periodically fetch the RSS feed for that project.
  - 'Create Issues' (createissue) users add new issues to the instance.
  - 'Search Issues' (search) users search for issues using the quick search textbox.

There is **no execution of JavaScript** by the JMeter client. JavaScript performance will depend on several factors such as your browser and workstation speed. JMeter does not measure this.

Running the fixed load test:

```
<jmeter location>/bin/jmeter -n -t jmeter-test-fixedload.jmx
```

As with the setup test (above), this command will run the **fixed load test** with the default values. Similarly, it is possible to control the execution of JMeter with `-J` parameters. The fixed load test has the following available parameters:

Configuration Control

Parameter	Default	Explanation
jira.host	localhost	The hostname or address of the JIRA installation.
jira.port	8000	The network port that the JIRA installation is running on.
jira.context	/	JIRA webapp context.
admin.user	admin	Administrator username.
admin.pass	admin	Administrator password.
script.base	.	The location of the performance tests. This should only be set if you run the tests from outside the `scripts` directory.
script.runtime	1800	The amount of time to run for (in seconds).
resource.dir	resources	The subdirectory which contains the resource CSV files. Replace this if you wish to customise the backend data.

Edit Issue

Parameter	Default	Explanation
editissue.threads	5	The number of simultaneous 'Edit Issue' users to simulate.
editissue.pause	15000	The pause between each 'Edit Issue' user request (in milliseconds).
workflow.matchname	(Close | Resolve)	A regular expression to match the workflow to action.
editworkflow.percentage	20	The percentage of 'Edit Issue' user requests that will attempt to change the issue workflow.

addcomment.percentage	60	The percentage of 'Edit Issue' user requests that will attempt to add a comment to an issue.
editissue.percentage	20	The percentage of 'Edit Issue' user requests that will attempt to edit an issue.
editissue.issuestoown	5	The number of issues the test attempts to assign to an 'Edit Issue' user.

User Actions

Parameter	Default	Explanation
useraction.threads	1	The number of simultaneous 'User Action' users to simulate.
useraction.pause	40000	The pause between each 'User Action' user request (in milliseconds).
createfilter.percentage	10	The percentage of 'User Action' user requests that will attempt to create a filter.
viewwatches.percentage	10	The percentage of 'User Action' user requests that will attempt to 'view watches'.
viewvotes.percentage	10	The percentage of 'User Action' user requests that will attempt to view votes.

Browse Issues and Projects

Parameter	Default	Explanation
browseissue.threads	40	The number of simultaneous 'Browse Issue' users to simulate.
browseissue.pause	3000	The pause between each 'Browse Issue' user request (in milliseconds).
userprofile.percentage	10	The percentage of 'Browse Issue' user requests that will attempt to browse a user profile.
browsedashboard.percentage	20	The percentage of 'Browse Issue' user requests that will attempt to browse the dashboard.
dashboard.category	allprojects	The project category for project status gadget requests.

RSS

Parameter	Default	Explanation
browserss.threads	10	The number of simultaneous 'RSS' users to simulate.
browserss.pause	60000	The pause between each 'RSS' user request (in milliseconds).

Create Issues

Parameter	Default	Explanation
issue.create.threads	3	The number of simultaneous 'Create Issue' users to simulate.
issue.create.pause	15000	The pause between each 'Create Issue' user request (in milliseconds).
issue.comment.max	2	The maximum number of comments on an issue. The actual number is chosen randomly between 0 and this value.

Search For Issues

Parameter	Default	Explanation
search.threads	2	The number of simultaneous 'Search' users to simulate.
search.pause	15000	The pause between each 'Search' user request (in milliseconds).

Test Output

Once you have chosen your target settings, run JMeter and you should get output similar to the following:

```
jmeter -n -t jmeter-test-fixedload.jmx
Created the tree successfully using jmeter-test-fixedload.jmx
Starting the test @ Wed Oct 28 01:13:22 CDT 2009 (1256710402435)
Waiting for possible shutdown message on port 4445
Generate Summary Results +   568 in  97.9s =     5.8/s Avg:    62 Min:     1 Max:  1534 Err:
0 (0.00%)
Generate Summary Results +  3861 in 179.4s =    21.5/s Avg:    39 Min:     0 Max:   494 Err:
0 (0.00%)
Generate Summary Results =  4429 in 277.4s =    16.0/s Avg:    42 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results +  7356 in 180.0s =    40.9/s Avg:    37 Min:     0 Max:   481 Err:
0 (0.00%)
Generate Summary Results = 11785 in 457.3s =    25.8/s Avg:    39 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 10841 in 180.1s =    60.2/s Avg:    38 Min:     0 Max:   995 Err:
0 (0.00%)
Generate Summary Results = 22626 in 637.4s =    35.5/s Avg:    39 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 11821 in 180.3s =    65.6/s Avg:    37 Min:     0 Max:   507 Err:
0 (0.00%)
Generate Summary Results = 34447 in 817.3s =    42.1/s Avg:    38 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 11904 in 180.9s =    65.8/s Avg:    38 Min:     0 Max:   658 Err:
0 (0.00%)
Generate Summary Results = 46351 in 997.4s =    46.5/s Avg:    38 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 11697 in 180.3s =    64.9/s Avg:    38 Min:     0 Max:   488 Err:
0 (0.00%)
Generate Summary Results = 58048 in 1177.4s=    49.3/s Avg:    38 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 11731 in 180.0s =    65.2/s Avg:    39 Min:     0 Max:   810 Err:
0 (0.00%)
Generate Summary Results = 69779 in 1357.4s=    51.4/s Avg:    38 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 11646 in 180.0s =    64.7/s Avg:    39 Min:     0 Max:   776 Err:
0 (0.00%)
Generate Summary Results = 81425 in 1537.4s=    53.0/s Avg:    38 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results + 11810 in 180.0s =    65.6/s Avg:    39 Min:     0 Max:   798 Err:
0 (0.00%)
Generate Summary Results = 93235 in 1717.3s=    54.3/s Avg:    38 Min:     0 Max:  1534 Err:
0 (0.00%)
Generate Summary Results +  5453 in 109.1s =    50.0/s Avg:    42 Min:     0 Max:   858 Err:
0 (0.00%)
Generate Summary Results = 98688 in 1826.4s=    54.0/s Avg:    39 Min:     0 Max:  1534 Err:
0 (0.00%)
Tidying up ...    @ Wed Oct 28 01:43:49 CDT 2009 (1256712229128)
... end of run
```

This output will be updated every 3 minutes, showing the most recent activity as well as a summary for the whole test.

Result Logs

As above, there will be output on the command line and log files will be created for both the successful (`jmeter-results-setup.jtl`) and unsuccessful (`jmeter-assertions-setup.jtl`) results. These log files are saved in the JTL format (based on XML). There are a number of parsers which will generate graphs from these logs files. For more information, see the JMeter wiki page on Log Analysis.

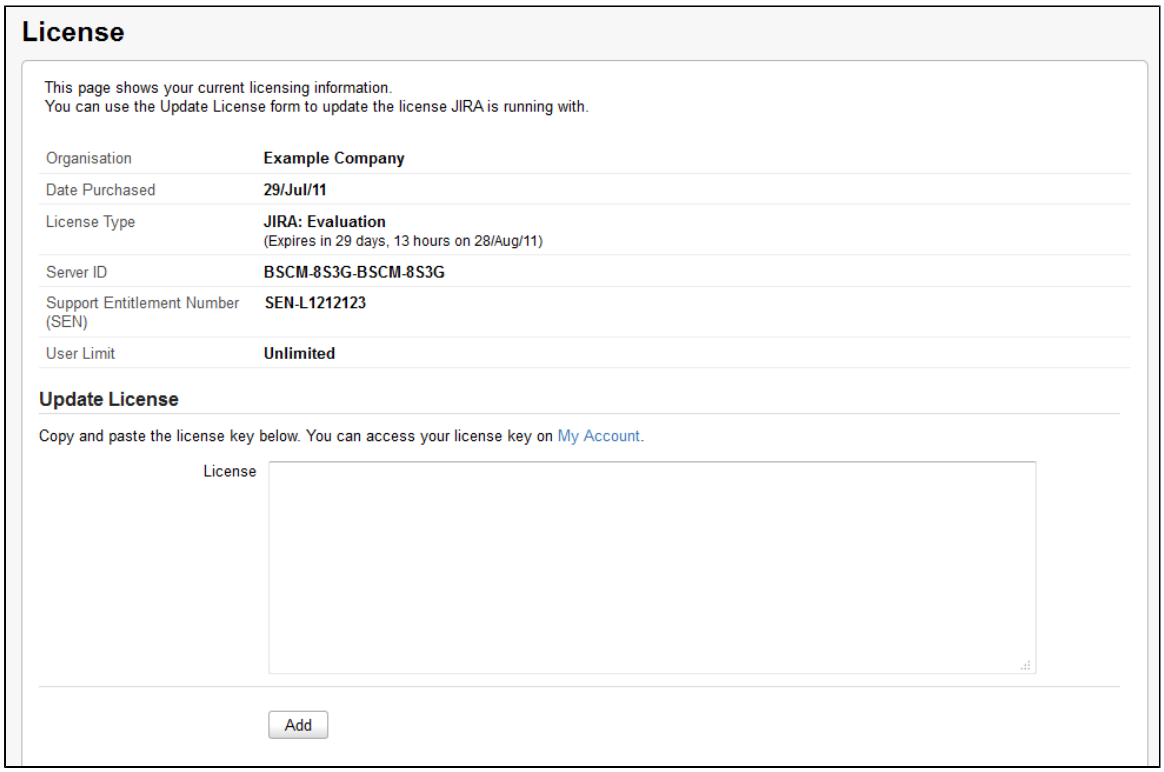
Finding the JIRA Support Entitlement Number (SEN)

There are three ways to find you Support Entitlement Number (SEN):

▼ Method 1: Check in the JIRA Administration Interface

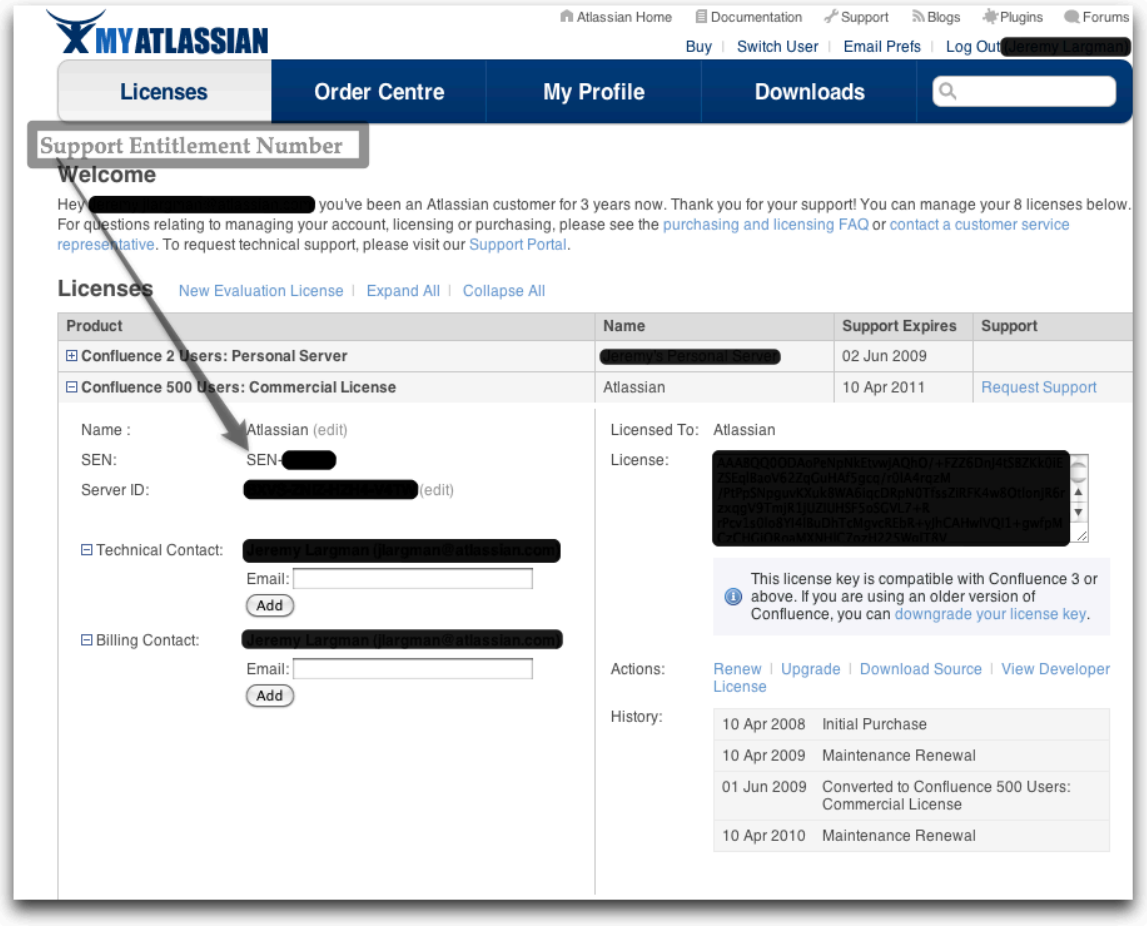
Tip: If you are using GreenHopper, the GreenHopper license appears above the JIRA one. The JIRA 'license details' link is towards the bottom of the list on the left hand side.

Access the JIRA license page, which indicates your Support Entitlement Number (SEN). See Updating your JIRA License Details for details on how to access this page.



▼ Method 2: Log into my.atlassian.com as the Account Holder or Technical Contact

Your Support Entitlement Number is available from the licenses page after logging in to http://my.atlassian.com:



▼ Method 3: Atlassian Invoice

Your Support Entitlement Number (SEN) also appears on the third page of your Atlassian Invoice.

See Finding Your Support Entitlement Number in the support space for more general information about how Atlassian Support uses this number.

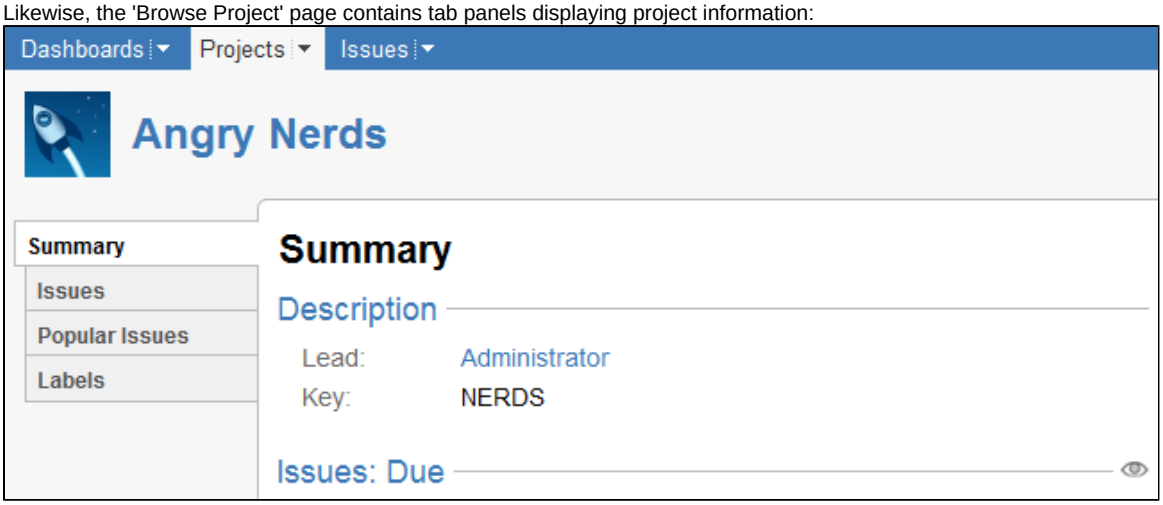
> *The information on this page does not apply to **JIRA OnDemand**.*

Appendix A - Extending JIRA

JIRA is very flexible, and has a number of extension points where JIRA's data can be queried or its functionality extended. You can also follow further instructions on Developing your Plugin using the Atlassian Plugin SDK.

JIRA Plugins: For information on installing and/or enabling existing plugins, please read the Managing JIRA's Plugins documentation.

Custom Field Types	JIRA comes with various custom field types defined. New types can be written and plugged into JIRA. See the Writing Custom Field Types Tutorial for more information.
User Formats	JIRA comes with many options to change the look and feel of features in the system. User formats are a feature that can be customised by plugins. You can write your own User Format plugin to change the display of user details in JIRA, e.g. display a profile picture. See the User Format Plugin Module for more information.
Gadgets	New gadgets can be created by writing an XML descriptor file, packaged as an Atlassian plugin. See Writing an Atlassian Gadget for more information.

Reports	JIRA comes with various reports built-in. Using the plugin system, new reports can be written, providing new ways of viewing and summarising JIRA's data.
Workflow functions and conditions	JIRA's issue workflow (states and state transitions an issue can go through) can be customized through the web interface (see the workflow documentation. The workflow engine (OSWorkflow) provides hooks where you can plug in your own behaviour: • Run arbitrary Java when a certain transition occurs, via post-functions • Limit visibility of transitions to certain users, via conditions • Validate input on transition screens (eg. in comments), via validators. See the guide to creating custom workflow elements for how to write your own workflow post-functions, conditions and validators. Once written, these can be packaged as plugins and reused.
Issue and Project Tabs	One the 'View Issue' page, some issue information (comments, change history) is displayed in tabs: Likewise, the 'Browse Project' page contains tab panels displaying project information:  By writing a plugin, you can add new issue or project tab panels to JIRA. For instance, you may wish to display project/issue data pulled in from an external source. This is how the JIRA Subversion plugin works. See the plugin guide for more information on writing these plugin types.
Listeners	JIRA has a complete event subsystem which fires events whenever anything happens. For example an `ISSUE_CREATED` event is fired whenever an issue is created. A listener is just a class which implements a `JiraListener` interface and is called whenever events occur in JIRA. Using those events, you can then perform any action you want. For example the email sent by JIRA is driven by the `MailListener`. This is useful when you want to drive or affect external systems from events which occur within JIRA - usually used to *push* data into outside systems. For more information, read the listeners documentation.
Services	Services are classes which implement the `JiraService` interface. When installed, you specify an update period and JIRA will call the `run()` method of your service periodically. A sample service is provided called `POPCommentService`. This service checks a particular POP mailbox periodically and if it finds messages, tries to extract an issue key from the subject. If the subject contains a key, the body of the mail is added as a comment to the message. Services are useful when you want to periodically *pull* data into JIRA from outside systems. For more information, see the services guide.
SOAP and XML-RPC remote interfaces	JIRA has a growing SOAP and XML-RPC interface. This enables you to drive JIRA automatically from external systems. For example you can have a Java program, Perl script or C# client add issues to JIRA. See the JIRA RPC overview for general information. For building RPC clients, check out the SOAP client tutorial and XML-RPC client tutorial. New RPC endpoints can also be added to JIRA as plugins - see RPC Endpoint Plugins.
Java	JIRA has a full set of Java APIs that can be used to update information with in JIRA. You can view the API here. JIRA commercial customers get full access to the JIRA source (see bottom of the downloads page), so you can modify JIRA itself if necessary. See the Building JIRA from Source page for more information.

Managing JIRA's Plugins

About Plugins

A plugin is a .jar file (containing code, resources and a configuration file) that can be installed into an Atlassian product to add new functionality or change the behaviour of existing features.

You can use plugins to customise and extend the functionality of JIRA in a variety of ways, including:

- creating new dashboard gadgets
- creating new reports
- creating new types of custom fields
- customising workflow
- modifying the availability of 'issue operations' links ('Create Issue', etc)

JIRA ships with a number of preinstalled plugins, and many more are available for download from the Atlassian Plugin Exchange. You can also create your own plugins. Please visit the JIRA Developer Documentation for information on building plugins.

Installed JIRA plugins can be enabled or disabled using the Universal Plugin (bundled with JIRA) as described below.

Every plugin is made up of one or more *plugin modules*. A single plugin may do many things, and each module represents a single function of the plugin. Individual modules can be enabled or disabled, although this is not recommended as modules can often depend on each other. To do so safely would require knowledge of the module interdependencies.

On this page:

- About Plugins
  - About 'Version 1' Plugins and 'Version 2' Plugins
  - How do I tell if a Plugin is 'Version 1' or 'Version 2'?
- Installing a JIRA Plugin
  - Installing a plugin from the Atlassian Plugin Exchange
  - Installing your own plugin
  - Installing a plugin manually
- Viewing Installed Plugins
- Updating a JIRA Plugin
- Disabling a JIRA Plugin
- Enabling a JIRA Plugin
- Disabling/Enabling all User Installed Plugins (Safe Mode)
- Uninstalling a JIRA Plugin
  - Uninstalling a 'Version 1' plugin manually
- Configuring a JIRA Plugin

*Some functionality described on this page is restricted in **JIRA OnDemand**.*

About 'Version 1' Plugins and 'Version 2' Plugins

There are two types of Atlassian plugins:

- **'Version 1' Plugins** — These plugins are compatible with the 'old' plugin framework that was used in JIRA prior to version 4.0, and also with the 'new' plugin framework used in JIRA version 4.0 and later. 'Version 1' plugins are installed into your JIRA Installation Directory under `atlassian-jira/WEB-INF/lib/`. **'Version 1' plugins must be installed manually** (see below) **and uninstalled manually** (see below).
- **'Version 2' Plugins** — These plugins are compatible with the 'new' plugin framework used in JIRA version 4.0 and later. **'Version 2' plugins are installed automatically, using the JIRA administration interface** (see below). They are installed into your JIRA Home Directory under `plugins/installed-plugins/`. 'Version 2' plugins are deployed in an internal OSGi container to isolate the plugin from the core code and other plugins, and to allow sophisticated dependency management. 'Version 2' plugins have to be specifically declared as such, using the `plugins-version="2"` attribute in the plugin's configuration file (`atlassian-plugin.xml`).

How do I tell if a Plugin is 'Version 1' or 'Version 2'?

Go to https://plugins.atlassian.com and look at the plugin's details. The '**Plugin System**' field will show 'ONE' for Version 1 plugins and 'TWO' for Version 2 plugins.

Installing a JIRA Plugin

Installing a plugin from the Atlassian Plugin Exchange

To access the Universal Plugin Manager in JIRA:

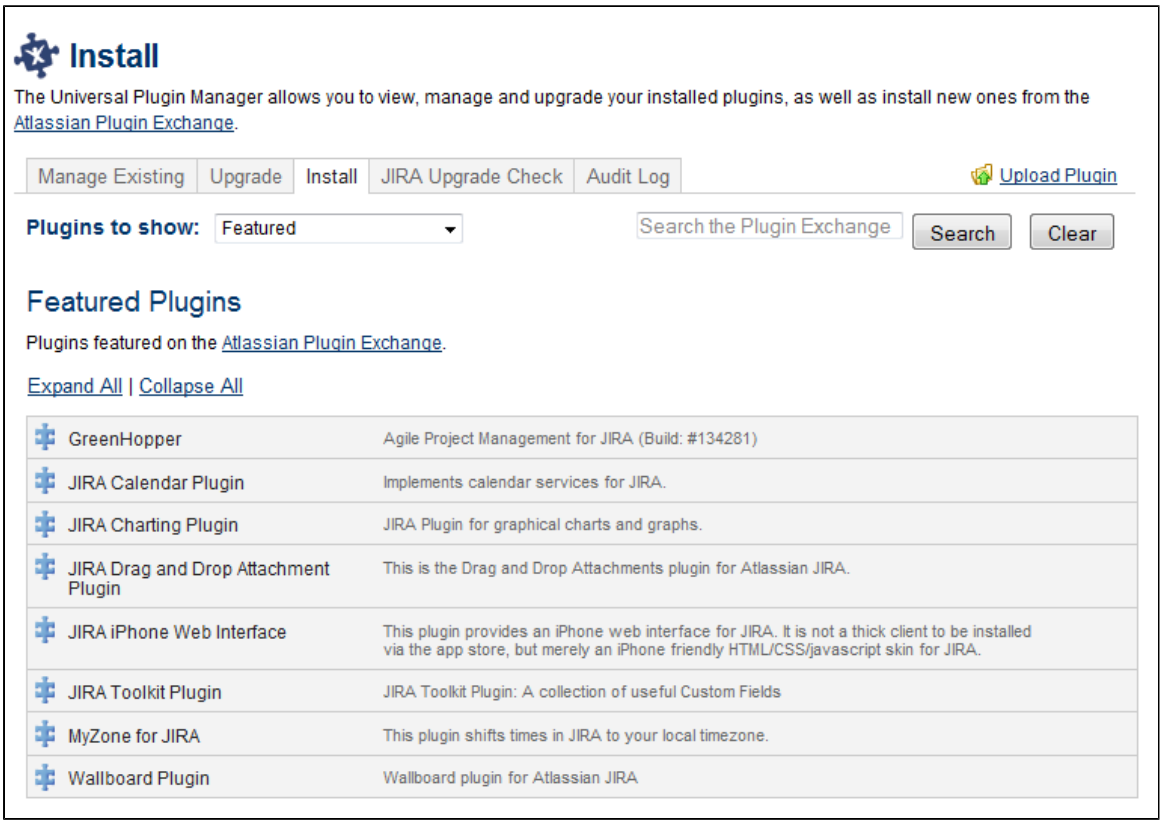
1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.

Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To find and add a plugin from the Atlassian Plugin Exchange:

1. Click the '**Install**' tab in the UPM. You will see a list of featured plugins.
2. Search for your plugin as follows:
   - Enter some keywords that describe the plugin in the '**Search the Plugin Exchange**' search box and press 'Enter'.
   - Alternatively, browse to the desired plugin in the list. You can choose '**Featured**', '**Popular**', '**Supported**' *(by Atlassian)* or '**All available**' from the '**Plugins to show**' dropdown to see a different list of plugins.
3. Click the '**Install**' button for the desired plugin to add it to your application. A confirmation message and the plugin details will appear when the plugin is installed successfully.
   Note: You may need to restart your application for your change to take effect. The Universal Plugin Manager will inform you if this is the case.
   Note: Not all plugins can be automatically installed. Some required manual installation. These plugins will have a 'Download' button instead of an 'Install' button. In these cases, you should read and follow the plugin's installation instructions.

Screenshot: Finding a new plugin from the Atlassian Plugin Exchange



Installing your own plugin

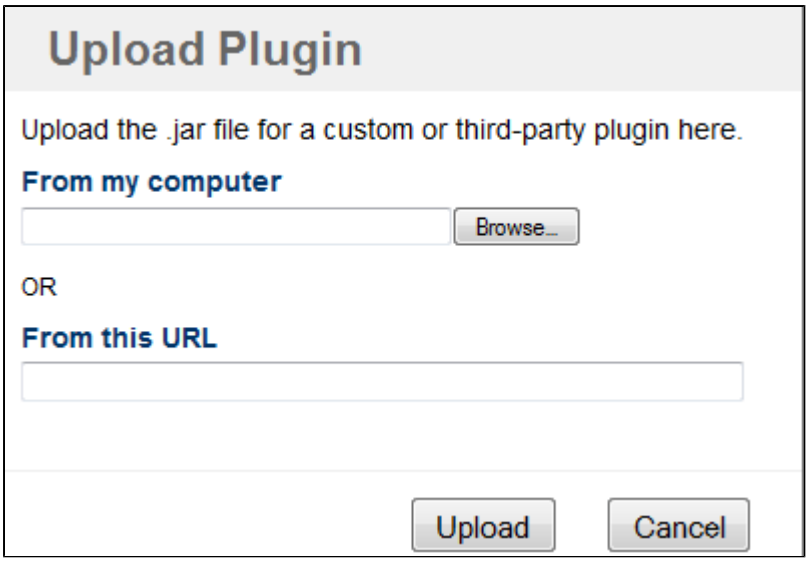
To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To upload your own plugin:

1. Click the '**Install**' tab in the UPM. You will see a list of featured plugins.
2. Click the '**Upload Plugin**' link. The 'Upload Plugin' window will appear.
3. Specify the location of your plugin:
   - If the plugin you want to install is on your computer, use the '**Browse**' dialogue to choose the plugin JAR file.
   - If you want to install a plugin from a remote location, enter the URL of the plugin JAR file in the '**From this URL**' text box.
4. Click the '**Upload**' button to upload and enable your plugin. A confirmation message will appear when the plugin is successfully installed.
   Note: You may need to restart your application for your change to take effect. The Universal Plugin Manager will inform you if this is the case.

Screenshot: Uploading a new plugin



Installing a plugin manually

You will need to install plugins manually if:

- you are installing a '**Version 1**' plugin; or
- your JIRA server does not have access to the internet (in which case the Universal Plugin Manager will be unable to access http://plugins.atlassian.com).

Once you have downloaded or created your plugin jar file, follow these steps to install it in your JIRA instance:

1. Shut down JIRA.
2. Copy your plugin jar (e.g. '**EXAMPLE_PLUGIN.jar**') into the appropriate plugins directory:
   - If you are installing a '**Version 1**' plugin, copy the jar into your JIRA installation directory under `atlassian-jira/WEB-INF/lib/`.
   - If you are installing a '**Version 2**' (OSGi) plugin, copy the jar into your JIRA home directory under `plugins/installed-plugins/`.

     To find out whether a plugin is Version 1 or Version 2, check the Plugin Exchange. Note that plugins have different versions, and older builds of plugins may be **Version 1**.
3. Start up JIRA.
4. Go to **'Administration' > 'Plugins' > 'Plugins'**. Your plugin (e.g. '**EXAMPLE_PLUGIN**') will be listed and enabled.

If you copy the JIRA jar file of a 'Version 1' plugin into the installation directory for 'Version 2' plugins (or vice versa), JIRA provides a warning, indicating that the plugin has been installed into the wrong directory.

However, there may be a very small number of 'Version 1' plugins (such as the JIRA RPC Plugin) which, when accidentally installed into the 'Version 2' plugins directory (`<JIRA home directory>/plugins/installed-plugins/`), will not generate this warning. Nevertheless, the plugin will not function correctly and the JIRA administration 'Plugins' page typically indicates that this plugin has zero modules.

To rectify this problem, reinstall the plugin jar to the correct directory by:

1. Shutting down JIRA,
2. Moving the plugin jar file to `<JIRA installation directory>/atlassian-jira/WEB-INF/lib/`
3. Restarting JIRA.

Viewing Installed Plugins

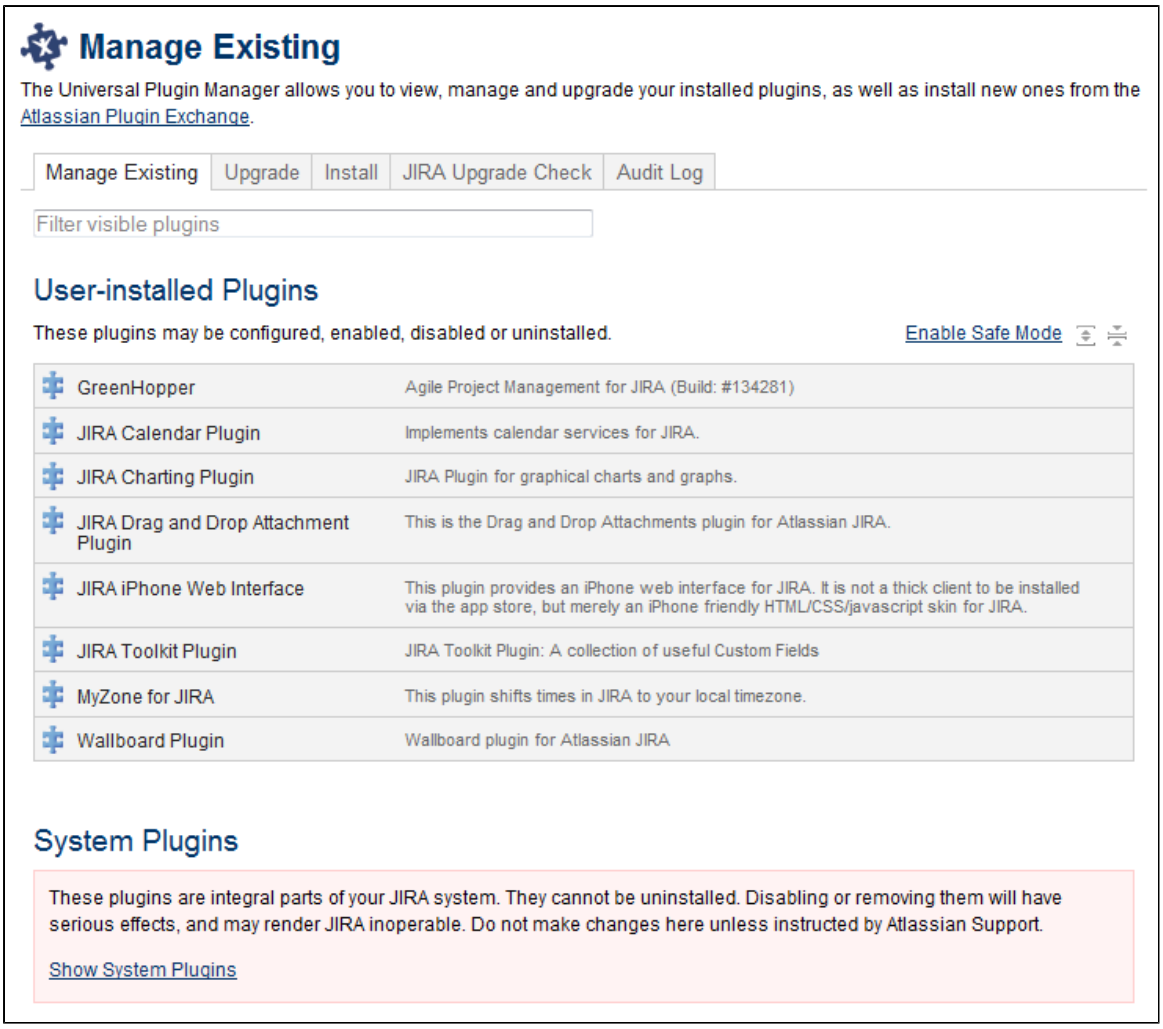
To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.

   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

Screenshot: Finding installed JIRA plugins, using the Universal Plugin Manager

Some tabs will not be visible if you do not have the JIRA System Administrators global permission



Updating a JIRA Plugin

To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To upgrade a plugin:

1. Click the '**Upgrade**' tab. The plugin upgrades page will appear.
   - If there is a later version of a plugin that you have already installed, this page will show the latest **compatible** version of the plugin.
   - You can click the plugin name to expand the row and see more information about the plugin.
   - You can filter your list by entering keywords in the 'Filter plugins' text box.
2. Click the '**Upgrade Now**' button next to the relevant plugin to update it to the plugin version shown.

To upgrade all available plugins:

1. Click the '**Upgrade**' tab. The plugin upgrades page will appear.
   - If there is a later version of a plugin that you have already installed, this page will show the latest **compatible** version of the plugin.
   - You can click the plugin name to expand the row and see more information about the plugin.
   - You can filter your list by entering keywords in the 'Filter plugins' text box.
2. Click the '**Upgrade All**' button to update every plugin to the plugin versions shown.
   Note: Some plugins cannot be installed via the Universal Plugin Manager. You must install these plugins manually. These plugins will not be upgraded automatically.

Disabling a JIRA Plugin

To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA Administrators** global permission.

2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To disable a plugin:

1. Click the '**Manage Existing**' tab. You will see a list of the plugins installed in your application. Enabled plugins will have this icon:
2. Locate the plugin that you want to disable and click the title to expand the plugin details section.
3. Click the '**Disable**' button.
4. Once a plugin has been disabled, you *may* need to restart your application for your change to take effect. If so, you will see a message for the plugin, 'Disabled, requires restart'.
   Once the plugin is fully disabled, you will see an '**Enable**' link for the plugin.

Also note that JIRA can automatically disable plugins if they cause errors on startup or initialisation. This is usually caused by having an old version of the plugin that is not compatible with your current version of JIRA. If this occurs, it will be indicated on the 'Current Plugins' page (shown above). You will need to stop JIRA, install the newer version of the plugin and restart JIRA.

Enabling a JIRA Plugin

Note: newly installed plugins are enabled by default.

To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To enable a plugin:

1. Click the '**Manage Existing**' tab. You will see a list of the plugins installed in your application. Disabled plugins will have this icon:
2. Locate the plugin that you want to enable and click the title to expand the plugin details section.
3. Click the '**Enable**' button.
4. Once a plugin has been enabled, you *may* need to restart your application for your change to take effect. If so, you will see a message for the plugin, 'Enabled, requires restart'.
   Once the plugin is fully disabled, you will see a '**Disable**' link for the plugin.

Disabling/Enabling all User Installed Plugins (Safe Mode)

Running JIRA in Safe Mode disables all user installed plugins at once. All plugins that were disabled when you entered Safe Mode will be re-enabled when you exit Safe Mode.

To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To enable Safe Mode:

1. Click the '**Manage Existing**' tab. You will see a list of the plugins installed in your application.
2. Click the '**Enable Safe Mode**' button.
3. Click the '**Continue**' button in the confirmation window. All user installed plugins will be disabled and your application will now be running in 'Safe Mode'.
4. You can now make changes to your installed plugins as desired. For example, you may want to enable/disable specific plugins or plugin modules.
5. Exit safe mode by clicking one of the links in the Safe Mode banner:
   - Click '**Exit Safe Mode and restore the previous configuration**' to restore your plugin configuration to its state before you entered Safe Mode.
   - Click '**Exit Safe Mode and keep the current configuration**' to keep all changes made to your plugin configuration during Safe Mode.

Uninstalling a JIRA Plugin

Note: If you only want to temporarily remove a plugin, you may wish to disable it instead of uninstalling it.

To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To uninstall a plugin:

1. Click the '**Manage Existing**' tab. You will see a list of the plugins installed in your application.
2. Click the name of the plugin that you wish to uninstall. The plugin details will appear.
3. Click the '**Uninstall**' button. The information summary will display an 'Uninstalling' message and the plugin will be uninstalled from your application.

Uninstalling a 'Version 1' plugin manually

1. Shut down JIRA.
2. Remove your plugin jar (e.g. 'EXAMPLE_PLUGIN.jar') from the appropriate plugins directory:
    - If you are removing a 'Version 1' plugin, remove the jar from your JIRA installation directory under atlassian-jira/WEB-INF/lib/.
    - If you are removing a 'Version 2' (OSGi) plugin, remove the jar from your JIRA home directory under plugins/installed-plugins/.
3. Start up JIRA.
4. Go to 'Administration > Plugins'. Your plugin (e.g. 'EXAMPLE_PLUGIN') will no longer be listed.

Configuring a JIRA Plugin

A number of plugins have advanced configuration options, e.g. a plugin may contain multiple 'modules' that you can enable/disable. If you have one of these plugins installed on your application instance, you can view and update these configuration options via the **Universal Plugin Manager (UPM)**.

To access the Universal Plugin Manager in JIRA:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**

To configure a plugin:

1. Click the '**Manage Existing**' tab.
2. Locate the plugin that you want to configure and click its title. The plugin details section will expand.
3. Click the '**Configure**' link for that plugin.
    - The link will be disabled if the plugin is disabled.
    - If there is no 'Configure' link, then there are no advanced configuration options available for that plugin.
4. The advanced configuration options for the plugin will appear. Update the configuration settings as desired and save your changes. **Note:** The advanced configuration screens are provided by the *plugin*. If you encounter any problems after you click the '**Configure**' link, the plugin is responsible for the issue, not the Universal Plugin Manager.

Listeners

Listeners are unique to JIRA, and a very powerful way to extend it.

JIRA has a complete event subsystem which fires events whenever anything happens inside the application. For example an `ISSUE_CREATED` event is fired whenever an issue is created.

A Listener is a class that implements one of the Listener interfaces. It is then called whenever events occur in JIRA. Using those events, you can then perform any action you want. For example the email sent by JIRA is driven by the MailListener.

Listeners are most useful when you want to drive or affect external systems from events which occur within JIRA.

On this page:

- Listener Interfaces
- Example Listeners
- Registering a Listener
- Editing Listener Properties
- Removing a Listener
- Custom Events
- See Also

*The information on this page does not apply to **JIRA OnDemand**.*

Listener Interfaces

JIRA has the following concrete Listeners (which extend the base JiraListener interface):

com.atlassian.jira.event. **JiraListener**	The base interface which all other JIRA listener interfaces extend. Covers core listener properties like uniqueness, description, parameters etc. API doc
com.atlassian.jira.event.issue. **IssueEventListener**	The main listener interface in JIRA, used whenever anything happens to an issue. API doc
com.atlassian.jira.event.user. **UserEventListener**	This listener is called whenever anything happens to a user within JIRA. API doc

Example Listeners

The examples provided may be freely used and modified for use in your own environment. The source of all examples is available and should give you good overview of how simple it is to write your own listeners. Both example listeners are included with JIRA 2.1, and both implement `UserEventListener` and `IssueEventListener`.

- **DebugListener** (source) — This is a very simple listener that prints events and their content to System.out whenever they are received. To test this listener, add a listener with the class `com.atlassian.jira.event.listeners.DebugListener` .
- **MailListener** (source) — This listener is how mail notifications are currently sent from within JIRA, and a good example of a more complex listener. It basically listens for events, and turns them into email notifications using Velocity templates to generate the mail bodies.
  This listener is usually always turned on in JIRA — see Email Notifications for more details. If you want to write more complex or more specific notifications, you can disable the internal MailListener and add your own.

Other examples of useful tasks that can be accomplished with listeners are:

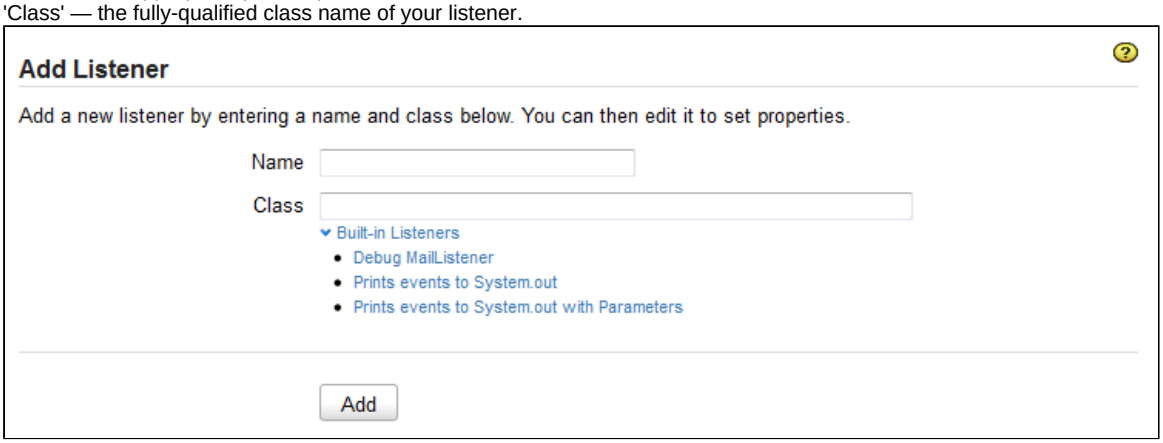
- **Send SMS or IM notifications** — A listener could easily send notifications for various events via SMS or instant messenger (e.g. ICQ or AIM) - or anywhere that you have a Java library to send messages.
- **Group notifications** — A listener could notify certain groups of issue changes, depending on the content of the issue. For example any issue containing "windows" in the environment could notify your "windows-developers" group.

Registering a Listener

For custom-written listener classes, make sure your listener class is in the classpath where JIRA can see it — the best locations are usually the `<jira-application-dir>/WEB-INF/classes` or `<jira-application-dir>/WEB-INF/lib` subdirectories within of your JIRA Installation Directory (as JAR files).

To register a listener:

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Advanced' > 'Listeners'** (tab) to open the 'Listeners' page.
   Keyboard shortcut: **'g' + 'g' +** type **'lis'**
3. In the 'Add Listener' form at the bottom of the page, complete the following fields:
   - 'Name' — an appropriately descriptive name for the listener.
   - 'Class' — the fully-qualified class name of your listener.



   To use one of JIRA's built-in listener classes, first click the '**Built-in Listeners**' link to expand the list of listener classes and then click the name of the specific class in the list. The fully-qualified class name of the built-in listener will be added to the 'Class' field.
4. Click the '**Add**' button and the listener will now be added to the list of listeners above.

Editing Listener Properties

If your listener accepts parameters or properties, you can edit these by clicking the '**Edit**' link associated with your listener (on the 'Listeners' page in JIRA's Administration area).

When defining your own Listener, there is a method `getAcceptedParams` to overload for defining the parameter names, pass as an array

of String objects. The `DebugParamListener` class is an example of doing this.

Removing a Listener

To remove a listener, click the '**Delete**' link associated with that listener (on the 'Listeners' page in JIRA's Administration area).

Custom Events

With the ability to add custom events to JIRA, the Listener must be updated to deal with the event as appropriate. This is possible by providing an implementation for the method `customEvent(IssueEvent event)` in the Listener. For example, the MailListener implementation passes the custom event on for notification processing. The DebugListener logs that the custom event has been fired.

See Also

- Plugin Tutorial - Writing event listeners with the atlassian-event library — this describes how to write listeners using the Atlassian Events library (see JIRA-specific Atlassian Events), rather than the JIRA Listener Events described above.

Services

A service is a class that runs periodically within JIRA. Since a service runs inside JIRA, it has the ability to use all of the JIRA API — and, as it is written in Java, it can use any Java libraries.

Services are useful because they enable you to integrate with external systems by pulling data into JIRA periodically. JIRA comes with a number of pre-written services, and custom services can be written and plugged in at runtime. If you want a service to perform typical operations on JIRA issues (eg. close a list of issues meeting certain criteria), then the Jelly Service can be configured to run a custom Jelly script.

Writing a new service?

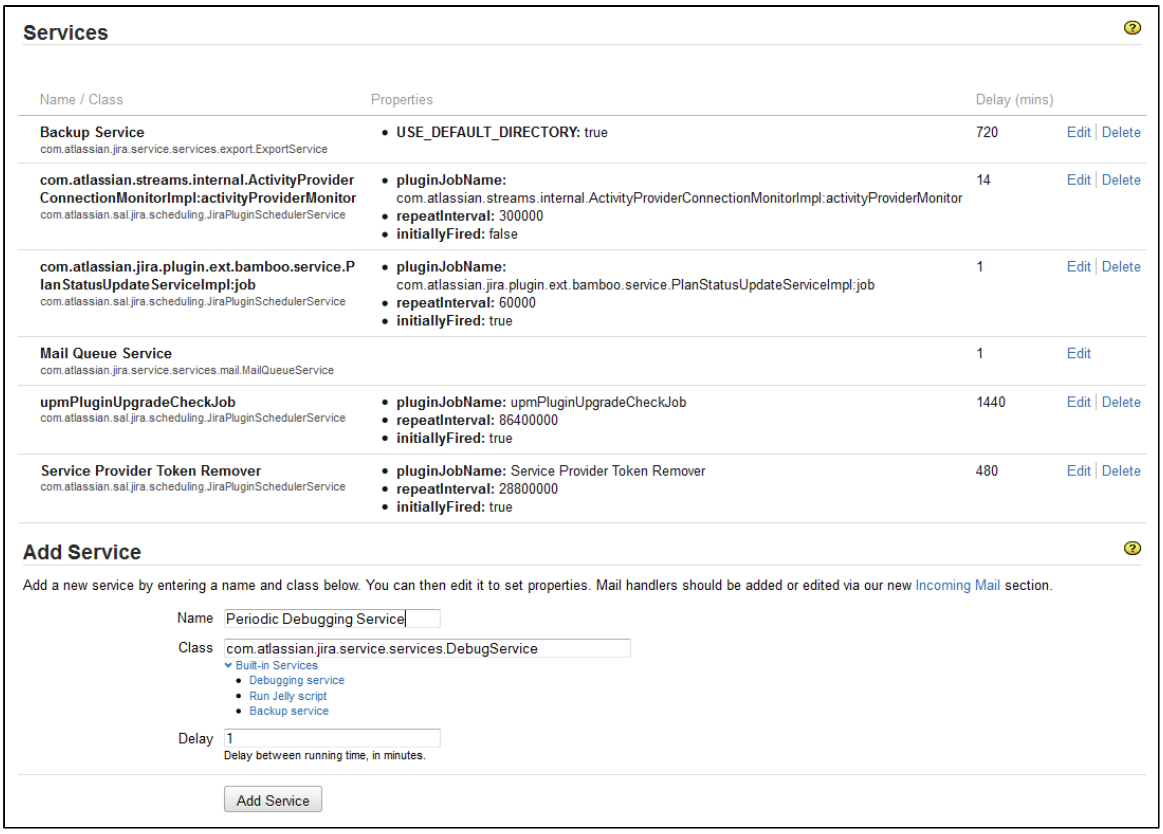
If you are not extending a built-in JIRA service, you should strongly consider writing your new service using the SAL API. Please see our Plugin Tutorial - Scheduling Events via SAL for more information.

Registering a service

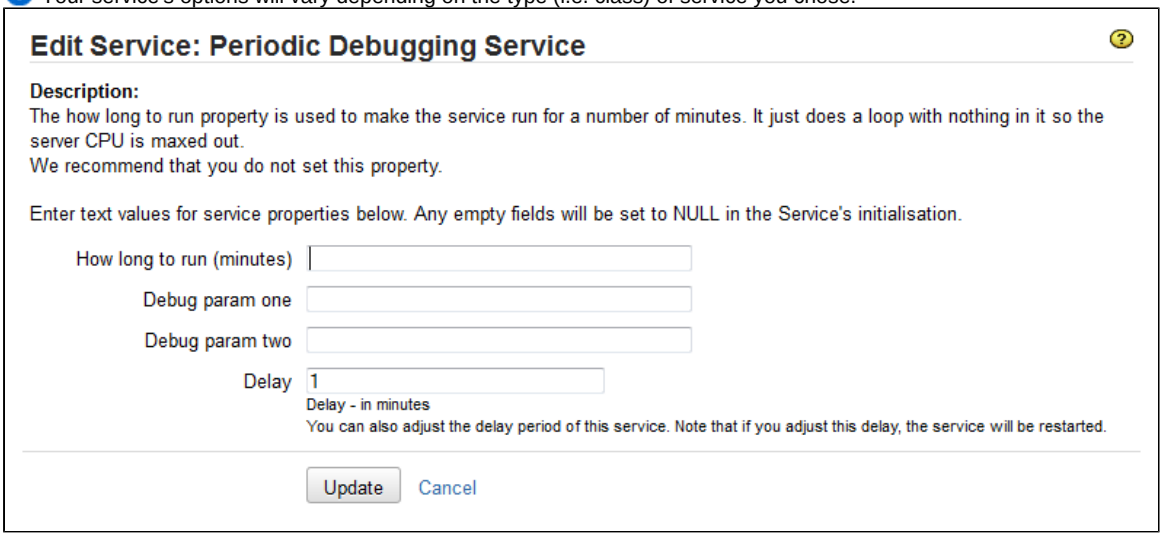
For custom-written services, make sure your service class is in the classpath where JIRA can see it — the best locations are usually the `<jira-application-dir>/WEB-INF/classes` or `<jira-application-dir>/WEB-INF/lib` subdirectories within of your JIRA Installation Directory (as JAR files).

To set up a JIRA service:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **Administration > System > Advanced > Services** (tab) to open the **Services** page.
   Keyboard shortcut: **g + g +** start typing **services**
3. In the **Add Service** form at the bottom of the page, complete the following fields:
   - **Name** — a descriptive name for this service.
   - **Class** — the fully-qualified class name of your service. This is likely to have the form `com.atlassian.jira.service.services.`*`type.Type`*`Service`
     See Sample services for provided service class names.
     To use one of JIRA's built-in service classes, first click the **Built-in Services** link to expand the list of service classes and then click the name of the specific class in the list. The fully-qualified class name of the built-in service will be added to the **Class** field.
   - **Delay** — the delay (in minutes) between service runs.
     For example, to add a debugging service, click the **Built-in Services** link followed by the **Debugging Service** link:



4. After completing the fields on the **Add Service** form, click the **Add Service** button. This opens the **Edit Service** page, where you can configure your new service's options.
   Your service's options will vary depending on the type (i.e. class) of service you chose.



5. After completing the remaining options on the **Edit Service** page, click the **Update** button to save your new service's options.

On this page:

- Registering a service
- Editing service properties
- Removing a service
- Built-in services
- Custom services

*The information on this page does not apply to **JIRA OnDemand**.*

Editing service properties

To edit a service's properties:

1. Log in as a user with the **JIRA System Administrators** global permission.

2. Select **Administration > System > Advanced > Services** (tab) to open the **Services** page.
   Keyboard shortcut: **g + g +** start typing **services**
3. Click the **Edit** link associated with the service whose properties you wish to edit.

For example, to change the interval at which email is sent from JIRA, edit the **Mail Queue Service** and change the **Delay** from the default value of 1 minute.

Removing a service

To remove a service:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Select **Administration > System > Advanced > Services** (tab) to open the **Services** page.
   Keyboard shortcut: **g + g +** start typing **services**
3. Click the **Delete** link associated with the service you wish to remove.

Built-in services

JIRA has some useful services out of the box, which may be used as-is or modified for use in your own environment. The source code for all built-in services is available and should give you a good overview of how simple it is to write your own services. All built-in services are included with JIRA and need only be configured to be used.

Export service

The Export Service is useful for periodically backing up JIRA. It exports all data from JIRA every time it is run, into a directory supplied as a parameter. The export files are timestamped, thus the service can act as a backup system.

To test this service, add a service with the class **com.atlassian.jira.service.services.export.ExportService**. JIRA sets up an ExportService in new JIRA installations (once the setup wizard has been completed). Hence, you may find you already have one.

You can find this class within the following directory of an expanded JIRA source archive (which can be downloaded by JIRA customers from https://my.atlassian.com):
`<source-installation-directory>/jira-project/jira-components/jira-core/src/main/java/com/atlassian/jira/`

Jelly service

Jelly is a scripting language which allows tasks in JIRA to be automated. The Jelly Service periodically runs a Jelly script. For example, you could use this to periodically run a search request, loop through the results and add a comment, change the issue state (see the Jelly examples).

If you are considering writing a custom service, often a periodically invoked Jelly script may be an easier alternative.

You can find this class within the following directory of an expanded JIRA source archive (which can be downloaded by JIRA customers from https://my.atlassian.com):
`<source-installation-directory>/jira-project/jira-components/jira-core/src/main/java/com/atlassian/jira/`

Mail handler services

JIRA mail handler services are not configurable through JIRA's **Services** page (with the exception of being able to be removed). For more information about configuring a mail handler in JIRA, including the creation of custom mail handlers, please refer to Creating Issues and Comments from Email.

Custom services

If you are a JIRA developer who wishes to write your own JIRA service, please note that JIRA Service classes must all extend com.atlassian.jira.service.JiraService. Most do so by extending com.atlassian.jira.service.AbstractService or some more specialised subclass.

Jelly Tags

Jelly is a scripting and templating language from Apache's Jakarta project. It is similar to Ant, in that scripts are XML, and each tag maps to a Java class, but has a more sophisticated internal pipeline model for tag interaction, much like JSP taglibs. See the Jelly website for more details.

JIRA comes with a number of Jelly tags implementing core operations in JIRA. This provides a scriptable interface to JIRA. There are many possible uses for JIRA Jelly tags, the most common being importing data into JIRA from other systems, and automating common administrative tasks (see the examples below).

Enabling Jelly

JIRA's Jelly support is **disabled** by default, as Jelly, in principle, allows running arbitrary Java code on the server under the Tomcat account. In some environments this may be considered a **security risk**, depending on who is allowed to configure and run Jelly scripts (a **'JIRA System Administrators'** permission is required). We recommend to use Jelly only when you absolutely cannot do without it and disable

Jelly support when you do not need it any more.

To enable Jelly, set the **jira.jelly.on** system property when starting your application server. System properties are set with parameters to the `java` command, e.g. `java -Djira.jelly.on=true ...` (You can set this parameter in the setenv.sh file in your /bin folder)

How to set this property depends on your application server. For example, set the environment variable **`JAVA_OPTS=-Djira.jelly.on=true`**, or when running JIRA as a service, set the service JVM parameter.

Running a Jelly script

To run a Jelly script once:

1. Log in as a user with the **'JIRA System Administrators'** global permission.
2. Select **'Administration' > 'System' > 'Advanced' > 'Jelly Runner'** (tab) to open the 'Jelly Runner' page.
   ✅ Keyboard shortcut: **'g' + 'g' +** type **'jel'**
3. Paste your Jelly script into the text area.

To run a Jelly script periodically:

- Configure a service with the following class: `com.atlassian.jira.jelly.service.JellyService`

On this page:

- Enabling Jelly
- Running a Jelly script
- Writing a Jelly script
  - jira:AddActorsToDefaultProjectRole
    - Examples
  - jira:AddActorsToProjectRole
    - Examples
  - jira:AddComment
    - Examples
  - jira:AddComponent
    - Examples
  - jira:AddFieldToScreen
    - Examples
  - jira:AddPermission
    - Examples
  - jira:AddUserToGroup
  - jira:AddVersion
    - Examples
  - jira:AssignIssue
    - Examples
  - jira:AttachFile
    - Examples
  - jira:CreateCustomField
    - Examples
  - jira:CreateGroup
    - Examples
  - jira:CreateIssue
    - Examples
  - jira:CreatePermissionScheme
  - jira:CreateProject
    - Examples
  - jira:CreateProjectRole
    - Examples
  - jira:CreateUser
    - Examples
  - jira:DeleteProjectRole
    - Examples
  - jira:GetDefaultRoleActors
    - Examples
  - jira:GetProjectRole
    - Examples
  - jira:GetProjectRoleActors
    - Examples
  - jira:IsProjectRoleNameUnique
    - Examples
  - jira:LinkIssue
    - Examples
  - jira:Login
    - Examples
  - jira:RemoveActorsFromDefaultProjectRole
    - Examples
  - jira:RemoveActorsFromProjectRole
    - Examples
  - jira:RemoveUser
    - Examples
  - jira:RunSearchRequest
    - Examples
  - jira:SelectComponentAssignees
    - Examples
  - jira:TransitionWorkflow
    - Examples
  - jira:UpdateProjectRole
    - Examples
- Beta Tags
- Sample scripts
  - Creating a new Project
  - For a list of projects, perform a project-specific operation.
  - Create a user, issue, and assign the issue to the user
  - Assigning and Starting Progress
  - Moving unreplied-to issues into an 'Inactive' state

*The information on this page does not apply to **JIRA OnDemand**.*

Writing a Jelly script

Scripts are generally of the form:

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<!--
    Add your own Jelly XML here
-->
</JiraJelly>
```

There are also a few extra tags that can be accessed by using the following outer tag, instead of the one above (these are tags that were formerly restricted to the now-defunct JIRA Enterprise edition):

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib">
<!--
    Add your own Jelly XML here
-->
</JiraJelly>
```

In addition to the JIRA tags, you can use tags from the email, http, soap, sql and core Jelly taglibs. More can be added by the user if necessary.

Many of JIRA's Jelly tags set *context variables,* so subsequent tags can refer to their output by dereferencing the context variable (e.g. `${jira.new.username}`). Other tags let you explicitly specify the name of a variable to store some output in, e.g., <jira:CreateUser> has `issueKeyVar` and `issueIdVar` parameters:

```
<jira:CreateIssue project-key="TP" summary="Issue One" issueKeyVar="issuekey"
issueIdVar="issueid"/>
          Raised issue ${issuekey} with id ${issueid}
```

Note that the variable is only set **after** the tag is closed, not inside the tag.

Please Note: Due to this variable interpolation, if your text contains anything of the form `${something}` , you need to escape this as `$${something}` to prevent the 'variable' being expanded to a blank string.

When specifying the value of an attribute, note the following special characters must be escaped.

Special Character	Escaped equivalent
ampersand (&)	`&`
apostrophe or single quote (')	`'`
double quote (")	`"`
less than (<)	`<`
greater than (>)	`>`

The list of currently available tags:

- Enabling Jelly

- Running a Jelly script
- Writing a Jelly script
  - jira:AddActorsToDefaultProjectRole
    - Examples
  - jira:AddActorsToProjectRole
    - Examples
  - jira:AddComment
    - Examples
  - jira:AddComponent
    - Examples
  - jira:AddFieldToScreen
    - Examples
  - jira:AddPermission
    - Examples
  - jira:AddUserToGroup
  - jira:AddVersion
    - Examples
  - jira:AssignIssue
    - Examples
  - jira:AttachFile
    - Examples
  - jira:CreateCustomField
    - Examples
  - jira:CreateGroup
    - Examples
  - jira:CreateIssue
    - Examples
  - jira:CreatePermissionScheme
  - jira:CreateProject
    - Examples
  - jira:CreateProjectRole
    - Examples
  - jira:CreateUser
    - Examples
  - jira:DeleteProjectRole
    - Examples
  - jira:GetDefaultRoleActors
    - Examples
  - jira:GetProjectRole
    - Examples
  - jira:GetProjectRoleActors
    - Examples
  - jira:IsProjectRoleNameUnique
    - Examples
  - jira:LinkIssue
    - Examples
  - jira:Login
    - Examples
  - jira:RemoveActorsFromDefaultProjectRole
    - Examples
  - jira:RemoveActorsFromProjectRole
    - Examples
  - jira:RemoveUser
    - Examples
  - jira:RunSearchRequest
    - Examples
  - jira:SelectComponentAssignees
    - Examples
  - jira:TransitionWorkflow
    - Examples
  - jira:UpdateProjectRole
    - Examples
- Beta Tags
- Sample scripts
  - Creating a new Project
  - For a list of projects, perform a project-specific operation.
  - Create a user, issue, and assign the issue to the user
  - Assigning and Starting Progress
  - Moving unreplied-to issues into an 'Inactive' state

jira:AddActorsToDefaultProjectRole

This tag will add 'actors' to the default membership for a given project role. Actors can be defined as groups or users, i.e. you can add both users and groups to a project role.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		This is the id of the project role.
actors	string		A comma delimited list of either users or groups
actortype	string		This defines the type 'actor' you are sending to the tag. Currently this field can contain either 'atlassian-user-role-actor' for users, or 'atlassian-group-role-actor' for groups.

Examples

Adding a list of default users or groups to a project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:AddActorsToDefaultProjectRole projectroleid="1" actors="fred,admin,tom"
        actortype="atlassian-user-role-actor" />
</JiraJelly>
```

jira:AddActorsToProjectRole

This tag will add 'actors' to a given project role for a particular project. Actors can be defined as groups or users, ie you can add both users and groups to a project role.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		This is the id of the project role.
actors	string		This a comma delimited list of either user names or group names
actortype	string		This defines the 'actor' type. Currently this field can contain either 'atlassian-user-role-actor' for users, or 'atlassian-group-role-actor' for groups.
projectkey	string		This is the key of the project you wish to add users or groups to for the specified role.

Examples

Adding a list of users or groups to a project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:AddActorsToProjectRole projectroleid="1" actors="jira-administrators,jira-users"
        projectkey="MKY" actortype="atlassian-group-role-actor" />
</JiraJelly>
```

jira:AddComment

This function adds a comment to an Issue.

Attributes

Attribute Name	Type	Default Value	Description
issue-key	string		The issue to add the comment to (required).

commenter	string	Currently logged in user	Username of the user to make the comment (Must have browse and comment permissions).
comment	string		Comment to be added to the issue (required).
groupLevel	string	none	Name of group that can see this comment. NOTE: If this is specified you can not specify the roleLevel parameter.
roleLevel	string	none	Name or Id of Project Role that can see this comment. NOTE: If this is specified you can not specify the groupLevel parameter.
created	string	Current Date/Time	Date/Time the Comment was created in format yyyy-MM-dd hh:mm:ss.0
updated	string	Current Date/Time	Date/Time the Comment was last updated in format yyyy-MM-dd hh:mm:ss.0. This can be used if you are trying to import a comment with specific pre-existing values.
editedBy	string	Currently logged in user	Username of the user who last updated the comment. This can be used if you are trying to import a comment with specific pre-existing values.
tweakIssueUpdateDate	boolean	true	If an updated date is provided, the issue's updated date will be updated with that value. If the tweakIssueUpdateDate parameter is set to false, the issue's updated timestamp will be left untouched.

Examples

Create comment

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:AddComment comment="Issue comment" issue-key="ABC-1"
groupLevel="admin-group"/>
</JiraJelly>
```

Create Issue and Comment

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:CreateIssue project-key="TP" issueType="Bug" summary="Issue summary"
issueKeyVar="key"/>
  <jira:AddComment issue-key="${key}" comment="A comment on ${key}"/>
</JiraJelly>
```

jira:AddComponent

Adds a component to a project.

Attributes

Attribute Name	Type	Default Value	Description
project-key	string		The key of the project you want to add the component to (not required if nested inside atag).
name	string		Name of the component (required).
description	string		Description of the component.
componentLead	string		The username of the Component's lead. Leave blank for no lead.

Examples

Create Component

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:AddComponent project-key="ABC" name="Comp 1" description="Comp 1 description"/>
</JiraJelly>
```

Create Component in a Project

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateProject key="ABC" name="A Project" lead="logged-in-user">
<jira:AddComponent name="Comp 1"/>
</jira:CreateProject>
</JiraJelly>
```

Create Component with a Component Lead

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:AddComponent project-key="ABC" name="Comp 1" description="Comp 1 with lead"
componentLead="user-name"/>
</JiraJelly>
```

jira:AddFieldToScreen

Adds a field to a specific tab on a screen. Can also specify in which position to insert the field.

Attributes

Attribute Name	Type	Default Value	Description
ffldId	string		Field ID of the field to add (required). e.g. "description", "duedate", etc.
screen	string		Screen ID or Name (required). e.g. "1" or "Default Screen".
tab	string	0	Tab ID or Name. e.g. "0" or "Field Tab".
fieldPosition	int	last position	Position to insert the field into. Range of values is from 1 to the number of fields on the screen.

Examples

Add Fields to a Screen

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<!-- Adds 'description' field to the 'Field Tab' on 'Default Screen' -->
<jira:AddFieldToScreen fieldId="description" screen="Default Screen" tab="Field Tab"/>

<!-- Adds 'duedate' field to same screen as above. duedate is inserted in position 1 -->
<jira:AddFieldToScreen fieldId="duedate" screen="1" tab="0" fieldPosition="1"/>
</JiraJelly>
```

Create a new Customfield and Add it to a Screen

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateCustomField fieldType="cascadingselect"
fieldScope="issuetype"
fieldName="Issue cascadingselect Bug"
issueType="Bug"
description="Bank have requested Y2K fixes to be sent as an EBF."
searcher="cascadingselectsearcher"
customFieldIdVar="customField"
>
<jira:AddCustomFieldSelectValue value="Parent 1" />
<jira:AddCustomFieldSelectValue value="Parent 2">
<jira:AddCustomFieldSelectValue value="Child 1" />
<jira:AddCustomFieldSelectValue value="Child 2" />
</jira:AddCustomFieldSelectValue>
<jira:AddCustomFieldSelectValue value="Parent 3" />
</jira:CreateCustomField>

<jira:AddFieldToScreen screen="Default Screen" fieldId="${customField.getId()}"/>
</JiraJelly>
```

jira:AddPermission

Grants permissions within a permission scheme. Often nested within a JIRADOC:CreatePermissionScheme tag.

Attributes

Attribute Name	Type	Default Value	Description
schemeId	string		If not nested in a CreatePermissionScheme tag, specifies the scheme Id to add the permission to (0 is the default permission scheme).
permissions	required string		A comma-separated list of permissions to grant: **String** — **Permission** • Project — Administer projects • Browse — Browse projects • Create — Create issues • Edit — Edit issues • ScheduleIssue — Schedule issues • Move — Move issues • Assign — Assign issues • Assignable — Assignable user • Resolve — Resolve issues • Close — Close issues • ModifyReporter — Modify reporter • Comment — Add comments • CommentEditAll — Edit all comments • CommentEditOwn — Edit own comments • CommentDeleteAll — Delete all comments • CommentDeleteOwn — Delete own comments • Delete — Delete issues • Work — Work on issues • WorklogEditAll — Edit all worklogs • WorklogEditOwn — Edit own worklogs • WorklogDeleteOwn — Delete own worklogs • WorklogDeleteAll — Delete all worklogs • Link — Link issues • Attach — Create attachments • AttachDeleteAll — Delete all attachments • AttachDeleteOwn — Delete own attachments • ViewVersionControl — View version control • ViewVotersAndWatchers — View voters and watchers • ManageWatcherList — Manage watcher list • SetSecurity — Set issue security level

type	string		Type of recipient for the permission: • group • projectrole • user • lead • assignee • reporter • userCF • groupCF
group	string		If type is 'group' (or type is unspecified), specifies the group name to grant permissions to.
projectroleid	int		If type is 'projectrole', specifies the id of the projectrole to grant permissions to.
user	string		If type is 'user', specifies the user name to grant permissions to.
userCF	string		If type is 'userCF', specifies the id of a User custom field, e.g. 'customfield_10000', identifying the user to be granted the permission.
groupCF	string		If type is 'groupCF', specifies the id of a group-selecting custom field (e.g. a select-list with group names as values) whose members should be granted this permission. E.g. 'customfield_10000'.

Examples

Grant permissions to jira-users and jira-developers in a new permission scheme
(See also the JIRADOC:example scripts)

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:CreatePermissionScheme name="New Permission Scheme">
    <jira:AddPermission group="jira-users" permissions="Browse,Create,Comment,Attach"
type="group"/>
    <jira:AddPermission group="jira-developers"
permissions="Move,Assignable,Link,ViewVersionControl" type="group"/>
  </jira:CreatePermissionScheme>
</JiraJelly>
```

Grant issue reporters the ability to edit/delete their own issues, in a new permission scheme

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib">
  <jira:CreatePermissionScheme name="New Permission Scheme">
    <jira:AddPermission type="reporter" permissions="Delete, Edit"/>
  </jira:CreatePermissionScheme>
</JiraJelly>
```

Make projects using default permission scheme visible to certain users

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib">
  <jira:AddPermission schemeId="0" permissions="Browse" type="user" user="johnc"/>
  <jira:AddPermission schemeId="0" permissions="Browse" type="user" user="ebf"/>
</JiraJelly>
```

Granting a group selector custom field's members the ability to assign/be assigned the issue.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:AddPermission schemeId="10164" type="groupCF" groupCF="customfield_10000"
                      permissions="Assign,Assignable" />
</JiraJelly>
```

jira:AddUserToGroup

Makes a user a member of a Group. Adds the username and/or group name into the context if specified.

Attributes

Attribute Name	Type	Default Value	Description
username	string		Username to add to Group (required if not in atag).
group-name	string		Group to add User to (required if not in atag). Note: if the group has the 'JIRA System Administrators' global permission, and the logged-in user does not, an error message will be displayed and the operation will not succeed.

Username is set in the context if specified in the tag.Group name is set in the context if specified in the tag.h4. Examples

Add User to Group

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:AddUserToGroup username="new-user" group-name="new-group"/>
</JiraJelly>
```

Add New User to Group

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateUser username="new-user" password="password" confirm="password"
fullname="Full name" email="test@test.com">
<jira:AddUserToGroup group-name="new-group"/>
</jira:CreateUser>
</JiraJelly>
```

Add User to New Group

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateGroup group-name="new-group">
<jira:AddUserToGroup username="new-user"/>
</jira:CreateGroup>
</JiraJelly>
```

Add New User to New Group

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateUser username="new-user" password="password" confirm="password"
fullname="Full name" email="test@test.com"/>
<jira:CreateGroup group-name="new-group">
<jira:AddUserToGroup/>
</jira:CreateGroup>
</jira:CreateUser>
</JiraJelly>
```

jira:AddVersion

Adds a version to a project.
Attributes

Attribute Name	Type	Default Value	Description
project-key	string		The key of the project you want to add the component too (not required if nested inside atag).

name	string		Name of the version (required).
description	string		The description of the version.
releaseDate	string		The release date of the version.
schedule	string		Schedule of the version.

Examples

Create a Version

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:AddVersion project-key="ABC" name="Ver 1"/>
</JiraJelly>
```

Create a Version in a Project

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateProject key="ABC" name="A Project" lead="logged-in-user">
<jira:AddVersion name="Ver 1"/>
</jira:CreateProject>
</JiraJelly>
```

jira:AssignIssue

Assigns an issue to a user.

Attributes

Attribute Name	Type	Default Value	Description
key	string		Key of the issue to assign.
assignee	string		User to assign issue to.

Examples

Create and assign issue

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:CreateIssue project-key="TST" summary="My Issue summary" issueKeyVar="keyvar"/>
  <jira:AssignIssue key="${keyvar}" assignee="testuser"/>
</JiraJelly>
```

jira:AttachFile

Attaches a file to an issue.

Attributes

Attribute Name	Type	Default Value	Description
key	string		Key of the issue to attach the file to. (Required)
filepath	string		Path (on the server) of the file to attach. (Required)

option	string	add	Behaviour when a file with same name is already attached. (Optional). The options are: • **skip** — do not attach file if a file with this name is already attached. • **override** — overwrite existing attached file • **add** — add the file as another attachment
created	string	Current Date/Time	Date/Time the attachment was created, in format yyyy-MM-dd hh:mm:ss.0 (Optional)

Examples

Adding an attachment

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:AttachFile key="TST-1" filepath="/tmp/somefile" option="override"/>
</JiraJelly>
```

jira:CreateCustomField

The tag creates a new Custom Field. Only System custom fields can be added with Jelly tags.

Attributes

Attribute Name	Type	Default Value	Description
fieldType	string		Field type as appears as the key in the plugin descriptor
fieldScope	string		One of global, project or issuetype
fieldName	string		Name of custom field
projectKey	string		Key of the related project. Only valid for scope "project"
issueType	string		Issue type. Only valid for scope "issuetype"
description	string		Description of the field to be displayed when adding a value
searcher	string		A valid related custom field searcher
customFieldIdVar	string		The name of the variable to place the new custom field.

Examples

Create Cascading Custom Field
`jira:AddCustomFieldSelectValue` subtag can be used to add values for select lists. They can also be nested for Cascading Select Lists.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateCustomField fieldType="cascadingselect"
fieldScope="issuetype"
fieldName="Issue cascadingselect Bug"
issueType="Bug"
description="Bank have requested Y2K fixes to be sent as an EBF."
searcher="cascadingselectsearcher"
>
<jira:AddCustomFieldSelectValue value="Parent 1" />
<jira:AddCustomFieldSelectValue value="Parent 2">
<jira:AddCustomFieldSelectValue value="Child 1" />
<jira:AddCustomFieldSelectValue value="Child 2" />
</jira:AddCustomFieldSelectValue>
<jira:AddCustomFieldSelectValue value="Parent 3" />
</jira:CreateCustomField>
</JiraJelly>
```

jira:CreateGroup

Creates a Group in JIRA.

Attributes

Attribute Name	Type	Default Value	Description
group-name	string		Name of group to create (required).

Context Variables

Context Variable	Type	Description
jelly.group.name	string	Name of group being created.

Examples

Create Group

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateGroup group-name="new-group"/>
</JiraJelly>
```

jira:CreateIssue

This tag creates a new issue in JIRA and places the issue id in the context.

Attributes

Attribute Name	Type	Default Value	Description
project-key	string		Key of the project to add the issue to (required if not nested in atag).
issueType	string	First issue type	The string name of the Issue Type this issue should be created for (e.g. Bug).
summary	string		Summary of the issue being created (required).
priority	string	First priority	The string name of the Priority (e.g. Major).

components	string		The string name of the Component.
versions	string		The string name of the Affected Version.
fixVersions	string		The string name of the Fix For Version.
assignee	string		The username of the user to assign this issue to (logged in user requires the assign issue permission and user specified requires the assignable permission). Set to "-1" for Automatic assignment.
reporter	string	(see description)	The username of the user who is reporting this issue. The user is logged in and then the issue is created. The user is logged out again when the Create Issue tag closes. If the logged in user does not have Modify Reporter privilege, then the default value of this attribute is the username of the logged in user. If, however, the logged in user does have Modify Reporter privilege, there is not a default value, and this attribute is mandatory. See JRA-12984 for further explanation. (Broken? See JRA-5620.)
environment	string		Description of the environment.
description	string		Detailed description of the issue.
duedate	string		Due date of the issue. The format required is the current JIRA date format. **Note:** As the default JIRA date format is locale-specific (e.g. 12/Jan/05), you may wish to use the yyyy-mm-dd ISO format instead. To do this, set the following properties on JIRA's 'Advanced Settings' page: • `jira.date.picker.java.format` to value `yyyy-MM-dd` • `jira.date.picker.javascript.format` to value `%Y-%m-%e` See Changing the Due Date Input Format for more information about changing these values.
created	string	Current Date/Time	Date/Time the Issue was created in format yyyy-MM-dd hh:mm:ss.0
updated	string	Current Date/Time	Date/Time the Issue was updated in format yyyy-MM-dd hh:mm:ss.0
issueIdVar	string		The name of the variable to place the ID of the new Issue.
issueKeyVar	string		The name of the variable to place the Key of the new Issue.
duplicateSummary	string		Setting this attribute to 'ignore' will allow Issue with the same summary to be created.
security-level	string		Sets the security level of an issue. Value is the name of a level, e.g. 'Secret'.

Examples

Create Issue

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateIssue project-key="ABC" assignee="-1" summary="Issue summary">
<!-- other jelly tags -->
</jira:CreateIssue>
</JiraJelly>
```

Create Issue from Project
This example is more complicated as a permission scheme is required for the project before an issue can be created.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateProject key="ABC" name="A Project" lead="logged-in-user">
<jira:CreatePermissionScheme name="admin-scheme">
<jira:AddPermission permissions="Assignable,Browse,Create,Assign"
type="group"/>
<jira:SelectProjectScheme/>
</jira:CreatePermissionScheme>
<jira:CreateIssue summary="Issue summary">
<!-- other jelly tags -->
</jira:CreateIssue>
</jira:CreateProject>
</JiraJelly>
```

Create Issue with Custom Field values
Use the subtag `jira:AddCustomFieldValue`

Attribute Name	Type	Description
id	long	ID of the custom field with the customfield_ prefix
value	string	string representation of the custom field value. Note that this may be different to the displayed value (e.g. The project picker uses the project id as the String value but displays the project name)
key	string	Key is used for multi-dimensional data. Currently, only Cascading selects supports its use. Omit to specify the value of parent, use "1" as the value for child
name	String	**deprecated** Name of the custom field.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateIssue project-key="ABC" summary="Issue summary">
<jira:AddCustomFieldValue id="customfield_10000" value="field value"/>
<jira:AddCustomFieldValue name="Environment Select list" value="Windows XP"/>

<!-- For Cascading Selects : Note also that the value for cascading selects is the optionId-->
<jira:AddCustomFieldValue id="customfield_10001" value="Parent Option Id" />
<jira:AddCustomFieldValue id="customfield_10001" value="Child Option Id" key="1" />

<!-- For Version Pickers and Single Version Pickers : Note also that the value for version
pickers is the versionId-->
<jira:AddCustomFieldValue id="customfield_10002" value="Version Id"/>

<!-- For Multi Selects -->
<jira:AddCustomFieldValue id="customfield_10003" value="Value 1" />
<jira:AddCustomFieldValue id="customfield_10003" value="Value 2" />

<!-- For Multi User Pickers : Note also that the value for multi user pickers is a comma
separated list of users-->
<jira:AddCustomFieldValue id="customfield_10004" value="User 1,User 2" />

</jira:CreateIssue>
</JiraJelly>
```

Using the name attribute has been **deprecated.** While it will work in 3.0 its use is discouraged.

Note:

- To view the <customFieldId>,
  1. Navigate to Administration -> Issue Fields -> Custom Fields
  2. Hover your cursor over the "Configure" link of the custom field
  3. You can view the <customFieldId> in the status bar of your browser
- To view the "Parent Option Id" and "Child Option Id" for Cascading Select fields,
  1. Navigate to Administration -> Issue Fields -> Custom Fields -> Configure -> Edit Options -> Edit
  2. You can view the <selectedParentOptionId> ("Parent Option Id") and <selectedValue> ("Child Option Id") in the status bar of your browser

jira:CreatePermissionScheme

Creates a Permission Scheme

Attributes

Attribute Name	Type	Default Value	Description
name	required string		Name of the permission scheme.
description	string		Permission scheme description.

Context Variables

Context Variable	Type	Description
jelly.permission.scheme.id	string	Id of the created permission scheme

jira:CreateProject

This tag creates a new project in JIRA and places the project id in the context.

Attributes

Attribute Name	Type	Default Value	Description
key	string		The project key used to create Issue Keys (required).
name	string		The name of the project (required).
lead	string		The username of the user that is the project lead (required).
url	string		The URL of the site for this project.
description	string		The description of this project.

Context Variables

Context Variable	Type	Description
jelly.project.id	string	Id of the Project that was created.
jelly.project.key	string	Key of the Project that was created.

Examples

Create Project

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateProject key="ABC" name="A Project" lead="a-user">
<!-- other jelly tags -->
</jira:CreateProject>
</JiraJelly>
```

jira:CreateProjectRole

This tag will create a project role with the given name and description.

Attributes

Attribute Name	Type	Default Value	Description
name	string		The name for the project role you will be creating
description	string		The description for the project role you will be creating

Context Variables

Context Variable	Type	Description
jelly.role.id	Long	The id of the project role
jelly.role.name	string	The name of the project role
jelly.role.description	string	The description of the project role

Examples

Creating a new project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:CreateProjectRole name="lion-tamer" description="tames the lions">
        ${jelly.role.id} ${jelly.role.name} ${jelly.role.description}
    </jira:CreateProjectRole>
</JiraJelly>
```

jira:CreateUser

Creates a user in JIRA and places their username in the context.

Attributes

Attribute Name	Type	Default Value	Description
username	string		Username of the user being created (required).
password	string		User's password. If the password field is left blank, a random password will be auto-generated.
confirm	string		Confirmation of users password (required).
fullname	string		Descriptive name of the user (required).
email	string		Email address of the user (required).
sendEmail	boolean	false	If provided, specifies whether to send a confirmation email.

Context Variables

Context Variable	Type	Description
jelly.new.username	string	Username of the user being created.

Examples

Create User

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateUser username="new-user" password="password" confirm="password"
fullname="Full name" email="test@test.com"/>
</JiraJelly>
```

jira:DeleteProjectRole

This tag will delete the project role with the given id.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		The id of the project role you want to delete.
confirm	string		To delete the project role this value must be set to 'true'.

Examples

Deleting a project role from JIRA

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:DeleteProjectRole projectroleid="1" confirm="true" />
</JiraJelly>
```

jira:GetDefaultRoleActors

This tag will return a ProjectRoleActors object for a given project role for a particular project. This object carries the members of a project role, i.e. users and/or groups. To get the collection of users in this object, use the expression ${roleactors.users} where roleactors is the variable name of the object. For more information on the RoleActors object, consult the JIRA API.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		The id of the project role you want to query
var	string		The name of the variable you wish to have the returned role actors placed into

Examples

Returning a List of role actors and iterating over the users in each of these actors.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib" xmlns:core="jelly:core">
    <jira:GetDefaultRoleActors projectroleid="1" var="roleactors" >
        <core:forEach var="actor" items="${roleactors.users}">
            ${actor.name}
        </core:forEach>
    </jira:GetDefaultRoleActors>
</JiraJelly>
```

jira:GetProjectRole

This tag will return the project role with the given id.

Attributes

Attribute Name	Type	Default Value	Description

projectroleid	int		The id of the project role you want
var	string		The name of the variable you wish to have the project role assigned to

Examples

Returning a project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:GetProjectRole projectroleid="1" var="role" >
            ${role.name}
    </jira:GetProjectRole>
</JiraJelly>
```

jira:GetProjectRoleActors

This tag will return a ProjectRoleActors object for the given project role and project. This object is a placeholder for the internal members of a project role, i.e. users and/or groups. To get the collection of users in this object, use the expression ${roleactors.users} where roleactors is the variable name of the object. For more information on the RoleActors object, consult the JIRA API.

Attributes

Attribute Name	Type	Default Value	Description
projectkey	string		The key of the project you want to query
projectroleid	int		The id of the project role you want to query
var	string		The name of the variable you want the returned 'role actors' object assigned to

Examples

Return a list of users for a given 'Role Actors' object

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib" xmlns:core="jelly:core">
    <jira:GetProjectRoleActors projectkey="MKY" projectroleid="1" var="roleactors" >
        <core:forEach var="actor" items="${roleactors.users}">
            ${actor.name}
        </core:forEach>
    </jira:GetProjectRoleActors>
</JiraJelly>
```

jira:IsProjectRoleNameUnique

This tag will return 'true' or 'false' to let you know if there is already a project role with the given name.

Attributes

Attribute Name	Type	Default Value	Description
name	string		The name of the project role
var	string		The name of the variable you want the returned result assigned to.

Examples

Determining if a project role is unique.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:IsProjectRoleNameUnique name="unique name" var="isUnique" >
        ${isUnique}
    </jira:IsProjectRoleNameUnique>
</JiraJelly>
```

jira:LinkIssue

This tag creates a link from one issue to another issue.

Attributes

Attribute Name	Type	Default Value	Description
key	string		The key of the issue to link from (origin of link - required)
linkKey	string		The key of the issue to link to (destination of link - required)
linkDesc	string		linkDesc is taken from the 'Inward Description' or the 'Outward Description' of the link. (required)

Examples

Create a Link between two existing issues

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:LinkIssue key="TST-1" linkKey="TST-2" linkDesc="duplicates"/>
</JiraJelly>
```

Create two issues and link them

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:CreateIssue project-key="HSP" assignee="-1" summary="Issue summary 1" reporter="admin"
issueKeyVar="issuekey1"/>
<jira:CreateIssue project-key="NDT" assignee="-1" summary="Issue summary 2" reporter="admin"
issueKeyVar="issuekey2"/>
<jira:LinkIssue key="${issuekey1}" linkKey="${issuekey2}" linkDesc="duplicates"/>
</JiraJelly>
```

jira:Login

This tag logs a user into JIRA using the username and password provided. Use this tag when you are running the Jelly script in a manner in which you are not logged in (for example, if you are running a JellyService instead of using the Jelly Runner), or if you want to run the Jelly script as a different user to the one you are logged in as.

Attributes

Attribute Name	Type	Default Value	Description
username	string		Username of the user to log in.
password	string		Password of the user to log in.

Context Variables

Context Variable	Type	Description

jelly.user	User	User logged in.
jelly.username	string	Username of the User logged in.

Examples

Login a user in with username and password and set in context

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:Login username="misc-user" password="password">
<!-- other jelly tags -->
</jira:Login>
</JiraJelly>
```

jira:RemoveActorsFromDefaultProjectRole

This tag will remove a list of role actors (i.e. users and/or groups) from the default membership of a given project role.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		The id of the project role you wish to remove default actors from
actors	string		A comma delimited list of users or groups you wish to remove from the default project role
actortype	string		The type of 'actor' you are removing. Currently the available options are 'atlassian-group-role-actor' or 'atlassian-user-role-actor'

Examples

Removing a list of groups from a default project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:RemoveActorsFromDefaultProjectRole projectroleid="1"
        actors="jira-administrators, jira-users" actortype="atlassian-group-role-actor" />
</JiraJelly>
```

jira:RemoveActorsFromProjectRole

This tag will remove a list of role actors from a given project role for a given project.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		The id of the project role you wish to remove members from
actors	string		A comma delimited list of users or groups you wish to remove from the project role
projectkey	string		The key of the project the project role is associated with
actortype	string		The type of 'actor' you are working with. Currently the available options are 'atlassian-group-role-actor' or 'atlassian-user-role-actor'

Examples

Removing a list of groups from a project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:RemoveActorsFromProjectRole projectroleid="1"
        actors="jira-administrators, jira-users" projectkey="MKY"
        actortype="atlassian-group-role-actor" />
</JiraJelly>
```

jira:RemoveUser

Removes an existing JIRA user by their username

Attributes

Attribute Name	Type	Default Value	Description
name	string		Username of the user to remove (required).

Examples

Remove User

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
<jira:RemoveUser name="existing-user"/>
</JiraJelly>
```

jira:RunSearchRequest

This tag runs a search request against JIRA using a predefined filter.
Note: This tag will return a GenericValue for each issue which matches the search request.
A GenericValue consists of key-value pairs, e.g.

```
[GenericEntity:Issue]
[created,2007-11-01 15:51:25.0]
[summary,Testing]
[component,null]
[workflowId,12530]
[timeoriginalestimate,null]
[fixfor,null]
[type,2]
[timespent,null]
[environment,Windows]
[resolution,null]
[status,1]
[updated,2007-11-01 15:51:25.0]
[timeestimate,null]
[id,11540]
[key,TSTA-5]
[duedate,null]
[description,Test]
[project,10063]
[reporter,admin]
[security,null]
[votes,0]
[assignee,null]
[priority,3]
```

To retrieve a value, e.g. key, you can call gv.getString("key"). For full details, see the OFBiz GenericValue API.

Attributes

Attribute Name	Type	Default Value	Description

filterid	int		The id of the filter which will be used to run the search request.
size-var	string		The variable that will hold the number of issues returned from the search request.
var	string		The variable that will hold the issues returned from the search request.

Examples

Running a search request and iterating through the keys of the returned issues

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:RunSearchRequest filterid="10524" var="issues" size-var="issuecount"/>
    <core:forEach var="issue" items="${issues}">
        ${issue.key}
    </core:forEach>
</JiraJelly>
```

jira:SelectComponentAssignees

Selects the default assignees for newly created issues of the component.

Attributes

Attribute Name	Type	Default Value	Description
project-key	string		The key of the project you want to add the component to (required).
componentName	string		Name of the component (required).
assigneeType	string		Default assignee type (required). ***Assignee Types:*** • projectDefault • componentLead • projectLead • unassigned

Examples

Select a Component Assignee

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib">
<jira:SelectComponentAssignees project-key="ABC" componentName="Comp 1"
assigneeType="componentLead"/>
</JiraJelly>
```

jira:TransitionWorkflow

Please Note: This tag is not available in 3.3 and 3.3.1 — see JRA-7690 for details.

This tag executes a workflow transition on an issue.

Please keep in mind that if you are specifying field attribute/value pairs in your Jelly tag then these fields MUST be on the associted workflow transition screen. If the field is not on the screen then the value will not be set on the issue. For example, if you want to set the resolution attribute in your Jelly XML then your transition MUST have a screen associated with it that includes the resolution field on that screen.

Attributes

Attribute Name	Type	Default Value	Description

user	string	Currently logged in user	Username of the user to execute the workflow transition. The user needs to have the adequate permissions to execute the transition. Please note that the permissions required also depend on the fields that are updated during the transition. (See other attributes below).
key	string		The key of the issue to execute the transition on.
workflowAction	string		The id or name of the workflow transition to execute. If the argument can be converted to a number it is assumed to be an id of the transition. Otherwise it is assumed to be a name.
resolution	string		The id or name of the resolution to set on the issue during the transition. Please note that the transition must expect the resolution to be updated, otherwise an error is generated if this attribute is supplied. If the argument can be converted to a number it is assumed to be an id of the resolution. Otherwise it is assumed to be a name.
assignee	string		The username of the user to assign an issue to during the transition. The "user" executing the transition must have permissions to assign issues if this attribute is supplied. Please note that the transition must expect the assignee to be updated, otherwise an error is generated if this attribute is supplied. Use value "-automatic-" to let JIRA assign the issue to the default assignee.
fixVersions	string		A comma separated list of version ids or names to set as "fix for" versions during the transition. The "user" executing the transition must have permissions to set "fix for" versions if this attribute is supplied. Please note that the transition must expect the "fix for" versions to be updated, otherwise an error is generated if this attribute is supplied. If a value in the provided comma separated list can be converted to a number it is assumed to be an id of a version. Otherwise it is assumed to be a name.
comment	string		The comment to add to the issue during the transition. The "user" executing the transition must have permissions to add comments and the transition must be expecting comments to be added during its execution for the comment to be added successfully.
groupLevel	string		The level for the comment. The level must be a name of a group the user is a member of. NOTE: If this is specified you can not specify the roleLevel parameter.
roleLevel	string		Name or Id of Project Role that can see this comment. NOTE: If this is specified you can not specify the groupLevel parameter.

Examples

Execute Workflow Transition

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
                    <jira:TransitionWorkflow key="TST-6" user="testuser"
workflowAction="Resolve issue"
                    resolution="fixed" fixVersions="version 1,version 3" assignee="-automatic-"
                    comment="Test comment" groupLevel="jira-developers" />
</JiraJelly>
```

jira:UpdateProjectRole

This tag will update the name and description for a given project role id.

Attributes

Attribute Name	Type	Default Value	Description
projectroleid	int		The id of the project role you want to query
name	string		The name you want the project role updated with
description	string		The description you want the project role updated with

Examples

Updating a project role

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
    <jira:UpdateProjectRole projectroleid="123" name="unique name"
        description="my project role is nice" />
</JiraJelly>
```

Beta Tags

There are also a number of BETA tags that have not been fully tested or documented. The following list contains the tags and the attributes that can be passed to them:

- AddIssueSecurity
  - schemeId (required)
  - security (required)
  - type (required)
- AddIssueSecurityLevel
  - name (required)
  - description (required)
  - Output
    - jelly.issue.scheme.level.id
- CreateIssueSecurityScheme
  - name (required)
  - description (required)
  - Output
    - jelly.issue.scheme.id
- LoadManager
  - var (variable to put manager in)
  - manager (name of manager e.g. IssueManager)
- LoadProject
  - var (variable to put project in)
  - project-name (name of project)
- RemoveGroup
  - name (required)
- RemovePermissionScheme
  - schemeId (required)
  - confirm (required))
- RemoveProject
  - pId (required)
- SelectProjectScheme
  - projectKey (required)
  - permission-scheme (Name of permission scheme)or
  - issue-scheme (Name of issue security scheme)
- StringContains
  - value (String to look in)
  - possiblyContains (String to look for)
  - doesContain (true or false) if value contains possiblyContains == doesContain, the inside of the tag is executed.

If you would like more information on how to use the Beta tags, please read the source and/or post to the Atlassian Answers for JIRA.

Sample scripts

Creating a new Project

To properly partition projects, one needs a permission scheme per project, and project-specific groups to allocate permissions to. Setting up a new project can be a time-intensive process. The following sample Jelly scripts automate this:
This script might be used for a publicly visible project:

```
<?xml version="1.0"?>
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib"
xmlns:j="jelly:core">

  <j:set var="name" value="Test Project"/>
  <j:set var="key" value="TEST"/>
  <j:set var="lowerkey" value="test"/>
  <j:set var="lead_username" value="joe"/>
  <j:set var="lead_password" value="joe"/>
  <j:set var="lead_fullname" value="Joe Bloggs"/>
  <j:set var="lead_email" value="joe@example.com"/>
  <j:set var="url" value="http://example.com/TestProj"/>

  <jira:CreateUser username="${lead_username}" password="${lead_password}"
confirm="${lead_password}"
    fullname="${lead_fullname}" email="${lead_email}"/>
  <jira:CreateGroup group-name="${lowerkey}-developers">
    <jira:AddUserToGroup username="${lead}"/>
  </jira:CreateGroup>

  <jira:CreateProject key="${key}" name="${name}" url="${url}" lead="${lead_username}">
    <jira:CreatePermissionScheme name="${name} permissions">
      <jira:AddPermission type="reporter" permissions="Close"/>
      <jira:AddPermission group="jira-administrators" permissions="Close,Delete" type="group"/>
      <jira:AddPermission group="jira-users" permissions="Create,Edit,Comment,Link,Attach"
type="group"/>
      <jira:AddPermission group="${lowerkey}-developers"
        permissions="Project,ScheduleIssue,Move,Assign,Assignable,Resolve,Close,Work"
type="group"/>
      <jira:AddPermission group="Anyone" permissions="Browse,ViewVersionControl"/>
      <jira:SelectProjectScheme/>
    </jira:CreatePermissionScheme>
  </jira:CreateProject>
</JiraJelly>
```

This script is more complicated, with multiple groups per project:

```
<?xml version="1.0"?>
<!-- This script handles some of the administrative chores required when adding
a new project to JIRA. It creates the project, groups, permission scheme, and gives
groups the relevant permissions in the permission scheme. -->
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib" xmlns:j="jelly:core">

<!-- Name of the project to create -->
<j:set var="name" value="Jelly Test Project"/>
<!-- Key for the new project -->
<j:set var="key" value="TEST"/>
<!-- Existing user who will become the project lead (default assignee) -->
<j:set var="admin" value="admin"/>

<jira:CreateGroup group-name="${key}-users"/>
<jira:CreateGroup group-name="${key}-developers"/>
<jira:CreateGroup group-name="${key}-managers"/>
<jira:CreateGroup group-name="${key}-bizusers"/>
<jira:CreateGroup group-name="${key}-qa"/>

<jira:CreateProject key="${key}" name="${name}" lead="${admin}">
<jira:CreatePermissionScheme name="${key} Permission Scheme">
<jira:AddPermission type="reporter" permissions="Edit"/>
<jira:AddPermission type="assignee" permissions="Resolve"/>
<jira:AddPermission group="jira-administrators" permissions="Project,Delete" type="group"/>
<jira:AddPermission group="${key}-users" permissions="Browse,Create,Comment,Attach"
type="group"/>
<jira:AddPermission group="${key}-developers"
permissions="Move,Assignable,Link,ViewVersionControl"
    type="group"/>
<jira:AddPermission group="${key}-managers"
permissions="Edit,Assign,Assignable,Resolve,Close,Delete"
    type="group"/>
<jira:AddPermission group="${key}-bizusers" permissions="Assignable" type="group"/>
<jira:AddPermission group="${key}-qa" permissions="Assignable" type="group"/>
<jira:AddPermission group="opsmgrs" permissions="Browse,Edit,Assignable,Comment" type="group"/>
<jira:AddPermission group="dba-user-group" permissions="Browse,Assign,Assignable,Comment"
type="group"/>
<jira:AddPermission group="help-desk-group" permissions="Browse,Assign,Assignable,Comment"
type="group"/>
<jira:AddPermission group="webadmin-group" permissions="Browse,Assign,Assignable,Comment"
type="group"/>
<jira:AddPermission group="unix-admin-group" permissions="Browse,Assign,Assignable,Comment"
type="group"/>
<jira:SelectProjectScheme/>
</jira:CreatePermissionScheme>
</jira:CreateProject>
</JiraJelly>
```

For a list of projects, perform a project-specific operation.

This script iterates through a (comma-separated) list of projects, creates a project-specific group, and adds a user to that group.

```
<?xml version="1.0"?>
<!-- Jelly script to create 'support' group per project -->
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib" xmlns:util="jelly:util"
xmlns:j="jelly:core">
<util:tokenize var="projects"
delim=",">ARM,QWI,DWI,DBOR,DBSQ,LYX,MMM,MOI,TPAI,SEP,AMR,SLA,TP,TRBC,YRD</util:tokenize>
<j:forEach var="proj" items="${projects}">
<jira:CreateGroup group-name="${proj}-support"/>
<jira:AddUserToGroup username="jeff" group-name="${proj}-support"/>
</j:forEach>
</JiraJelly>
```

Create a user, issue, and assign the issue to the user

The following script creates a user (called `new-user`), creates a new issue, adds the user to the `jira-developers` group and assigns the issue to the user. It illustrates the use of context variables.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">

<jira:CreateUser username="new-user" password="password" confirm="password"
    fullname="Full name" email="test@test.com"/>
Username is ${jelly.new.username}
 <jira:CreateIssue project-key="TP" summary="New issue summary" issueKeyVar="ik"/>
 <jira:AddUserToGroup username="new-user" group-name="jira-developers"/>
 <jira:AssignIssue key="${ik}" assignee="${jelly.new.username}"/>

</JiraJelly>
```

Assigning and Starting Progress

Here we create an issue, assign it to 'bob' (who must be in `jira-developers`), and start progress:

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">

 <jira:CreateIssue project-key="TP" summary="New issue" issueKeyVar="ik"/>
 <jira:AssignIssue key="${ik}" assignee="bob"/>
 <jira:TransitionWorkflow key="${ik}" user="bob" workflowAction="Start Progress" />
</JiraJelly>
```

Moving unreplied-to issues into an 'Inactive' state

When JIRA is used for interacting with customers, this script is useful for finding issues which are awaiting customer response, and haven't been responded to in a while. It moves such issues into an 'Inactive' state.

You would typically invoke this script periodically with the Jelly Service.

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib"
xmlns:core="jelly:core" xmlns:log="jelly:log" >
<jira:Login username="customersupport" password="XXXXXX">
        <log:warn>Running Inactivate issues service</log:warn>
        <core:set var="comment">This issue has not been updated for 5 business days.

If you have an update, please use "Add Comments For Vendor" action to let us know.
If you need more time to gather information please let us know and we will 'freeze' this issue.

If you have no other questions, please Close this issue.

If no update is received in the next 5 business days, this issue will be automatically closed.

Thank you,

  The Support Team</core:set>
        <core:set var="workflowStep" value="Mark Inactive" />
        <core:set var="workflowUser" value="customersupport" />

        <!-- Run the SearchRequestFilter -->
        <jira:RunSearchRequest filterid="11505" var="issues" />

        <core:forEach var="issue" items="${issues}">
                <log:warn>Inactivating issue ${issue.key}</log:warn>
                <jira:TransitionWorkflow key="${issue.key}" user="${workflowUser}"
workflowAction="${workflowStep}" comment="${comment}"/>
        </core:forEach>
</jira:Login>
</JiraJelly>
```

Where:

- **workflowStep** is the name of a workflow transition, e.g "Close Issue", "Start Progress", just as they appear in the left-hand menu on the issue screen.
- **workflowUser** is the user to run the transition as
- **filterid** is the id of a saved search (filter), which finds issues needing to be inactivated (transitioned). This ID can be discovered from the filter URL on the "Manage" tab in "Find issues".

The JIRA Toolkit is useful in conjuction with this script, to find issues awaiting customer response.

JIRA Toolkit (Customer Support Extensions)

As an extension to JIRA, Atlassian have developed a set of JIRA custom fields, collectively called the "JIRA Toolkit". It can be found online at the Atlassian Plugin Exchange

These custom fields are particularly useful in customer-facing JIRA instances. They were initially developed for use in Atlassian's own JIRA Support installation at http://support.atlassian.com. See the JIRA Toolkit documentation for details.

Developer Guides

Please refer to the JIRA Developer Documentation.

Building JIRA from Source

Commercial users at any level receive access to JIRA's source code. This topic explains how to build this source code into a deployable JIRA application.

Building all of JIRA from source is only necessary if you need to make extensive modifications to JIRA's source code and are using a WAR approach.

You should not need to rebuild JIRA if:

- You need to change many JSP files. JSP files are the template files for many JIRA webpages. They can also be more easily changed directly in the standalone JIRA installation directory.
- Create a JIRA plugin that adds functionality to JIRA. For more information, read the JIRA Plugin Guide. Changes and enhancements to JIRA's functionality can often be made with JIRA plugins without requiring core JIRA source code modifications.
- Recompiling a small number of source files can be done using the instructions in the standalone external-source directory.

On this page:

- Building a JIRA WAR file from a JIRA Source release
- Developing using the IDE Connectors
- Obtaining the source of JIRA's dependencies
- Compiling Single Class Patches

*The information on this page does not apply to **JIRA OnDemand**.*

Building a JIRA WAR file from a JIRA Source release

1. Ensure you have JDK 1.6 or higher *and* have a Subversion client installed.
2. Download Maven 2.1.0 from the Apache archives of the Maven website.
   We have not yet tested building JIRA from source using Maven 3 at this time.
3. Extract Maven to an appropriate location on your operating system. For example,
   On Windows, extract to:

   ```
   C:\apache-maven-2.1.0
   ```

   On Mac/Linux, extract to:

   ```
   /usr/local/apache-maven-2.1.0
   ```

4. Set the M2_HOME environment variable. For example,
   On Windows:

   ```
   > set M2_HOME=C:\apache-maven-2.1.0
   ```

Alternatively, the Windows environment variables can be configured by choosing `My Computer >> Properties >> Advanced >> Environment Variables`.

On Mac/Linux:

```
export M2_HOME=/usr/local/apache-maven-2.1.0
```

5. Add Maven's bin directory to your path. For example,
   On Windows:

```
> set PATH=%M2_HOME%\bin
```

You can set this via `My Computer >> Properties >> Advanced >> Environment Variables` again if you wish.

On Mac/Linux:

```
export PATH=$PATH:$M2_HOME/bin
```

6. Install all of the following restricted third party (`.jar`) libraries to your local Maven repository (`.m2`), ensuring that you download the version specified below. All of these libraries are required to successfully build JIRA from source. If any of these libraries are missing, the build process will fail.
   Due to licensing restrictions, we are unable to distribute these third party libraries from Atlassian's public Maven repository. If you have built previous versions of JIRA from source, you may already have some of these libraries in your local Maven repository.

Libraries	Maven groupId and artifactId	Version	Download URL
activation	javax.activation:activation	1.0.2	http://repository.jboss.org/nexus/service/local/repositories/deprecated/conter
jms	javax.jms:jms	1.1	http://repository.jboss.org/nexus/content/groups/public-jboss/javax/jms/jms/1
jmxri and jmxtools	com.sun.jmx:jmxri and com.sun.jdmk:jmxtools	1.2.1	http://www.oracle.com/technetwork/java/javase/tech/download-jsp-141676.ht (Download 'JMX 1.2.1 Reference Implementation')
jndi	jndi:jndi	1.2.1	http://java.sun.com/products/jndi/downloads/index.html
jta	jta:jta	1.0.1B	http://www.oracle.com/technetwork/java/javaee/jta/index.html (Download 'Class Files 1.0.1B')
mail	javax.mail:mail	1.3.2	http://www.oracle.com/technetwork/java/javamail-1-3-2-138617.html
ojdbc6	com.oracle:ojdbc6	11.2.0.2.0	http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-112010- Database 11g Release 2 (11.2.0.2.0) JDBC Drivers')

To install these restricted third party libraries:

a. Download each one (from its link above) into a directory on your file system, for example, `downloads` in your home directory area.

The **jmxri**, **jmxtools**, **jndi**, **jta** and **mail** libraries are downloaded as `.zip`files and before you can install these libraries into your local Maven repository, either:

- a key `.jar` file must be extracted from them or
- they need to be in `.jar` form.

i. For **jmxri** and **jmxtools**:

- **On Windows:**
  Use Windows Explorer to enter the `downloads` directory and extract the `jmxri.jar` and `jmxtools.jar` files from the `jmx-1_2_1-bin\lib` subdirectory of the `jmx-1_2_1-ri.zip` file.
- **On Linux:**

```
cd $HOME/Downloads
unzip jmx-1_2_1-ri.zip jmx-1_2_1-bin/lib/jmxri.jar
jmx-1_2_1-bin/lib/jmxtools.jar
```

ii. For **jndi**:

- **On Windows:**
  Use Windows Explorer to enter the `downloads` directory and extract the `jndi.jar` file from the `lib` subdirectory of the `jndi-1_2_1.zip` file.
- **On Mac/Linux:**

```
cd $HOME/Downloads
unzip jndi-1_2_1.zip lib/jndi.jar
```

iii. For **jta**:

- **On Windows:**
  Use Windows Explorer to enter the `downloads` directory and rename the `jta-1_0_1B-classes.zip` file to `jta-1_0_1B-classes.jar`
- **On Mac/Linux:**

```
cd $HOME/Downloads
mv jta-1_0_1B.zip jta-1_0_1B.jar
```

iv. For **mail**:

1. **On Windows:**
   Use Windows Explorer to enter the `downloads` directory and extract the `mail.jar` file from the `javamail-1.3.2` subdirectory of the `javamail-1.3.2.zip` file.
2. **On Mac/Linux:**

```
cd $HOME/Downloads
unzip javamail-1.3.2.zip javamail-1.3.2/mail.jar
```

b. Once you have downloaded, expanded and renamed each of these libraries, install them into your local Maven repository. For example, in your `downloads` directory, enter the following commands:

```
mvn install:install-file -DgroupId=javax.activation -DartifactId=activation
-Dversion=1.0.2 -Dpackaging=jar -Dfile=activation-1.0.2.jar
mvn install:install-file -DgroupId=javax.jms -DartifactId=jms -Dversion=1.1
-Dpackaging=jar -Dfile=jms-1.1.jar
mvn install:install-file -DgroupId=com.sun.jmx -DartifactId=jmxri -Dversion=1.2.1
-Dpackaging=jar -Dfile=jmxri.jar
mvn install:install-file -DgroupId=com.sun.jdmk -DartifactId=jmxtools
-Dversion=1.2.1 -Dpackaging=jar -Dfile=jmxtools.jar
mvn install:install-file -DgroupId=jndi -DartifactId=jndi -Dversion=1.2.1
-Dpackaging=jar -Dfile=jndi.jar
mvn install:install-file -DgroupId=jta -DartifactId=jta -Dversion=1.0.1
-Dpackaging=jar -Dfile=jta-1_0_1B-classes.jar
mvn install:install-file -DgroupId=javax.mail -DartifactId=mail -Dversion=1.3.2
-Dpackaging=jar -Dfile=mail.jar
mvn install:install-file -DgroupId=com.oracle -DartifactId=ojdbc6
-Dversion=11.2.0.2.0 -Dpackaging=jar -Dfile=ojdbc6.jar
```

7. Download the JIRA source archive from http://www.atlassian.com/software/jira/JIRASourceDownloads.jspa.
   You will need to log in as a user with a commercial license to access this page.

8. Extract the JIRA source archive to a location of your choice. This will create a subdirectory with the name `atlassian-jira-X.Y-source`, where X.Y is your version of JIRA. For example, `C:\atlassian-jira-4.3-source`
9. Change directory into this subdirectory and build JIRA by executing the following Maven 2 command. For example,
   On Windows:

   ```
   C:\atlassian-jira-4.3-source\> build.bat
   ```

   On Mac/Linux:

   ```
   > build.sh
   ```

   The build script will download several dependencies from Atlassian's public Maven repository.

   On rare occasions, however, the build process may fail and you may receive an error similar to the one encountered when an Atlassian product is unable to install a plugin from Atlassian's public Maven repository. This problem is caused by the JVM being unable to access its default 'cacerts' file, which contains a certificate that trusts Atlassian's public Maven repository.

   To resolve this problem:

   - Try one of the solutions mentioned in the Confluence Knowledge Base article: Unable to Install Plugin Due to Maven SSL.
   - If that does not resolve the problem, you may have a 'jssecacerts' file which does not contain a certificate that trusts Atlassian's public Maven repository. If a 'jssecacerts' file is present in your JVM, the 'cacerts' file may be ignored.

10. A WAR file called `jira-webapp-dist-X.Y.war` (where X.Y is your version of JIRA), will be built in the `jira-project/jira-distribution/jira-webapp-dist/target` subdirectory of your extracted JIRA source directory.

    For example, if the subdirectory created above was `C:\atlassian-jira-4.3-source`, the WAR file will be found in: `C:\atlassian-jira-4.3-source\jira-project\jira-distribution\jira-webapp-dist\target\jira-webapp-di`

    An unpacked version of your JIRA source build can also be found in the `jira-project/jira-distribution/jira-webapp-dist/target/jira-webapp-dist-X.Y` subdirectory of your extracted JIRA source directory.
11. The WAR file generated can now be installed into your application server to run the JIRA application you just built. For more information, refer to the JIRA WAR Configuration Overview. For specific installation instructions, you can follow these procedures, skipping stages 1 and 3:
    - Installing JIRA on Tomcat 5.5
    - Installing JIRA on Tomcat 6.0

Developing using the IDE Connectors

Learn about the IDE Connectors from the IDE Connector Documentation.

Obtaining the source of JIRA's dependencies

When building JIRA from source, Maven will fetch the binary (compiled) dependencies that it requires automatically during the build process, so you do not have to do it manually (with the exception of the third party libraries mentioned above).

It is worth noting that JIRA's source distribution not only ships with JIRA's source code, it also includes the source of the internal Atlassian projects that JIRA depends on (e.g. atlassian-bonnie, atlassian-core, etc.). These internal Atlassian dependencies are also built from source when you build JIRA.

Other dependencies are available on Atlassian's public repository. The source of these dependencies is usually available on the library's website (try googling for the library name), or can be identified in the SCM information of the relevant library.

If you have any questions regarding the build process, please post to the JIRA Development Forum, which is monitored continually by the development community, and by Atlassian as often as possible.

Compiling Single Class Patches

If you just want to compile one class (perhaps a service), we have a step-by-step guide for how to do this in IDEA. See How to Make a JIRA Patch for details.

How to Make a JIRA Patch

To make any substantial modifications or additions to JIRA's source, you should read Building JIRA from Source. This implies building a WAR and deploying this to your Application Server.

> *The information on this page does not apply to **JIRA OnDemand**.*

Making a Single Class Patch

This guide describes how to make a source code modification to a single class file.

1. Download Maven 2.1.0 from the Apache archives of the Maven website.
2. Set your `PATH` and `M2_HOME` environment variables where you install Maven (and its /bin directory for the PATH).
   Refer to the Building JIRA from Source documentation for details.
3. Download JIRA source.
4. Changed directory into your extracted JIRA source directory and then into its `jira-project` subdirectory.
5. Run one of the following, depending on your preferred IDE:

   ```
   mvn idea:idea
   ```

   OR

   ```
   mvn eclipse:eclipse
   ```

6. Open the resulting project.
7. From your IDE, build the project.
8. From your IDE, open and compile a file. The compiled file will appear in the `target/classes` directory of the maven module that you are working on. If you are working with JIRA's core classes, this is likely to be in `/jira-project/jira-components/jira-core/target/classes`

Deploying the Patch

To deploy a patch, drop the file in the classpath from `<jira-install>/WEB-INF/classes`. For example, if you compile the class:

```
com.atlassian.jira.appconsistency.integrity.check.SearchRequestRelationCheck
```

...it will be available from:

```
/jira-project/jira-components/jira-core/target/classes/com/atlassian/jira/appconsistency/integrity/
```

To deploy this class, place it in:

```
<jira-install>/WEB-INF/classes/com/atlassian/jira/appconsistency/integrity/check/SearchRequestRelat
```

...then restart JIRA.

API Documentation

The JIRA API docs are available online. They are most useful for:

- users writing Plugins, Listeners and Services
- users with commercial licenses who wish to modify JIRA
- partners embedding JIRA as a J2EE component

The latest API docs are available at http://docs.atlassian.com/software/jira/docs/api/latest/. The 4.2 docs are available at http://docs.atlassian.com/software/jira/docs/api/4.2/. JDiff reports listing the changes to the JIRA API between releases are available here. For previous versions, substitute the appropriate version in the URL.

JIRA Installation and Upgrade Guide

The pages listed below contain information on installing and upgrading JIRA:

- JIRA Requirements
  - Installing Java
- Supported Platforms
  - End of Support Announcements for JIRA
  - Caveats in using Firefox 3.6.0 with JIRA
- Installing JIRA
  - Installing JIRA on Windows
  - Installing JIRA on Linux
  - Installing JIRA from an Archive File on Windows, Linux or Solaris
  - Installing JIRA WAR
  - Configuring Your JIRA Installation
- Running the Setup Wizard
  - Next Steps - Adding Users
  - Next Steps - Creating a Project
  - Next Steps - Creating an Issue
- Connecting JIRA to a Database
  - Connecting JIRA to PostgreSQL
  - Connecting JIRA to MySQL
  - Connecting JIRA to Oracle
  - Connecting JIRA to SQL Server 2005
  - Connecting JIRA to SQL Server 2008
  - Connecting JIRA to HSQLDB
  - Switching Databases
  - Surviving Connection Closures
- Upgrading JIRA
  - Upgrading JIRA Manually
  - Migrating JIRA to Another Server
  - Disabling Auto-Export
  - Rolling Back a JIRA Upgrade
- Important Directories and Files
  - JIRA Installation Directory
  - JIRA Home Directory
  - Setting your JIRA Home Directory
- Tomcat security best practices
- Customising Your JIRA Installation
  - Changing JIRA's TCP Ports
  - Running JIRA in a Virtualised Environment
  - Running JIRA over SSL or HTTPS
  - Installing Confluence and JIRA Together
  - Integrating JIRA with a Web Server
  - Securing JIRA with Apache HTTP Server
- Deployment Planning Activity

JIRA Requirements

JIRA is a 'web application', meaning it runs centrally on a server, and users interact with it through web browsers from any computer.

If you are considering running JIRA on VMware, please read the Running JIRA in a Virtualised Environment.

- JIRA Client/Server Software Requirements
  - 1. Browser
  - 2. Java
  - 3. Application Server
  - 4. Database
- JIRA Server Hardware Recommendations
- Next Steps

JIRA Client/Server Software Requirements

Please read the Supported Platforms page for JIRA, which lists the required server and client software supported by **JIRA 5.0.x**, including:

- Browsers (client-side)
- Java platforms (JDK/JRE) (server-side)
- Operating systems (server-side)
- Application servers *(if you are installing the JIRA WAR distribution)* (server-side)
- Databases (server-side)

Please also read the information below regarding server and client software requirements for JIRA.

1. Browser

If you have disabled JavaScript in your browser or are using a script blocking tool like NoScript, you must enable your browser to execute JavaScript from JIRA to access JIRA's full functionality.

2. Java

JIRA requires a **Java Developers Kit (JDK)** or **Java Runtime Environment (JRE)** platform to be installed on your server's operating

system.

If you intend to use the Windows Installer or Linux Installer to install JIRA, there is no need to install and configure a separate JDK/JRE since these executable files will install and configure their own JRE to run JIRA.

If, however, you intend to install JIRA from an archive or you plan to install the JIRA WAR distribution, then you will first need to install a supported Java platform. (Refer to Supported Platforms for supported Java Platforms). For instructions on how to install a supported Java platform for JIRA, please refer to Installing Java.

Please Note:

- Currently, Oracle JDK/JRE (formerly Sun JDK/JRE) is available for Windows (32-bit + 64-bit), Linux (32-bit + 64-bit) and Solaris Platforms (32-bit + 64-bit).
  Mac OS X systems are packaged with a JDK optimised for their hardware and operating systems. However, these JDKs are not supported by JIRA.
- A JIRA installation running on a 64-bit Java platform may require additional memory (to run at a similar level of performance) to a JIRA installation running on a 32-bit Java platform.
  This is because a 64-bit Java platform's object references are twice the size as those for a 32-bit Java platform.

3. Application Server

JIRA is a web application that requires an application server. However, this requirement differs based on the type of JIRA distribution you intend to install:

- **'Recommended' JIRA distributions** (installed using 'Windows Installer', 'Linux Installer' or from an 'Archive File') are pre-configured with Apache Tomcat, which is a stable, lightweight and fast-performing application server. (There is no need to install a separate application server if you intend to install one of these recommended JIRA distributions.)
- The **JIRA WAR distribution** can be installed into an application server (supported by Atlassian), provided this application server is compatible with your operating system and Java platform. You must manually configure your JIRA WAR installation to operate with an existing application server installation.

4. Database

JIRA requires a relational database to store its issue data. JIRA supports most popular relational database servers, so we suggest using the one that you are most comfortable with administering. JIRA ships pre-configured with the HSQLDB database, which is suitable for evaluation purposes only, since HSQLDB is prone to database corruption.

Hence, if you intend to use JIRA in a production environment, we **strongly recommend** that you connect JIRA to an enterprise database (supported by Atlassian).

JIRA Server Hardware Recommendations

During evaluation, JIRA will run well on any reasonably fast workstation computer (e.g. 1.5+ Ghz processor). Memory requirements depend on how many projects and issues you will store, but 300MB – 1GB (of Java heap size) is enough for most evaluation purposes.

Most people start by installing JIRA on their local computer and migrate this to a production server later.

The hardware required to run JIRA in production depends mainly on the number of issues and users that your installation will have, as well as the maximum number of concurrent requests that the system will experience during peak hours.

- If you are planning to have a small number of projects (10-20) with 1,000 to 5,000 issues in total and about 100-200 users, a recent server (2.0+ GHz quad core CPU) with 1-2 GB of available RAM should cater for your needs.
- If you are planning for a greater number of issues and users, adding more memory will help. We have reports that allocating 1GB of RAM to JIRA is sufficient for 100,000 issues. If in doubt, allocate more memory than you think you need.
- If your system will experience a large number of concurrent requests, running JIRA on a multi-CPU machine will increase the concurrency of processing the requests, and therefore speed up the response time.

For reference, our http://jira.atlassian.com site has over 125,000 issues and over 70,000 user accounts. The system runs on a server with a 64-bit quad core processor and 32 GB of memory, of which only 1.5 GB is allocated to JIRA.

Please note that performance heavily depends on your usage pattern. As a general rule, if you wish to store more than 200,000 issues in JIRA we recommend that you set up separate instances of JIRA on different physical machines and split your projects and issues between the instances. You can follow the instructions on splitting a JIRA instance, if you need to convert an existing JIRA instance into multiple instances.

As your JIRA system evolves, you may need to eventually increase JIRA's memory to maintain performance. Factors that will affect JIRA's memory requirements include an increase in the following:

- the number of users concurrently on the system
- the number of concurrent operations
- the number of plugins (some of which may have their own memory requirements)
- the number of workflow step executions (such as Transitions and Post Functions)
- the number of jobs and schedule services
- the volume of email notifications

See JIRA Performance Tuning for tips on scaling JIRA. We would appreciate it if you let us know what hardware configuration works for you. Please create a ticket in our support system with your hardware specification and mention the number of users and issues in your JIRA installation.

Your JIRA database's size depends predominantly on the number of issues stored in your JIRA instance. Also, using more custom fields may also increase the size of your JIRA database.

Please Note: JIRA requires access to a local disk for certain functionality. If JIRA does not have read and write access to a local disk, searching and saving/accessing attachments will not work.

While some of our customers run JIRA on SPARC-based hardware, Atlassian only officially supports JIRA running on x86 hardware and 64-bit derivatives of x86 hardware.

Next Steps

Install JIRA.

Installing Java

Please skip these instructions if you are intend to use or have used the Windows Installer or Linux Installer to install JIRA, since these executable files will install and configure their own JRE to run JIRA.

On this page:

- 1. Installing Java
- 2. Setting JAVA_HOME
  - Linux-based computers
  - Windows-based computers
- 3. Confirming that Java Works
- Next Step

1. Installing Java

JIRA requires Oracle's (formerly Sun's) Java Development Kit (JDK) or Java Runtime Environment (JRE) platform to run. Refer to Supported Platforms for details on the Java platform versions that JIRA supports.

Oracle's JDK/JRE can be downloaded from Oracle's website.

Linux distributions frequently have an open-source implementation of Java called GCJ installed. Do not use this Java platform — it is incomplete and JIRA will not run successfully on it.

You can test whether you have the correct Java platform by running `java -version`:

```
~$ java -version
java version "1.6.0"
Java(TM) SE Runtime Environment (build 1.6.0-b105)
Java HotSpot(TM) Client VM (build 1.6.0-b105, mixed mode, sharing)
```

On recent Linux distributions, Oracle's (formerly Sun's) JDK can be installed with a command like `sudo apt-get install sun-java6-jdk` (for Ubuntu).

On some X.org-based distros (eg. Fedora Core 4), you may see an error like this:

```
java.lang.UnsatisfiedLinkError: /opt/j2sdk1.4.2_11/jre/lib/i386/libawt.so: libXp.so.6: cannot
open
        shared object file: No such file or directory
```

If you do, you will need to install the **xorg-x11-deprecated-libs** package (Fedora) or equivalent (check Google).

2. Setting JAVA_HOME

Once the JDK or JRE is installed, you will need to set the `JAVA_HOME` environment variable, whose value is the root directory of the JDK/JRE.

Some JDK/JRE installers set this automatically (check by typing '`echo %JAVA_HOME%`' in a Windows command prompt, or '`echo $JAVA_HOME`' in a Linux/UNIX console).

Linux-based computers

On many Linux-based computers, the `JAVA_HOME` environment variable is set in the `/etc/environment` file.

If `JAVA_HOME` is not defined in this file, you can set it using the following command at a shell prompt, when logged in with 'root' level permissions:

- `echo JAVA_HOME="path/to/JAVA_HOME" >> /etc/environment`

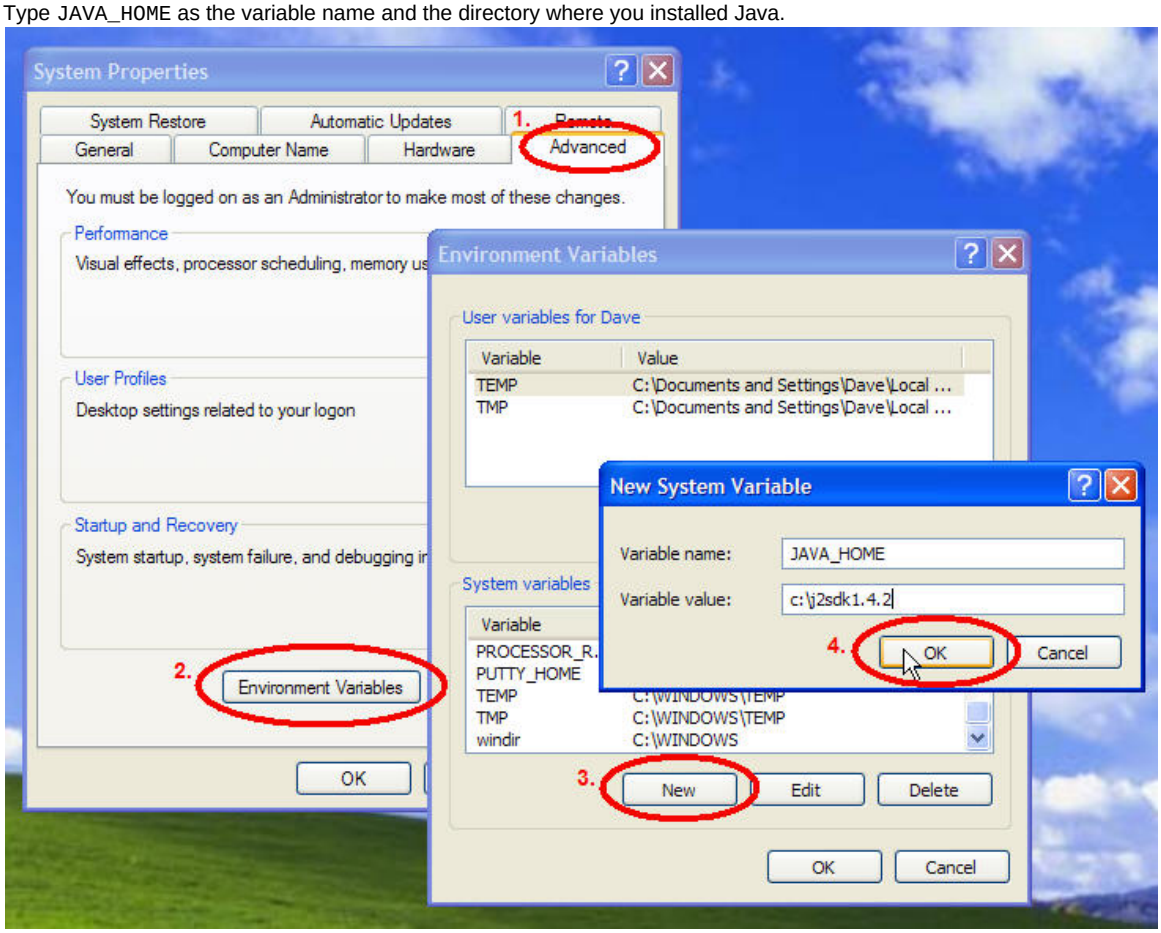
If, however, `JAVA_HOME` is already defined in this file, open the `/etc/environment` file in a text editor and modify its value to the appropriate `path/to/JAVA_HOME` — that is:

- `JAVA_HOME="path/to/JAVA_HOME"`

Windows-based computers

If this environment variable is not set on a Windows-based computer, you can set it in the Control Panel using the following procedure:

1. Open the Windows 'Advanced' system properties dialog box:
   - On Windows XP-based operating systems, right-click on the **My Computer** icon on your desktop (or via the **Start** menu), select '**Properties**' and click the '**Advanced**' tab.
   - On Windows 7-based operating systems, right-click the **Computer** icon on your desktop (or via the **Start** menu), select '**Properties**', click '**Advanced system settings**', select '**Properties**' and click the '**Advanced**' tab.
2. Click the **Environment Variables** button.
3. Click one of the **New** buttons (to define a new environment variable for your user account, or if available, system-wide).
4. Type `JAVA_HOME` as the variable name and the directory where you installed Java.



5. After clicking the required '**OK**' buttons to save your changes, your `JAVA_HOME` environment variable should be available in a *new* command prompt window. If not or if necessary, restart your computer.

3. Confirming that Java Works

Once the steps above have been done, it should be possible to open a Windows command prompt and type `%JAVA_HOME%\bin\java -version` (or `"%JAVA_HOME%"\bin\java -version` if your `%JAVA_HOME%` value contains spaces) and see output similar to this:

```
java version "1.6.0_19"
Java(TM) SE Runtime Environment (build 1.6.0_19-b04)
Java HotSpot(TM) Client VM (build 16.2-b04, mixed mode, sharing)
```

If you subsequently start JIRA and you receive an error like **Windows cannot find '-Xms128m'**, then you may not have correctly set `JAVA_HOME`. Please verify step 2 of the procedure above.

Next Step

Installing JIRA from an Archive File on Windows, Linux or Solaris

Supported Platforms

This page describes the supported platforms for **JIRA 5.0.x**. If a particular platform's version is not mentioned on this page, then we do not support that version of the platform for **JIRA 5.0.x**.

Please also ensure you have read the JIRA Requirements page, since not all the platforms listed below may be required for your specific JIRA setup.

End of Support Announcements

Please read End of Support Announcements for JIRA for important information regarding the end of support for various platforms and browsers when used with JIRA.

Key: ✅ = Supported; ❌ = Not Supported

Java Platforms	
Oracle JDK / JRE (formerly Sun JDK / JRE)	✅ 1.6 (update 18 or higher is required; update 24 or higher is recommended) ❌ 1.7
Operating Systems	
Microsoft Windows (1)	✅
Linux / Solaris (1, 2, 3)	✅
Application Servers	
Apache Tomcat (4)	✅ 6.0.32 ❌ 6.0.0 – 6.0.19 ✅ 5.5.27, 5.5.28, 5.5.29
Databases	
MySQL	✅ 5.x (excluding 5.0) with JDBC Connector/J 5.1 ❌ 5.0
PostgreSQL	✅ PostgreSQL 8.2 and later with PostgreSQL Driver 8.4.x
Microsoft SQL Server	✅ 2008 with the JTDS 1.2.4 driver ✅ 2005 with the JTDS 1.2.4 driver ❌ Express Editions are not supported
Oracle	✅ 11G with Oracle 11.2.x drivers ❌ 10G
HSQLDB (5)	✅ (Bundled with JIRA but supported for evaluation use only)

Web Browsers [6]	
Microsoft Internet Explorer [7]	✅ 8.0, 9.0 ❌ 7.0
Mozilla Firefox [7]	✅ Latest stable version supported ✅ Tested with 3.6 [8], 4.0, 5.0, 6.0
Safari [7]	✅ Latest stable version supported ✅ Tested with 5
Chrome	✅ Latest stable version supported ✅ Tested with 12

Notes:

1. JIRA is a pure Java-based application and should run on any supported operating system, provided that the JDK / JRE requirements are satisfied.
2. Atlassian only officially supports JIRA running on x86 hardware and 64-bit derivatives of x86 hardware. If you are installing JIRA from an archive, you should create a dedicated user account on the operating system to run JIRA, since JIRA runs as the user it is invoked under and therefore can potentially be abused.
3. Although the JIRA Linux Installer is designed to install successfully on all 'flavours' of Linux, we only test the JIRA Linux Installer on CentOS Linux. If you encounter problems with the JIRA Linux Installer on your particular flavour of Linux, we recommend installing JIRA on Linux from an archive file.
4. Deploying multiple Atlassian applications in a single Tomcat container is **not supported**. Please see Deploying Multiple Atlassian Applications in a Single Tomcat Container for reasons why we do not support this configuration.
5. JIRA ships with a built-in database (HyperSQL DataBase or HSQLDB). While this database is suitable for evaluation purposes, it is susceptible to data loss during system crashes. Hence, for production environments we strongly recommend that you configure JIRA to use an external database.
6. Please refer to our Patch Policy on fixing browser issues.
7. Minimum screen resolution of 1024 x 768 (when these browsers are maximised).
8. Firefox version 3.6.0 possesses a bug that could result in data loss when used with JIRA. If you use this version of Firefox, please either upgrade your version of Firefox to 3.6.2 or above (preferably 5.0), or refer to our Firefox 3.6.0 caveats document for more details about this issue and how to avoid it.

End of Support Announcements for JIRA

This page contains announcements of the end of support for various platforms and browsers used with JIRA. These are summarised for upcoming JIRA releases in the table below. Please see the following sections for the full announcements.

End of Support Matrix for JIRA

The table below summarises the end of support announcements for **upcoming** JIRA releases. If a platform is not (or no longer) supported by **JIRA 5.0.x**, it is **not** listed in this table.

Platform	**JIRA End of Support**
Internet Explorer 7 web browser	After the last version of JIRA 4.4.x (announcement)
Oracle 10.1 & 10.2 databases	After the last version of JIRA 4.4.x (announcement)
MySQL 5.0 database	After the last version of JIRA 4.4.x (announcement)

Why is Atlassian ending support for these platforms?

Atlassian is committed to delivering improvements and bug fixes as fast as possible. We are also committed to providing world class support for all the platforms our customers run our software on. However, as the complexity of our applications grows, the cost of supporting multiple platforms increases exponentially. Each new feature has to be tested on several combinations of application servers, databases, web browsers, etc, with setup and ongoing maintenance of automated tests. Moving forward, we want to reduce the time spent there to increase Confluence development speed significantly.

On this page (most recent announcements first):

- Deprecated Web Browsers for JIRA (6 April 2011)
- Deprecated Databases for JIRA (6 April 2011)
- Deprecated Web Browsers for JIRA (27 September 2010)
- Deprecated Databases for JIRA (13 August 2010)
- Deprecated Application Servers for JIRA (27 January 2010)
- Deprecated Java Platforms for JIRA (27 January 2010)
- Deprecated Web Browsers for JIRA (11 December 2009)

Deprecated Web Browsers for JIRA (6 April 2011)

This section announces the end of Atlassian support for certain web browsers for JIRA.

We will be **ending support for older versions of web browsers** as follows:

- JIRA 4.4.x will be the last versions of JIRA to support Internet Explorer 7.

The details are below. Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Internet Explorer 7 End of Support Notes:

- IE7, now 4+ years after its release, has less than 10% of browser market share. Microsoft has released IE8 and recently IE9.
- JIRA 4.4 (due mid 2011) will be the last major version of JIRA to officially support Internet Explorer 7.
- JIRA 4.4.x and earlier versions will continue to work with Internet Explorer 7. However, we will not fix bugs affecting this browser version past the last version of JIRA 4.4.x to be released.

Deprecated Databases for JIRA (6 April 2011)

This section announces the end of Atlassian support for certain databases for JIRA.

We will be **ending support for older versions of databases** as follows:

- JIRA 4.4.x will be the last versions of JIRA to support Oracle 10.1, Oracle 10.2 and MySQL 5.0.

The details are below. Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Oracle 10.1 & 10.2 End of Support Notes:

- Oracle Premier support for 10.2 ended on July 31, 2010.
- JIRA 4.4 (due mid 2011) will be the last major version of JIRA to officially support Oracle 10.1 and Oracle 10.2.
- JIRA 4.4.x and earlier versions will continue to work with Oracle 10.1 and Oracle 10.2. However, we will not fix bugs affecting Oracle 10.1 and Oracle 10.2 past the last version of JIRA 4.4.x to be released.

MySQL 5.0 End of Support Notes:

- MySQL support for MySQL 5.0 ended on December 31, 2009 (http://www.mysql.com/support/eol-notice.html).
- JIRA 4.4 (due mid 2011) will be the last major version of JIRA to officially support MySQL 5.0.
- JIRA 4.4.x and earlier versions will continue to work with MySQL 5.0. However, we will not fix bugs affecting MySQL 5.0 past the las version of JIRA 4.4.x to be released.

Deprecated Web Browsers for JIRA (27 September 2010)

This section announces the end of Atlassian support for certain web browsers for JIRA.

We will **stop supporting older versions of web browsers** as follows:

- From JIRA 4.3, due in Quarter 1 2011, JIRA will no longer support Safari 4 and Firefox 3.0.x.

The details are below. Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

End of Life Announcement for Web Browser Support

Web Browsers	Support End Date

Safari 4	When JIRA 4.3 releases (due in Quarter 1 2011)
Firefox 3.0.x	When JIRA 4.3 releases (due in Quarter 1 2011)

- **General End of Support Notes:**
  - JIRA 4.2 (due Quarter 4 2010) will be the last JIRA version to officially support Safari 4 and Firefox 3.0.x.
  - 'Support End Date' means that JIRA 4.2 and previous released versions will continue to work with Safari 4 and Firefox 3.0.x However, we will not fix bugs affecting these browser versions past the support end date.
- **Firefox End of Support Notes:**
  - The decision to end support for Firefox 3.0.x in JIRA 4.3 was made in line with Mozilla's support strategy, which indicates that Firefox 3.0.x will be maintained with security and stability updates until January 2010.

Deprecated Databases for JIRA (13 August 2010)

This section confirms that Atlassian support for DB2 for JIRA ended in JIRA 4.0. End of support means that Atlassian will no longer fix bugs related to DB2.

We **do not support the following databases**:

- Atlassian ended support for DB2 at the release of JIRA 4.0 (October 2009), with the final support for these platforms in JIRA 3.13.

We made this decision in order to reduce our database support, to reduce testing time and help us speed up our ability to deliver market-driven features. We are committed to helping our customers understand this decision and assist them in migrating to a supported database, if needed.

Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

- **DB2 End of Support Notes:**
  - 'Support End Date' means that JIRA 3.13 and versions prior to JIRA 3.13 will continue to work with the DB2. However, we will not fix bugs affecting DB2 past the support end date.
  - JIRA 4.0 (released in October 2009) and later versions of JIRA have not been tested with DB2.

Deprecated Application Servers for JIRA (27 January 2010)

This section announces the end of Atlassian support for certain application server platforms for JIRA WAR/EAR. End of support means that Atlassian will not fix bugs in certain application servers past the support end date.

We will **stop supporting the following application servers**:

- From JIRA 4.1, due late Q1 2010, JIRA will no longer support JBoss application servers.
- From JIRA 4.2, due in Q3 2010, JIRA will no longer support Oracle WebLogic or IBM WebSphere.

We are reducing our application server platform support to reduce the amount of testing time and help us speed up our ability to deliver market-driven features. We are committed to helping our customers understand this decision and assisting them in migrating to Tomcat, our supported Application Server. You have the option of installing the JIRA Standalone version which includes our supported Tomcat application server. For instructions, please see Switching Application Servers to Apache Tomcat.

Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

End of Life Announcement for Application Server Support

Application Servers	Support End Date
JBoss 4.2.2	When JIRA 4.1 releases, due late Q1 2010
Oracle WebLogic 9.2	When JIRA 4.2 releases, due Q3 2010
IBM WebSphere 6.1	When JIRA 4.2 releases, due Q3 2010

- **JBoss End of Support Notes:**

- 'Support End Date' means that JIRA 4.0 and previous released versions will continue to work with JBoss Application Servers. However, we will not fix bugs affecting JBoss application servers.
- JIRA 4.1 will not support JBoss application servers.

- **WebSphere and WebLogic End of Support Notes:**
  - Atlassian is targeting a support end of life for Oracle WebLogic and IBM WebSphere in Q3 2010, with the final support for these platforms in JIRA 4.1.
  - 'Support End Date' means that JIRA 4.1 and previous released versions will continue to work with the stated application servers. However, we will not fix bugs affecting Oracle WebLogic and IBM WebSphere application servers past the support end date.
  - JIRA 4.2 (due to release in Q3 2010) will only be tested with and support Tomcat 5.5 and 6.0.
  - If you have concerns with this end of support announcement, please email `eol-announcement at atlassian dot com`.

Why is Atlassian doing this?

Atlassian is committed to delivering improvements and bug fixes as fast as possible. We are also committed to providing world class support for all the platforms our customers run our software on. However, as the complexity of our applications grows, the cost of supporting multiple platforms increases exponentially. Each new feature has to be tested on several combinations of application servers, with setup and ongoing maintenance of automated tests. At times, 30% of the development team is busy coding solutions for edge cases in various application servers. Moving forward, we want to reduce the time spent there in order to increase JIRA development speed significantly.

We have chosen to standardise on Tomcat, because it is the most widely used application server in our user population. It is fast, robust, secure, well-documented, easy to operate, open source, and has a huge community driving improvements. It is the de facto industry standard, with several companies available that specialise in providing enterprise grade support contracts for it, ranging from customisations to 24/7 support.

Deprecated Java Platforms for JIRA (27 January 2010)

This section announces the end of Atlassian support for certain Java Platforms for JIRA.

We will **stop supporting the following Java Platforms**:

- From JIRA 4.2, due Q3 2010, support for Java Platform 5 (JDK/JRE 1.5) will end.

We are ending support for Java Platform 5, in line with Sun's Java SE Support Road Map (i.e. "End of Service Life" for Java Platform 5 datec October 30, 2009). We are committed to helping our customers understand this decision and assisting them in updating to Java Platform 6, our supported Java Platform.

The details are below. Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

End of Life Announcement for Java Platform Support

Java Platform	Support End Date
Java Platform 5 (JDK/JRE 1.5)	When JIRA 4.2 releases, due Q3 2010

- **Java Platform 5 End of Support Notes:**
  - Atlassian intends to end support for Java Platform 5 in Q3 2010.
  - 'Support End Date' means that JIRA 4.1.x and previous released versions will continue to work with Java Platform 5 (JDK/JRE 1.5). However, we will not fix bugs related to Java Platform 5 past the support end date.
  - JIRA 4.2 (due to release in Q3 2010) will only be tested with and support Java Platform 6 (JDK/JRE 1.6).
  - If you have concerns with this end of support announcement, please email `eol-announcement at atlassian dot com`.

Deprecated Web Browsers for JIRA (11 December 2009)

This section announces the end of Atlassian support for certain web browsers for JIRA.

We will **stop supporting older versions of web browsers** as follows:

- JIRA 4.1 will be the last version of JIRA to support IE6. (From JIRA 4.0 to JIRA 4.1, all of the main functionality will work in IE 6. However, some of the visual effects will be missing.)

The details are below. Please refer to the Supported Platforms for more details regarding platform support for JIRA. If you have questions or

concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

End of Life Announcement for Web Browser Support

Web Browsers	Support End Date
Internet Explorer 6	When JIRA 4.2 releases (target Q3 2010)

- **Internet Explorer 6 End of Support Notes:**
  - JIRA 4.1 (due late Q1 2010) will be the last version to officially support Internet Explorer 6.
  - JIRA 4.2 is currently targeted to release Q3 2010 and will **not** support IE6.
  - This decision was made in line with Microsoft's Support Lifecycle policy, which indicates the official end of support for Internet Explorer 6 on 13th July, 2010. Please note that released versions of JIRA up to and including JIRA 4.1 will continue working with IE6 just as they did before, but we will not fix bugs affecting Internet Explorer 6.
  - You may be able to use Internet Explorer 6 for the most common use cases like viewing and editing content in JIRA 4.1 and earlier, but official support for this browser will end once you upgrade to JIRA 4.2.

Caveats in using Firefox 3.6.0 with JIRA

Overview

A **bug in Firefox 3.6.0** results in this browser version failing to submit form data to JIRA, resulting in data loss. For more information about this bug, please refer to the following links:

- Mozilla forum support posting
- Bug report in JIRA describing the symptoms
- Mozilla's own bug report

Symptoms

If you spend more than 5 minutes either creating or editing an issue in Firefox 3.6.0 and then attempt to submit the issue, you may lose all the data that you just entered or modified and the data changes will not be saved to JIRA. This Firefox 3.6.0 bug may also affect other JIRA screens on which form data has been left for more than 5 minutes.

This Firefox bug will affect you if the following points are true

- You are running Firefox 3.6.0 on Windows or Linux
  AND
- Your Firefox browser's proxy settings have been set to any of the following options:
  - 'Auto-detect proxy settings for this network'
  - 'Manual proxy configuration'
  - 'Automatic proxy configuration URL'

Proxy server configurations for web browsers are typically required in networks where Internet access is monitored or controlled, such as some corporate environments. Your organisation may *require* that your web browser be configured through a proxy server for Internet access.

How can I prevent this bug affecting me?

If you are using Firefox 3.6.0, there are two ways you can avoid this bug:

1. Ensure Firefox's proxy settings have been set to 'No proxy'. (See below for details.)
2. Upgrade Firefox to version 3.6.2.

   Firefox did not officially release a '3.6.1' version.

To determine Firefox's current proxy settings:

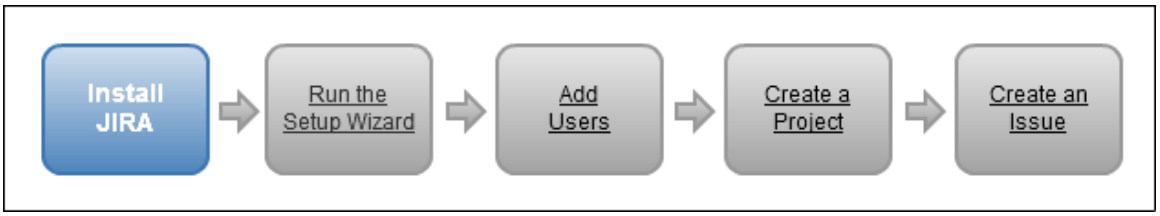
1. Click the '**Tools**' menu in Firefox and then the '**Options**' menu item, which opens the 'Options' dialog box.
2. Click '**Advanced**' and then select the '**Network**' tab.
3. Click the '**Settings**' button, which opens the 'Connection Settings' dialog box.
   - If 'No proxy' is selected in the 'Connection Settings' dialog box, then you do not need to take any further action.
   -
     If any of the other options (listed above) are selected, your network administrator may require that your web browser be configured through a proxy server for Internet access. Please consult your network administrator about this issue before making any further changes to these settings.
4. Click the '**Cancel**' buttons to close the Firefox dialog boxes.

To set Firefox's proxy setting to 'No proxy':

1. Open Firefox's 'Connection Settings' dialog box (as described above).

2. Select the '**No proxy**' option and click the '**OK**' button.

Installing JIRA



Use this Installation Guide if you are installing JIRA for the first time. If you are upgrading JIRA, please refer to the Upgrade Guide.

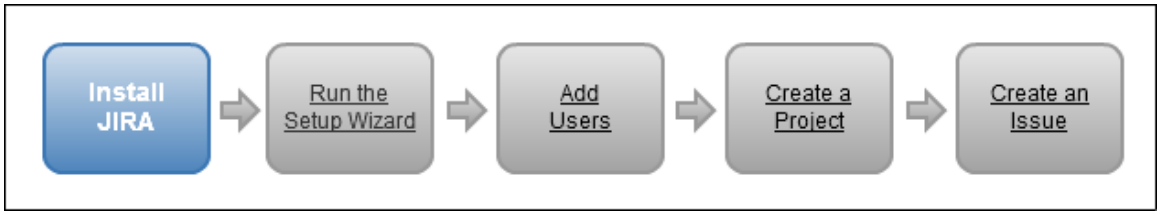
Installing JIRA

To install JIRA, follow the instructions for your operating system:

- Installing JIRA on Windows
- Installing JIRA on Linux
- Installing JIRA on Solaris

Each of the instructions above install 'recommended' distributions of JIRA. Another JIRA distribution known as 'JIRA WAR' is also available for more advanced setups that require the building and deployment of JIRA to a separate application server installation. To install the JIRA WAR distribution, see Installing JIRA WAR.

Installing JIRA on Windows



This guide describes how to install a new JIRA installation on Windows using the automated 'Windows Installer'. If you are upgrading JIRA, please refer to the Upgrading JIRA guide.

You can also install JIRA from a 'zip' archive — see Installing JIRA from an Archive File on Windows, Linux or Solaris for details. This is useful if you want JIRA to use a pre-existing supported Java platform, since the Windows Installer installs its own JRE to run JIRA.

Please Note: Some anti-virus or other Internet security tools may interfere with the JIRA installation process and prevent the process from completing successfully. If you experience or anticipate experiencing such an issue with your anti-virus/Internet security tool, disable thi tool first before proceeding with the JIRA installation.

There are two ways to install JIRA using the Windows Installer:

- Using the Installation Wizard
- Performing an Unattended Installation

On this page:

- Using the Installation Wizard
  - 1. Download and Run the JIRA 'Windows Installer'
  - 2. Starting JIRA
  - 3. Run the Setup Wizard
  - 4. Next Steps
- Performing an Unattended Installation
  - Download and Run the JIRA 'Windows Installer' in Unattended Mode

Using the Installation Wizard

Use the installation wizard if you are installing JIRA on your server for the first time or you wish to specify your installation options.

If you have previously installed JIRA using the installation wizard and wish to re-install JIRA again with the same installation options, you can re-install JIRA in 'unattended mode' without any user input required (see below for details).

1. Download and Run the JIRA 'Windows Installer'

To install JIRA as a service, the Windows Installer must be run using a Windows administrator account. While you can run the Windows Installer with a non-administrator account, your installation options will be much more limited.

1. Download the **JIRA 'Windows Installer'** (.exe) file from the JIRA Download page.
2. Run the '.exe' file to start the installation wizard.
   If a Windows 7 (or Vista) 'User Account Control' dialog box requests if you want to allow the installation wizard to make changes to your computer, specify '**Yes**'. If you do not, the installation wizard will have restricted access to your operating system and any subsequent installation options will be limited.
3. At the 'Upgrading JIRA?' step, choose between the '**Express Install**' or '**Custom Install**' options:
   - **Express Install** — If you choose this option, JIRA will be installed with default settings which are shown in the next step of the installation wizard. If you want to customise any of these options, click the '**Back**' button and choose the '**Custom Install** ' option instead.
   - **Custom Install** — If you choose this option, JIRA will prompt you to specify the following options (which are presented during subsequent steps of the installation wizard and pre-populated with default values):
     - The 'Destination Directory' in which to install JIRA.
     - The JIRA Home directory (which must be unique for each JIRA installation).
     - The Windows 'Start' menu folder options.
     - The *TCP ports* (i.e. an HTTP and a Control port) that JIRA will run through.
     - If you are running the installer using an administrator account, you will be prompted to 'Install JIRA as a service' (recommended). You can also do this manually later, as described in Running JIRA as a Service.
       If you installed JIRA as a service, you must start JIRA through the Windows 'Start' menu, since JIRA will not start if you run `start-jira.bat` at the Windows Command Prompt.
4. The installation wizard will install JIRA onto your operating system and will start JIRA automatically when the wizard finishes. JIRA will also be launched automatically in your browser window if you chose this option.

Please Note:

- If you chose to install JIRA as a service, the JIRA service will be run as the Windows 'SYSTEM' user account. To change this user account, see Changing the Windows user that the JIRA service uses.
- If you do not install JIRA as a service, then once started, JIRA will be run as the Windows user account under which JIRA was installed.
- If you use JIRA running on a Windows Server in production, we strongly recommend creating a dedicated user account (e.g. with username 'jira') for running JIRA.
  - For more information about creating a dedicated user account and defining which directories this account should have write access to, refer to our guidelines.
    If your Windows Server is operating under a Microsoft Active Directory, ask your Active Directory administrator to create a dedicated user account that you can use to run JIRA (with no prior privileges).
  - If JIRA is installed as a service, do not forget to change the user account that runs the JIRA service to your dedicated user account for running JIRA.

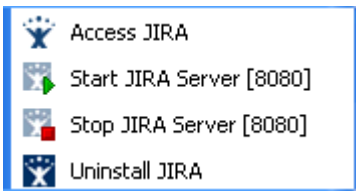
2. Starting JIRA

If JIRA is not already started, you can start JIRA using the appropriate Windows 'Start' menu shortcut or command prompt option.

Once JIRA is started, you can access JIRA from the appropriate Windows 'Start' menu shortcut or a browser on any computer with network access to your JIRA server.

2.1 Windows 'Start' Menu Shortcuts

The Installer will have created the following Windows 'Start' menu shortcuts:



- **Access JIRA** — opens a web browser window to access your JIRA application.
  Your JIRA server must have been started for this shortcut to work.
- **Start JIRA Server** — starts up the Apache Tomcat application server which runs your JIRA installation, so that you can access JIRA through your web browser.
- **Stop JIRA Server** — stops the Apache Tomcat application server which runs your JIRA installation. You will not be able to access JIRA through your web browser after choosing this shortcut.
- **Uninstall JIRA** — uninstalls JIRA from your Windows operating system.

2.2 Starting and Stopping JIRA from a Command Prompt

Enter the `bin` subdirectory of your JIRA installation directory and run the appropriate file:

- `start-jira.bat` (to start JIRA)

- `stop-jira.bat` (to stop JIRA)

If you followed our guidelines for running JIRA with a dedicated user account, then to run JIRA as this user account (e.g. 'jira'), use the `runas` command to execute `start-jira.bat`. For example:

- `> runas /env /user:<DOMAIN>\jira start-jira.bat`
  (where `<DOMAIN>` is your Windows domain or computer name.)

2.3 Accessing JIRA from a Browser

You can access JIRA from any computer with network access to your JIRA server by opening a supported web browser on the computer and visiting this URL:

- `http://<computer_name_or_IP_address>:<HTTP_port_number>`

where:

- `<computer_name_or_IP_address>` is the name or IP address of the computer on which JIRA is installed and
- `<HTTP_port_number>` is the HTTP port number specified when you installed JIRA (above).

If JIRA does not appear in your web browser, you may need to change the port that JIRA runs on.

3. Run the Setup Wizard

See Running the Setup Wizard.

4. Next Steps

- See JIRA 101 to start creating Projects, creating Users, and customising your JIRA instance.
- If you did not install JIRA as a service, you will need to start JIRA manually every time you restart your computer. To change your JIRA installation to run as a service, please see Running JIRA as a Service.
- To get the most out of JIRA, please see Optimising Performance.

Performing an Unattended Installation

If you have previously installed JIRA using the installation wizard (above), you can use a configuration file from this JIRA installation (called `response.varfile`) to re-install 'unattended mode' without any user input required.

Installing JIRA in unattended mode saves you time if your previous JIRA installation was used for testing purposes and you need to install JIRA on multiple server machines based on the same configuration.

Please Note:

- The `response.varfile` file contains the options specified during the installation wizard steps of your previous JIRA installation. Hence, do not uninstall your previous JIRA installation just yet.
- If you intend to modify the `response.varfile` file, please ensure all directory paths specified are absolute, for example, `sys.installationDir=C\:\\Program Files\\Atlassian\\JIRA`
  Unattended installations will fail if any relative directory paths have been specified in this file.

Download and Run the JIRA 'Windows Installer' in Unattended Mode

1. Download the **JIRA 'Windows Installer'** (.exe) file from the JIRA Download Center to a suitable location.
2. Open the Windows command prompt and perform the remaining steps in the command prompt.
3. `copy` the `response.varfile` file located in the `.install4j` subdirectory of your previous JIRA installation directory, to the same location as the downloaded 'Windows Installer' file.
   You can uninstall your previous JIRA installation after this step. Save your `response.varfile` if you need to install JIRA on multiple machines.
4. Change directory (`cd`) to the location of the 'Windows Installer' file and run the following command:

   ```
   atlassian-jira-X.Y.exe -q -varfile response.varfile
   ```

   Where:

   - `X.Y` — refers to the version of JIRA you are about to install.
   - `-q` — instructs the installer to operate in unattended mode (i.e. 'quietly').
   - `-varfile response.varfile` — specifies the configuration file containing the configuration options used by the installer. The location and name of the configuration file should be specified after the `-varfile`option.

5. JIRA will start automatically when the silent installation finishes. Continue from step 2 Starting JIRA (above).

Uninstalling JIRA from Windows

This page describes the procedure for uninstalling JIRA, which had been installed using the Windows Installer.

If you wish to re-install JIRA in 'unattended mode', do not uninstall your previous installation of JIRA just yet. See Using the Silent Installation Feature for more information.

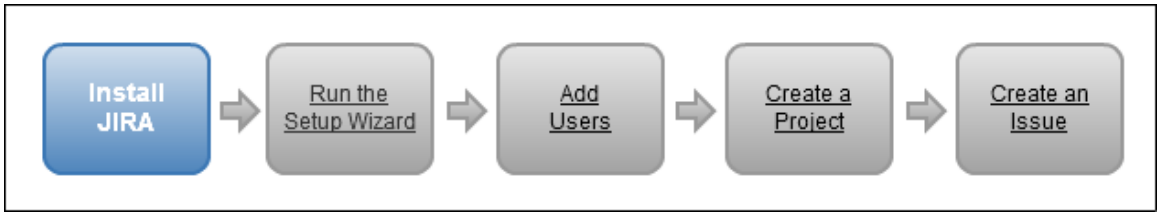
To uninstall JIRA from Windows:

1. Log in to Windows as the same user that was used to install JIRA with the Windows Installer.
2. Start the uninstaller by doing either of the following:
   - Click the Windows '**Start**' menu -> '**All Programs**' -> '**JIRA X.Y**' -> '**Uninstall JIRA X.Y**'
     (where 'X.Y' refers to the installed version of JIRA that you are about to uninstall)
     OR
   - Open the Windows Control Panel, choose '**Add or Remove Programs**' (on Windows XP) or '**Programs and Features**' on (Windows 7/Vista) and then uninstall '**JIRA X.Y**' from the list of applications
     OR
   - Open the Windows command prompt and do the following:
     a. Change directory `cd` to your JIRA installation directory
     b. Run the `uninstall.exe` file
3. Follow the prompts to uninstall JIRA from your computer.

Please note:

- The uninstaller will not delete the JIRA Home Directory.
- All log files that were generated while JIRA was running will not be deleted.
- All files within the JIRA Installation Directory will be deleted (with the exception of the Tomcat `log` folder located in the JIRA Installation Directory).
- The uninstaller can be made to operate in unattended mode by specifying the `-q` option at the Windows command prompt — i.e. `uninstall.exe -q`

Installing JIRA on Linux



This guide describes how to install a new JIRA installation on Linux using the automated 'Linux Installer'. If you are upgrading JIRA, please refer to the Upgrading JIRA guide.

You can also install JIRA from a 'zip' archive — see Installing JIRA from an Archive File on Windows, Linux or Solaris for details. This is useful if you want JIRA to use a pre-existing supported Java platform, since the Linux Installer installs its own JRE to run JIRA.

It is possible that any anti-virus or other Internet security tools installed on your Linux operating system may interfere with the JIRA installation process and prevent the process from completing successfully. If you experience or anticipate experiencing such an issue with your anti-virus/Internet security tool, disable this tool first before proceeding with the JIRA installation.

There are two ways to install JIRA using the Linux Installer:

- Using the Console Wizard
- Performing an Unattended Installation

On this page:

- Using the Console Wizard
  - 1. Download and Install the JIRA 'Linux Installer'
  - 2. Start JIRA
  - 3. Run the Setup Wizard
  - 4. Next Steps
- Performing an Unattended Installation
  - Download and Run the JIRA 'Linux Installer' in Unattended Mode

Using the Console Wizard

Use the console wizard if you are installing JIRA on your server for the first time or you wish to specify your installation options.

If you have previously installed JIRA using the installation wizard and wish to re-install JIRA again with the same installation options, you can re-install JIRA in 'unattended mode' without any user input required (see below for details).

1. Download and Install the JIRA 'Linux Installer'

If you execute the Linux Installer with 'root' user privileges, the installer will create and run JIRA using a dedicated user account. You can also execute the Linux Installer without 'root' user privileges, although your installation options will be much more limited and a dedicated user account (to run JIRA) will not be created. To run JIRA as a service, the Linux Installer must be executed with 'root' user privileges.

1. Download the appropriate **JIRA 'Linux 64-bit / 32-bit Installer'** (.bin) file from the JIRA Download page.
   Please Note:
   - To access the 32-bit installer, you may need to click the '**Show all**' link on the 'JIRA Download' page to access the other installation packages.
   - The difference between the 64-bit / 32-bit .bin installers relates to their bundled Java platforms that run JIRA. Bear in mind that a JIRA installation installed using the 64-bit installer may require additional memory (to run at a similar level of performance) to a JIRA installation installed using the 32-bit installer. This is because a 64-bit Java platform's object references are twice the size as those for a 32-bit Java platform.
2. Open a Linux console and change directory (`cd`) to the '.bin' file's directory.
   If the '.bin' file is not executable after downloading it, make it executable, for example:
   `chmod a+x atlassian-jira-X.Y.bin`
   (where X.Y represents your version of JIRA)
3. Execute the '.bin' file to start the console wizard.
4. When prompted to choose between '**Express Install**', '**Custom Install**' or '**Upgrade an existing JIRA installation**', choose either the '**Express Install**' or '**Custom Install**' options:
   - **Express Install**— If you choose this option, JIRA will be installed with default settings which are shown in the next step of the console wizard.
     Please Note:
     - If you are running the installer with 'root' user privileges, JIRA will be installed as a service.
     - If you want to customise any of these options:
       i. Enter '**e**' to exit the console wizard.
       ii. Execute the console wizard again (step 3 above).
       iii. Choose the '**Custom Install**' option instead.
   - **Custom Install** — If you choose this option, JIRA will prompt you to specify the following options (which are presented during subsequent steps of the console wizard and pre-populated with default values):
     - The 'Destination Directory' in which to install JIRA.
     - The JIRA Home directory (which must be unique for each JIRA installation).
     - The *TCP ports* (i.e. an HTTP and a Control port) that JIRA will run through.
     - If you are running the installer with 'root' user privileges, you will be prompted to 'Run JIRA as a service' (recommended). You can also do this manually later, as described in Starting JIRA Automatically on Linux.
5. The console wizard will install JIRA onto your operating system and will start JIRA automatically when the wizard finishes.

Please Note:

- If you executed the Linux Installer with 'root' user privileges, the Linux Installer creates a dedicated Linux user account with username 'jira', which is used to run JIRA. This account has only:
  - Full write access to your JIRA Home Directory.
  - Limited write access to your JIRA Installation Directory.
- If you executed the Linux Installer without 'root' user privileges, be aware that JIRA can still be run with 'root' privileges. However, to protect the security of your operating system, *this is not recommended*.

2. Start JIRA

If JIRA is not already started, you can start JIRA using the appropriate command at the Linux console.

Once JIRA is started, you can access JIRA from a browser on any computer with network access to your JIRA server.

2.1 Starting and Stopping JIRA manually

In the Linux console, enter the `bin` subdirectory of your JIRA installation directory and execute the appropriate file:

- `start-jira.sh` (to start JIRA)
- `stop-jira.sh` (to stop JIRA)

JIRA will be ready to access (from a browser window) when the following message appears in the application's log file:

```
*********************************************************
 ... You can now access JIRA through your web browser.
*********************************************************
```

2.2 Accessing JIRA from a Browser

You can access JIRA from any computer with network access to your JIRA server by opening a supported web browser on the computer and visiting this URL:

- `http://<computer_name_or_IP_address>:<HTTP_port_number>`

where:

- `<computer_name_or_IP_address>` is the name or IP address of the computer on which JIRA is installed and
- `<HTTP_port_number>` is the HTTP port number specified when you installed JIRA (above).

Please Note:

- If JIRA does not appear, you may need to change the port that JIRA runs on.
- Application server logs (i.e. for Apache Tomcat) will be written to the `logs/catalina-YYYY-MM-DD.log` file within the JIRA Installation Directory.

3. Run the Setup Wizard

See Running the Setup Wizard.

4. Next Steps

- See JIRA 101 to start creating Projects, creating Users, and customising your JIRA instance.
- If you did not install JIRA to run as a service, you will need to start JIRA manually every time you restart your computer. To change your JIRA installation to run as a service, please see Starting JIRA Automatically on Linux.
- To get the most out of JIRA, please see Optimising Performance.

Performing an Unattended Installation

If you have previously installed JIRA using the console wizard (above), you can use a configuration file from this JIRA installation (called `response.varfile`) to re-install JIRA in 'unattended mode' without any user input required.

Installing JIRA in unattended mode saves you time if your previous JIRA installation was used for testing purposes and you need to install JIRA on multiple server machines based on the same configuration.

Please Note:

- The `response.varfile` file contains the options specified during the installation wizard steps of your previous JIRA installation. Hence, do not uninstall your previous JIRA installation just yet.
- If you intend to modify the `response.varfile` file, please ensure all directory paths specified are absolute, for example, `sys.installationDir=/opt/atlassian/jira`
  Unattended installations will fail if any relative directory paths have been specified in this file.

Download and Run the JIRA 'Linux Installer' in Unattended Mode

1. Download the **JIRA 'Linux Installer'** (.bin) file from the JIRA Download Center to a suitable location.
2. Open a Linux console.
3. Copy (`cp`) the file `.install4j/response.varfile` located in your previous JIRA installation directory, to the same location as the downloaded 'Linux Installer' file.
   You can uninstall your previous JIRA installation after this step. Save your `response.varfile` if you need to install JIRA on multiple machines.
4. Change directory (`cd`) to the location of the 'Linux Installer' file and execute the following command:

   ```
   atlassian-jira-X.Y.bin -q -varfile response.varfile
   ```

   Where:

   - `X.Y` — refers to the version of JIRA you are about to install.
   - `-q` — instructs the installer to operate in unattended mode (i.e. 'quietly').
   - `-varfile response.varfile` — specifies the configuration file containing the configuration options used by the installer. The location and name of the configuration file should be specified after the `-varfile`option.
5. JIRA will start automatically when the silent installation finishes. Continue from step 2 Starting JIRA (above).

Uninstalling JIRA from Linux

This page describes the procedure for uninstalling JIRA, which had been installed using the Linux Installer.

If you wish to re-install JIRA in 'unattended mode', do not uninstall your previous installation of JIRA just yet. See Using the Silent Installation Feature for more information.

To uninstall JIRA from Linux:

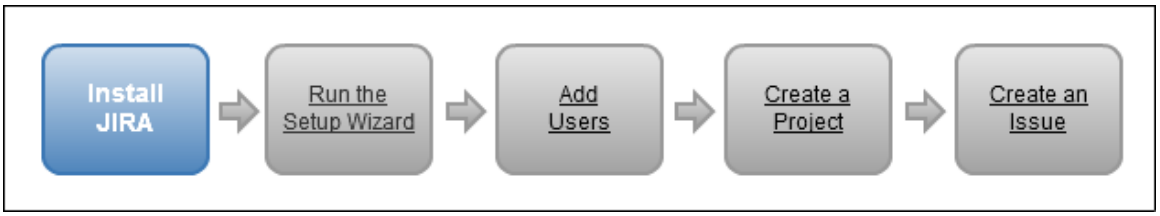
1. Open a Linux console.

2. Change directory (`cd`) to your JIRA installation directory.
3. Execute the command `uninstall`
   This command must be executed as the same user account that was used to install JIRA with the Linux Installer.
4. Follow the prompts to uninstall JIRA from your computer.

Please note:

- The uninstaller will not delete the JIRA Home Directory.
- All log files that were generated while JIRA was running will not be deleted.
- All files within the JIRA Installation Directory will be deleted (with the exception of the Tomcat `log` folder located in the JIRA Installation Directory).
- The uninstaller can be made to operate in unattended mode by specifying the `-q` option — i.e. `uninstall -q`

Installing JIRA from an Archive File on Windows, Linux or Solaris



To install JIRA on Windows from a 'zip' archive file or Linux/Solaris from a 'tar.gz' archive file, follow these steps:

- Before you begin
- 1. Download and Extract the JIRA Archive File
- 2. Set the JIRA Home Directory in JIRA
- 3. Create a Dedicated User Account on the Operating System to Run JIRA
- 4. Start JIRA
- 5. Run the Setup Wizard
- Next Steps

Before you begin

Please ensure that you have installed Java and set JAVA_HOME.

Linux distributions frequently have an open-source implementation of Java called GCJ installed. Do not use this Java platform — it is incomplete and JIRA will not run successfully on it.

1. Download and Extract the JIRA Archive File

1. Download the appropriate JIRA archive file for your operating system ('zip' for Windows or 'tar.gz' for Linux/Solaris), from the JIRA Download page.
   After selecting the appropriate operating system tab on the 'JIRA download' page, you may need to click the '**Show all**' link to access the required installation package.
2. Extract the downloaded file.
   For Windows, we recommend using a file extraction tool such as 7-Zip. **Avoid using Solaris' default tar utility!** Please use **GNU tar** on this operating system to extract JIRA, as GNU tar handles long filenames better.

2. Set the JIRA Home Directory in JIRA

- Use the JIRA Configuration Tool to change the location of your JIRA Home Directory.
  The JIRA Configuration Tool is not available in JIRA WAR distributions.
- Edit the `jira-application.properties` file and set the value of the '`jira.home`' property to the desired location for your JIRA Home Directory. If you are specifying this location's path on Windows, use double back-slashes ("\") between subdirectories. For example, `X:\\path\\to\\JIRA\\Home`
  See the JIRA Installation Directory page to find where this file is located.
- Set an environment variable named `JIRA_HOME` in your operating system whose value is the location of your JIRA Home Directory. To do this:
  - On Windows, do one of the following:
    - Configure this environment variable through the Windows user interface (typically through 'My Computer' or 'Computer')
    - At the command prompt, enter the following command (with your own JIRA Home path) before running JIRA from the command prompt:
      - `set JIRA_HOME=X:\path\to\JIRA\Home`
        Please set your `JIRA_HOME` environment variable value using this format, where:
        - `X` is the drive letter where your JIRA Home Directory is located and
        - no spacing has been added around the equal sign ('=')
    - Specify the command above in a batch file used to start JIRA.
  - On Linux/Solaris, do one of the following:
    - Enter the following command at a shell/console prompt (with your own JIRA Home path) before running JIRA:

    - `export JIRA_HOME=/path/to/jira/home`
  - Specify the command above in a script used to start JIRA.

You can specify any location on a disk for your JIRA home directory. Please be sure to specify an absolute path.

Please note that you cannot use the same JIRA home directory for multiple instances of JIRA. We recommend locating your JIRA Home Directory completely independently of the JIRA Installation Directory (i.e. not nesting one within the other) as this will minimise information being lost during major operations (e.g. backing up and restoring instances).

3. Create a Dedicated User Account on the Operating System to Run JIRA

This step is optional if you are evaluating JIRA but should be mandatory for JIRA installations used in production.

A dedicated user should be created to run JIRA, as JIRA runs as the user it is invoked under and therefore can potentially be abused. For example:

- If your operating system is *nix-based (for example, Linux or Solaris), type the following in a console:
  `$ sudo /usr/sbin/useradd --create-home --comment "Account for running JIRA" --shell /bin/bash jira`
- If your operating system is Windows:
  1. Create the dedicated user account by either:
     - Typing the following at the Windows command line:
       `> net user jira mypassword /add /comment:"Account for running JIRA"`
       (This creates a user account with user name 'jira' and password 'mypassword'. You should choose your own password.)
     - Opening the Windows 'Computer Management' console to add your 'jira' user with its own password.
  2. *(Optional)* Use the Windows 'Computer Management' console to remove the 'jira' user's membership of all unnecessary Windows groups, such as the default 'Users' group.
     If Windows is operating under a Microsoft Active Directory, ask your Active Directory administrator to create your 'jira' account (with no prior privileges).

Ensure that *only the following directories* can be written to by this dedicated user account (e.g. 'jira'):

- The following subdirectories of your JIRA Installation Directory for 'recommended' JIRA distributions (or for JIRA WAR distributions, the installation directory of the Apache Tomcat application running JIRA):
  - `logs`
  - `temp`
  - `work`
- Your JIRA Home Directory.

Do not make the JIRA Installation Directory itself writeable by the dedicated user account.

See also Tomcat security best practices.

4. Start JIRA

Enter the `bin` subdirectory of your JIRA installation directory and execute the appropriate file to start running JIRA:

- `start-jira.sh` (on Linux/Solaris)
- `start-jira.bat` (on Windows)

To run JIRA as the dedicated user account (e.g. 'jira') created above:

- On Windows, use the `runas` command to run `start-jira.bat`. For example,
  `runas /env /user:<DOMAIN>\jira start-jira.bat`
  (where `<DOMAIN>` is your Windows domain or computer name.)
- On Linux, switch to the 'jira' account using the `su` command before running `start-jira.sh` (or use `su` to run `start-jira.sh` as the 'jira' account).

Wait until the following message appears in the application's log file:

```
*************************************************
 ... You can now access JIRA through your web browser.
*************************************************
```

You can access JIRA from any computer with network access to your JIRA server by opening a supported web browser on the computer and visiting this URL:

- `http://<computer_name_or_IP_address>:<HTTP_port_number>`

where:

- `<computer_name_or_IP_address>` is the name or IP address of the computer on which JIRA is installed and
- `<HTTP_port_number>` is the HTTP port number (8080 by default).

If JIRA does not appear in your web browser, you may need to change the port that JIRA runs on.

Logs will be written to `logs/catalina.out`.

> If something goes wrong, please verify that Java is installed correctly. If the problem persists, please contact us — we're happy to help.

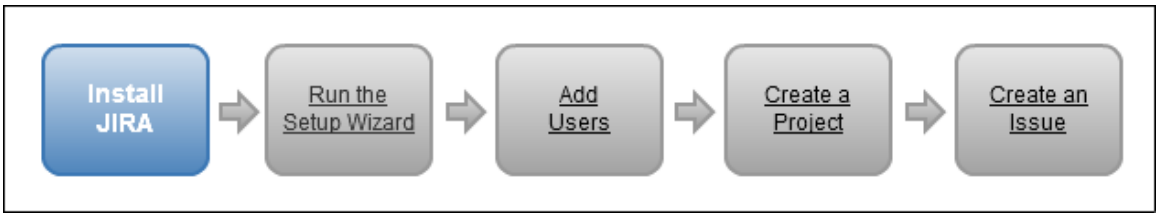
5. Run the Setup Wizard

See Running the Setup Wizard.

Next Steps

- See JIRA 101 to start creating Projects, creating Users, and customising your JIRA instance.
- If you like to set up JIRA to start automatically every time you restart your computer, please see Starting JIRA Automatically on Linux or Running JIRA as a Service (for Windows).
- By default, JIRA installed from an archive uses the standard Tomcat port (i.e. 8080). If you need another application to run on that port, either now or in the future, please see Changing JIRA's TCP Ports.
- To get the most out of JIRA, please see Optimising Performance.

Installing JIRA WAR



What is the JIRA WAR distribution?

JIRA is available in two types of **'distributions'**:

- 'Recommended' distributions (which include JIRA installations installed using the 'Windows Installer', 'Linux Installer' or from an ' Archive File')
  AND
- The '**WAR**' distribution for advanced or highly configured JIRA installations.

Recommended distributions	WAR distribution
Require minimal setup	Requires manual configuration
Pre-packaged with the Apache Tomcat application server	Requires building and deployment to an existing application server installation
Include the JIRA Configuration Tool	Does not include the JIRA Configuration Tool
Recommended for all users	Suitable only for system administrators

We recommend installing the 'recommended' distributions of JIRA over the WAR distribution — even for organisations with an existing application server environment.

To install a recommended distribution of JIRA, see Installing JIRA.

Installing JIRA WAR

To install JIRA WAR, follow the instructions for your application server:

- Installing JIRA on Tomcat 6.0
- Installing JIRA on Tomcat 5.5

Additional Notes

- Read the JIRA WAR Configuration Overview. This contains important configuration information, regardless of your application server.
- Read Switching Application Servers to Apache Tomcat if you are currently running JIRA on a non-Tomcat application server. Please be aware of JIRA's Supported Platforms page, which indicates what application servers are supported by JIRA.
- Also be aware that we do not recommend Deploying Multiple Atlassian Applications in a Single Tomcat Container.

JIRA WAR Configuration Overview

While the individual server install guides provide specific instructions, it is useful to have an overall conceptual overview of what the configuration process involves.

Webapp layout

After downloading and extracting the JIRA WAR web application archive, your are presented with a directory containing:

```
appendcp.bat
build.bat
build.sh
build.xml
edit-webapp/
etc/
licenses/
readme.txt
src/
tools/
updater/
webapp/
```

Directories are indicated by an appending slash symbol.

Please be aware: The `build.xml` file is an Ant file, which when invoked with the `build.sh` / `build.bat` script, will construct deployable web application archive (`.war`) files (for supported application servers). The `build.xml` file copies the contents of the `webapp` subdirectory of your JIRA Installation Directory and overwrites it with the contents of the sibling `edit-webapp` directory, when constructing its `.war` files. Thus, unless otherwise requested, never edit files within this `webapp` directory!
If a file needs editing, first copy it from `webapp/path/to/file` to `edit-webapp/path/to/file` subdirectories of your JIRA Installation Directory and edit it in the latter location.

Database Configuration

If you are setting up a new installation of JIRA WAR, the JIRA setup wizard will configure a direct JDBC connection to a new JIRA database. Upon completing the setup wizard, your database configuration will be defined and saved into a `dbconfig.xml` file located at the root of your JIRA Home Directory.

If you are upgrading JIRA WAR, please ensure a `dbconfig.xml` file (defining your appropriate database configuration) has been created at the root of your JIRA Home Directory *before* your new JIRA WAR installation is started. This is ensured if you follow the appropriate manual or migration procedures for upgrading JIRA. For specific database configuration details, refer to the appropriate **Configure the database connection manually** sections of the specific database connection instructions in the Connecting JIRA to a Database section of this documentation.

Transaction Manager configuration

J2EE-based applications also rely on a Transaction Manager to coordinate updates across multiple databases. While JIRA currently does not use this facility, JIRA's underlying library (OfBiz) still requires a transaction manager object. As with database connections, this is provided by the application server as a **javax.transaction.UserTransaction** object, which is looked up via JNDI.

Summing Up Database Configuration for JIRA WAR

Hence, JIRA needs to know at least three things:

1. What type of database JIRA is dealing with.
2. The JNDI address of a Transaction Manager (a **javax.transaction.UserTransaction** object).
3. You database connection details.

Points 1 and 2 are configured in the `entityengine.xml` file, as described in Configuring the Entity Engine for JIRA. An editable copy of the `entityengine.xml` file is located in `edit-webapp/WEB-INF/classes`.

Point 3 is configured in the `dbconfig.xml` file. The content of this file is generated after completing the JIRA setup wizard.

User management configuration

JIRA stores all user profiles in a database table. Occasionally, integration with external user management systems like LDAP is required. See The LDAP Integration guide for more information.

Configuring the Entity Engine for JIRA

The Entity Engine from the OFBiz project is what JIRA uses to persist data to a database. You can find out more about why we chose the EE at the bottom of this page. See the configuration overview for a conceptual overview of what is being done here.

On this page:

- Configuring the Entity Engine for JIRA
  - Transaction Factory
- Altering the Entity Model
- Why we chose the Entity Engine

Configuring the Entity Engine for JIRA

The configuration of the Entity Engine is done through an XML file called `entityengine.xml`. This file is used to define parameters for persistence servers.

For JIRA WAR distributions, this file is located in the `edit-webapp/WEB-INF/classes/entityengine.xml` subdirectory of the JIRA Installation Directory.

Ensure that your `entityengine.xml` XML file is **well-formed** when making changes. Some application server configurations may "swallow" the error messages you should get in your log file if `entityengine.xml` is not well-formed and instead, report spurious error messages.

Transaction Factory

By default the Entity Engine tries to obtain a JTA transaction factory from the application server using JNDI. The code sample(s) below show the different values for Apache Tomcat application servers.

Tomcat 5.5 (see also Installing JIRA on Tomcat 5.5):

```
<transaction-factory class="org.ofbiz.core.entity.transaction.JNDIFactory">
  <user-transaction-jndi jndi-server-name="default"
    jndi-name="java:comp/env/UserTransaction"/>
  <transaction-manager-jndi jndi-server-name="default"
    jndi-name="java:comp/env/UserTransaction"/>
</transaction-factory>
```

Altering the Entity Model

The Entity Model describes the table and column layout that JIRA uses in a database. It can be completely altered without changing any of the internal workings of JIRA.

The model provided should work with almost any database (care has been taken to ensure the column and table names are SQL compliant).

The entity model is configured through an XML file called `entitymodel.xml` (located in the `webapp/WEB-INF/classes/entitydefs/entitymodel.xml` subdirectory of JIRA WAR distribution's Installation Directory). To edit this file, copy it to the `edit-webapp/WEB-INF/classes/entitydefs/entitymodel.xml` subdirectory and make changes there. When the WAR is built using `build.(sh|bat)`, the version of the file in the `edit-webapp` subdirectory will be used.

The format of the file is fairly self explanatory. Essentially, JIRA always refers to the `entity-name` and `field-name` attributes within the code. The `type` attribute of a `<field>` tag should always match the `type` attribute of a `<field-type-def>` tag in your `fieldtype-*.xml` files.

To change where entities and fields are persisted in your database, simply add or edit the attribute `table-name` (for entities) or `col-name` (for fields).

Why we chose the Entity Engine

We chose the EE over CMPorBMPentity beans because:

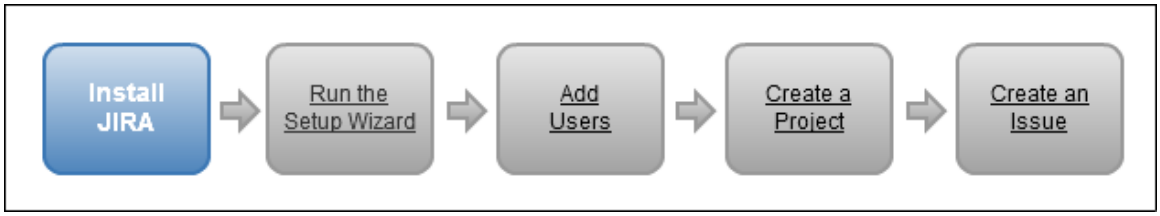
- it is more portable between application servers
- table schemas are automatically created and updated
- using the field type definitions, we can add support for new databases very quickly

- it is faster than most CMP implementations and has some nice caching features

This document deals with configuring the entity engine for JIRA (but should be applicable to most applications). For more details on the entity engine itself and it's inner workings, see:

- OFBiz Entity Engine Guide describes the theory behind the entity engine, its architecture and usage patterns
- OFBiz Entity Engine configuration guide describes all of the entity engine configuration options, whereas this document just describes configuring the entity engine for JIRA

Installing JIRA on Tomcat 6.0



This guide describes how to install JIRA WAR on **Tomcat 6.0**, a popular open-source server from the Apache project. Tomcat can be downloaded from the Apache site.

JIRA installations which have been installed using the 'Windows Installer', 'Linux Installer' or from an 'Archive File' are pre-configured to use their own dedicated Tomcat application server. To install JIRA using one of these distributions, follow the Installing JIRA guide instead of the instructions below.

Before You Begin

Please read the following important notes before you begin installing JIRA on Tomcat 6.0:

- Tomcat 6.0.24 contains a critical bug. Please use 6.0.32 instead.
- Deploying multiple Atlassian applications in a single Tomcat container is **not supported.** We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration (see this FAQ for more information).

  There are also a number of practical reasons why we do not support deploying multiple Atlassian applications in a single Tomcat container. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

  Finally, we recommend not deploying *any other applications* in the same Tomcat container that runs JIRA, especially if these other applications have large memory requirements or require additional libraries in Tomcat's `lib` subdirectory.

On this page:

- Before You Begin
- 1. Download and Extract the JIRA Archive
- 2. Configure JIRA
- 3. Build JIRA
- 4. Update Your Tomcat Installation's Libraries for JIRA
- 5. Configure JIRA's Context in Tomcat
- 6. Modify Tomcat's `server.xml` to Handle Internationalised Characters Correctly
- 7. Fix Memory and Mail Handling Settings in Tomcat
- 8. Start Tomcat
- 9. Run the Setup Wizard
- Troubleshooting
- User-contributed notes

1. Download and Extract the JIRA Archive

Download the **JIRA WAR distribution** archive from the JIRA Download page and extract its contents using a tool such as 7-zip for Windows or Linux's unzip or GNU tar tools.
✅ You may need to click the 'Show All' link on the download page to reveal the **WAR distribution**.

⚠️ **Avoid using Windows' built-in file extraction tool!** This tool silently fails to extract files with long names (see JRA-2153). Other users have also reported problems with WinRAR. ⚠️ **Avoid using Solaris' default tar utility!** Please use **GNU tar** on this operating system to extract JIRA, as GNU tar handles long filenames better.

The extracted directory is hereafter referred to as your JIRA Installation Directory.

A dedicated user should be created to run JIRA, as JIRA runs as the user it is invoked under and therefore can potentially be abused. For example:

- If your operating system is *nix-based (for example, Linux or Solaris), type the following in a console:

```
$ sudo /usr/sbin/useradd --create-home --comment "Account for running JIRA" --shell /bin/bash
jira
```

- If your operating system is Windows:
  1. Create the dedicated user account by either:
     - Typing the following at the Windows command line:
       `> net user jira mypassword /add /comment:"Account for running JIRA"`
       (This creates a user account with user name 'jira' and password 'mypassword'. You should choose your own password.)
     - Opening the Windows 'Computer Management' console to add your 'jira' user with its own password.
  2. *(Optional)* Use the Windows 'Computer Management' console to remove the 'jira' user's membership of all unnecessary Windows groups, such as the default 'Users' group.
     If Windows is operating under a Microsoft Active Directory, ask your Active Directory administrator to create your 'jira' account (with no prior privileges).

To maximise security, ensure that this user can only write to the `logs`, `temp` and `work` directories of your application server (Apache Tomcat) installation and your JIRA Home Directory.

2. Configure JIRA

2.1 Customising Your JIRA Installation Directory Files

(This section is optional and recommended for experts only.)

2.1.1 How to Customise Files in Your JIRA Installation Directory

If you wish to customise any files in the `<jira-application-dir>` (i.e. the `webapp` subdirectory) of your JIRA Installation Directory, please perform them in the sibling `edit-webapp` subdirectory only.

To edit a file within the `webapp` subdirectory, first copy it from the `webapp/path/to/file` subdirectory to the `edit-webapp/path/to/file` subdirectory of your JIRA Installation Directory and edit it in the latter location.

When building JIRA (below), `.war` files are constructed based on file contents copied from the `webapp` subdirectory of your JIRA Installation Directory, which are overwritten by file contents from the sibling `edit-webapp` directory. Thus, never edit files within this `webapp` directory!

Be aware that the more files you customise in your JIRA Installation Directory, the more difficult it will be to upgrade JIRA or migrate JIRA to another server, as your customisations will need to be migrated manually over to your new JIRA installation.

2.1.2 Configuring the `entityengine.xml` File

Ensure that the Transaction Factory has been specified correctly in JIRA's `entityengine.xml` file. For more information, see Configuring the Entity Engine for JIRA.

- In the `entityengine.xml` file (located in `edit-webapp/WEB-INF/classes/` of the JIRA Installation Directory), ensure the `<transaction-factory>...</transaction-factory>` tag contains:

```
<transaction-factory class="org.ofbiz.core.entity.transaction.JNDIFactory">
    <user-transaction-jndi jndi-server-name="default"
jndi-name="java:comp/env/UserTransaction"/>
    <transaction-manager-jndi jndi-server-name="default"
jndi-name="java:comp/env/UserTransaction"/>
</transaction-factory>
```

  See Configuring the Entity Engine for JIRA for more information about configuring JIRA's database access layer.

2.2 JIRA Home

- Use the JIRA Configuration Tool to change the location of your JIRA Home Directory.
  The JIRA Configuration Tool is not available in JIRA WAR distributions.
- Edit the `jira-application.properties` file and set the value of the '`jira.home`' property to the desired location for your JIRA Home Directory. If you are specifying this location's path on Windows, use double back-slashes ("\") between subdirectories. For example, `X:\\path\\to\\JIRA\\Home`
  See the JIRA Installation Directory page to find where this file is located.
- Set an environment variable named `JIRA_HOME` in your operating system whose value is the location of your JIRA Home Directory. To do this:
  - On Windows, do one of the following:
    - Configure this environment variable through the Windows user interface (typically through 'My Computer' or 'Computer')
    - At the command prompt, enter the following command (with your own JIRA Home path) before running JIRA from the command prompt:
      - `set JIRA_HOME=X:\path\to\JIRA\Home`
        Please set your `JIRA_HOME` environment variable value using this format, where:

        - X is the drive letter where your JIRA Home Directory is located and
        - no spacing has been added around the equal sign ('=')
    - Specify the command above in a batch file used to start JIRA.
- On Linux/Solaris, do one of the following:
    - Enter the following command at a shell/console prompt (with your own JIRA Home path) before running JIRA:
        - `export JIRA_HOME=/path/to/jira/home`
    - Specify the command above in a script used to start JIRA.

You can specify any location on a disk for your JIRA home directory. Please be sure to specify an absolute path.

Please note that you cannot use the same JIRA home directory for multiple instances of JIRA. We recommend locating your JIRA Home Directory completely independently of the JIRA Installation Directory (i.e. not nesting one within the other) as this will minimise information being lost during major operations (e.g. backing up and restoring instances).

For more information about setting up your JIRA Home Directory, please see Setting your JIRA Home Directory.

3. Build JIRA

Now build JIRA by running `build.bat` (Windows) or `./build.sh` (Linux/Solaris) on the command line in the JIRA Installation Directory. This will produce the deployable WAR file in the `dist-tomcat/tomcat-6` subdirectory of the JIRA Installation Directory.

4. Update Your Tomcat Installation's Libraries for JIRA

4.1 JDBC Drivers

Your Tomcat installation requires an appropriate JDBC driver to allow JIRA to communicate with the database. To add this JDBC driver to Tomcat, refer to the appropriate instructions:

- Copy the PostgreSQL JDBC Driver to Tomcat
- Copy the MySQL JDBC Driver to Tomcat
- Copy the Oracle JDBC Driver to Tomcat
- Copy the SQL Server 2005 JDBC Driver to Tomcat
- Copy the SQL Server 2008 JDBC Driver to Tomcat
- Copy the HSQL JDBC Driver to Tomcat
  If you intend to use the HSQL database, the HSQL JDBC driver is already included with the other JIRA library files that will be added to Tomcat in the following step.

4.2 Other JIRA Libraries for Tomcat

Tomcat does not come with some libraries required to run JIRA. To fix this, download http://www.atlassian.com/software/jira/downloads/binary/jira-jars-tomcat-distribution-5.0-rc2-tomcat-6x.zip extract and copy the `.jar` library files from this archive to the `lib` subdirectory of your Tomcat installation directory.

Please Note:

- Be sure to remove existing versions of these `.jar` library files before copying over new ones.
- To prevent exceptions related to logging, please ensure that the following files are present in Tomcat's `lib` directory. Also ensure that these files *are not present* in the `webapp/jira/WEB-INF/lib` subdirectory of the JIRA Installation Directory. If any of the following files are present in the `webapp/jira/WEB-INF/lib` subdirectory, remove them and rebuild the deployable JIRA WAR file as described in the previous step (above).

File
`jcl-over-slf4j-x.y.z.jar`
`jul-to-slf4j-x.y.z.jar`
`log4j-x.y.z.jar`
`slf4j-api-x.y.z.jar`
`slf4j-log4j12-x-y-z.jar`

5. Configure JIRA's Context in Tomcat

A JIRA 'context' now needs to be set up in Tomcat. To do this:

1. Create the directory structure `conf/Catalina/localhost/` within your Tomcat installation directory.
2. Copy the `jira.xml` file from the `dist-tomcat/tomcat-6` subdirectory of your JIRA Installation Directory to the `conf/Catalina/localhost` subdirectory of your Tomcat installation directory (created in the previous step).
3. If necessary, customise the copied `jira.xml` file within your Tomcat installation — in particular, the value of the `docBase` attribute of the `Context`element:

```
<Context path="/jira" docBase="path/to/atlassian-jira-4.x.war" debug="0"
useHttpOnly="true">

  <Resource name="UserTransaction" auth="Container"
type="javax.transaction.UserTransaction"
    factory="org.objectweb.jotm.UserTransactionFactory" jotm.timeout="60"/>
  <Manager pathname=""/>

</Context>
```

The paths (denoted as `path/to/`) will be correct by default, assuming you want to deploy the `.war` from the `dist-tomcat/tomcat-6` subdirectory of your JIRA Installation Directory.

If you are installing in Windows, make sure that the paths you specify for the location of the WAR file and database are full paths with drive letters (e.g. `C:\path\to\atlassian-jira-4.x.war`).

6. Modify Tomcat's `server.xml` to Handle Internationalised Characters Correctly

In order for JIRA to correctly display internationalised characters in user and group names, you need to modify the `conf/server.xml` file in your Tomcat installation directory by specifying the `URIEncoding="UTF-8"` property within the connector definition for your HTTP protocol.

The connector definition is specified by the following element in your `server.xml` file:

```
<Connector port="8080" protocol="HTTP/1.1"
        connectionTimeout="20000"
        redirectPort="8443"/>
```

You should modify this element by specifying the `URIEncoding="UTF-8"` attribute:

```
<Connector port="8080" protocol="HTTP/1.1"
        connectionTimeout="20000"
        redirectPort="8443" URIEncoding="UTF-8"/>
```

Please Note:

- Since this property must be specified at the connector level for your application server, this setting will effect all other web applications deployed to the same application server installation running JIRA. While this setting should not adversely effect these other web applications, you should be aware of this point.
- JIRA will run fine without this property set. However, you will run into issues if a user or group is created which contains international characters. Hence, it is recommended that you set this property.

7. Fix Memory and Mail Handling Settings in Tomcat

Memory and mail handling settings need to be modified in Tomcat to avoid the following issues:

- Tomcat effectively leaks memory by caching JSPs. This can result in `OutOfMemoryError` errors if large pages (such as RSS or Excel pages) are requested.
- JIRA requires more memory than what Tomcat provides by default. This may lead to `OutOfMemory` errors when running JIRA if these memory settings are not increased.
- For JIRA's mail handler to avoid problems with RFC 2231-compliant mail clients, set the `mail.mime.decodeparameters` startup parameter in Tomcat to `true`.

To prevent these issues, follow the appropriate instructions for your operating system below.

For Windows

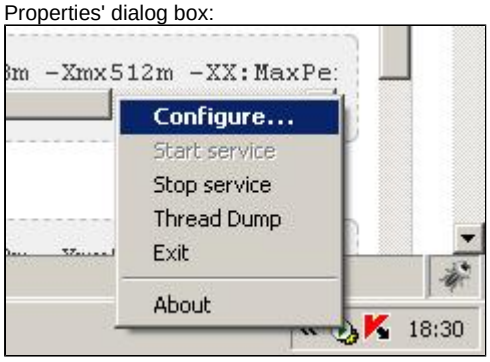
If Tomcat is not installed as a service:

- Edit Tomcat's `bin/setenv.bat`file (or create this file if it does not exist) and add the following to this file:

```
set CATALINA_OPTS=%CATALINA_OPTS%
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true
-Dmail.mime.decodeparameters=true -Xms128m -Xmx512m -XX:MaxPermSize=256m
```

If Tomcat is installed and running as a service:

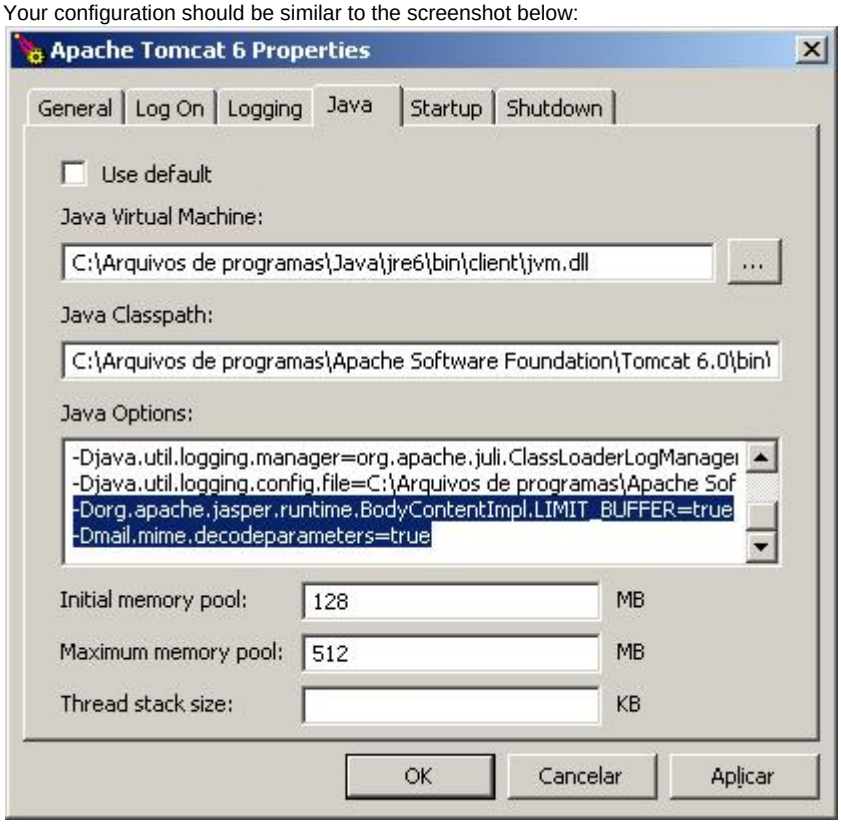
1. Right-click Tomcat's system tray icon and select '**Configure**' from the resulting popup menu, which opens the 'Apache Tomcat 6 Properties' dialog box:



2. In this dialog box, click on the '**Java**' tab and specify the following values:

Field	Value
Java Options (append to the existing value)	-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true -Dmail.mime.decodeparameters=true
Initial memory pool	128
Maximum memory pool	512

Your configuration should be similar to the screenshot below:



For Linux/Solaris

Edit Tomcat's `bin/setenv.sh` file (or create this file if it does not exist) and add the following to this file:

```
export CATALINA_OPTS="$CATALINA_OPTS
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true -Dmail.mime.decodeparameters=true
-Xms128m -Xmx512m -XX:MaxPermSize=256m"
```

For other environments and more information on memory settings, see Increasing JIRA Memory.

8. Start Tomcat

(Only required if Tomcat is not running as a service.)

JIRA should now be ready to run in Tomcat. To start up JIRA, start (or restart) the Tomcat server with Tomcat's `bin/startup.sh` or `bin/startup.bat` scripts.

9. Run the Setup Wizard

Point your browser to

http://localhost:8080/jira

You should now see the Setup Wizard, which will take you through the JIRA's setup procedure, including a configuration step for your database connection.

Troubleshooting

It is easy to make a mistake in this process. First, check that you have followed the process described above:

- Have you have made changes to `edit-webapp/WEB-INF/classes/entityengine.xml` in your JIRA Installation Directory (step 2 above) and re-run the build script (step 3 above), but your `entityengine.xml` changes were not picked up? If so, delete the `webapps/jira` subdirectory of your Tomcat installation directory and then restart JIRA. (In some circumstances, Tomcat does not correctly re-expand the web application.)
- If you are using an external database, did you copy the correct JDBC driver jar file to the `lib` subdirectory of your Tomcat installation directory? (Refer to step 4 above.)
- Have you updated your Tomcat installation's libraries for JIRA by copying across the additional jar files downloaded in step 4 above? Check if `objectweb-datasource-x.y.z.jar` present in the `lib` subdirectory of Tomcat's installation directory.
- Is the path to your built `.war` file within `conf/Catalina/localhost/jira.xml` of your Tomcat installation directory correct? (Refer to step 5 above.)
- Have you copied your built `.war` file to Tomcat's `webapps` directory? This is almost guaranteed to cause problems - please move this `.war` file elsewhere and delete any JIRA subdirectories created in Tomcat's `webapps` directory which Tomcat may have created (after Tomcat is initially started).
- Have you configured JIRA's context and other custom settings centrally in Tomcat's `conf/server.xml` file instead of the `conf/Catalina/localhost/jira.xml` file of your Tomcat installation directory? Although this is fine, be sure that you *do not* also have a `conf/Catalina/localhost/jira.xml` file present.
- The log files are usually vital to debugging problems. On Windows, these will appear in the console window that loads when running `startup.bat`, or in one of the log files in Tomcat's `logs` directory. On Linux/Solaris, logs will appear in a log file in `logs`, usually `logs/*` (not just `logs/catalina.out`). Check the log file for errors after startup.
- If you experience high memory usage / memory leaks (e.g. `OutOfMemoryError`), you may wish to set the system property `-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true` in the `setenv.sh` / `setenv.bat` file. For more information please see JRA-10145. (Refer to step 7 above.)
- If the connection to your database is dropping out (in particular with MySQL), you will need to set up Tomcat to survive connection closures.
- ⚠️ **Please be aware:** The `build.xml` file is an Ant file, which when invoked with the `build.sh` / `build.bat` script, will construct deployable web application archive (`.war`) files (for supported application servers). The `build.xml` file copies the contents of the `webapp` subdirectory of your JIRA Installation Directory and overwrites it with the contents of the sibling `edit-webapp` directory, when constructing its `.war` files. Thus, unless otherwise requested, never edit files within this `webapp` directory!
  If a file needs editing, first copy it from `webapp/path/to/file` to `edit-webapp/path/to/file` subdirectories of your JIRA Installation Directory and edit it in the latter location.

If you are stuck, please raise a support request. In your request, please attach your logs, configuration files, plus anything else relevant and we will get back to you as soon as possible.

User-contributed notes

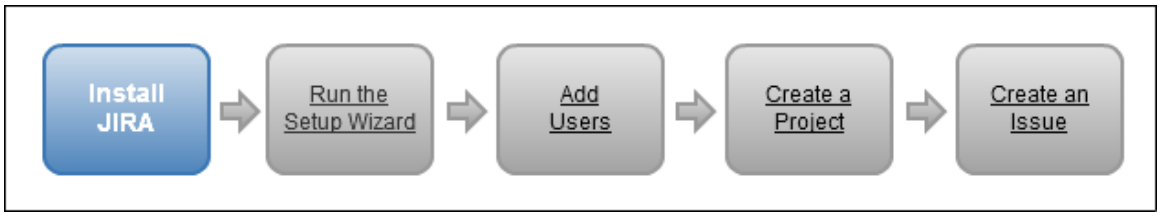
Do you have experiences to share with Tomcat 6.0.x and JIRA? We welcome your thoughts. Please see the user-contributed Tomcat 6.0.x notes.

Tomcat 6.0 notes

This page has general notes on installing JIRA on Tomcat 6.0.x. It supplements the official Tomcat installation docs.

Add your notes

Installing JIRA on Tomcat 5.5



This guide describes how to install JIRA WAR on **Tomcat 5.5**, a popular open-source server from the Apache project. Tomcat can be downloaded from the Apache site.

JIRA installations which have been installed using the 'Windows Installer', 'Linux Installer' or from an 'Archive File' are pre-configured to use their own dedicated Tomcat application server. To install JIRA using one of these distributions, follow the Installing JIRA guide instead of the instructions below.

Before You Begin

- Please use Tomcat 5.5.15 or later.
- Deploying multiple Atlassian applications in a single Tomcat container is **not supported.** We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration (see this FAQ for more information).

  There are also a number of practical reasons why we do not support deploying multiple Atlassian applications in a single Tomcat container. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

  Finally, we recommend not deploying *any other applications* in the same Tomcat container that runs JIRA, especially if these other applications have large memory requirements or require additional libraries in Tomcat's `lib` subdirectory.

On this page:

- Before You Begin
- 1. Download and Extract the JIRA Archive
- 2. Configure JIRA
- 3. Build JIRA
- 4. Update Your Tomcat Installation's Libraries for JIRA
- 5. Configure JIRA's Context in Tomcat
- 6. Modify Tomcat's `server.xml` to Handle Internationalised Characters Correctly
- 7. Fix Memory and Mail Handling Settings in Tomcat
- 8. Start Tomcat
- 9. Run the Setup Wizard
- Troubleshooting
- User-contributed notes

1. Download and Extract the JIRA Archive

Download the **JIRA WAR distribution** archive from the JIRA Download page and extract its contents using a tool such as 7-zip for Windows or Linux's unzip or GNU tar tools.
You may need to click the 'Show All' link on the download page to reveal the **WAR distribution**.

Avoid using Windows' built-in file extraction tool! This tool silently fails to extract files with long names (see JRA-2153). Other users have also reported problems with WinRAR. **Avoid using Solaris' default tar utility!** Please use **GNU tar** on this operating system to extract JIRA, as GNU tar handles long filenames better.

The extracted directory is hereafter referred to as your JIRA Installation Directory.

A dedicated user should be created to run JIRA, as JIRA runs as the user it is invoked under and therefore can potentially be abused. For example:

- If your operating system is *nix-based (for example, Linux or Solaris), type the following in a console:
  ```
  $ sudo /usr/sbin/useradd --create-home --comment "Account for running JIRA" --shell /bin/bash jira
  ```
- If your operating system is Windows:
  1. Create the dedicated user account by either:
     - Typing the following at the Windows command line:
       ```
       > net user jira mypassword /add /comment:"Account for running JIRA"
       ```
       (This creates a user account with user name 'jira' and password 'mypassword'. You should choose your own password.)
     - Opening the Windows 'Computer Management' console to add your 'jira' user with its own password.
  2. *(Optional)* Use the Windows 'Computer Management' console to remove the 'jira' user's membership of all unnecessary Windows groups, such as the default 'Users' group.
     If Windows is operating under a Microsoft Active Directory, ask your Active Directory administrator to create your 'jira'

account (with no prior privileges).

To maximise security, ensure that this user can only write to the `logs`, `temp` and `work` directories of your application server (Apache Tomcat) installation and your JIRA Home Directory.

2. Configure JIRA

2.1 Customising Your JIRA Installation Directory Files

(This section is optional and recommended for experts only.)

2.1.1 How to Customise Files in Your JIRA Installation Directory

If you wish to customise any files in the `<jira-application-dir>` (i.e. the `webapp` subdirectory) of your JIRA Installation Directory, please perform them in the sibling `edit-webapp` subdirectory only.

To edit a file within the `webapp` subdirectory, first copy it from the `webapp/path/to/file` subdirectory to the `edit-webapp/path/to/file` subdirectory of your JIRA Installation Directory and edit it in the latter location.

When building JIRA (below), `.war` files are constructed based on file contents copied from the `webapp` subdirectory of your JIRA Installation Directory, which are overwritten by file contents from the sibling `edit-webapp` directory. Thus, never edit files within this `webapp` directory!

Be aware that the more files you customise in your JIRA Installation Directory, the more difficult it will be to upgrade JIRA or migrate JIRA to another server, as your customisations will need to be migrated manually over to your new JIRA installation.

2.1.2 Configuring the `entityengine.xml` File

Ensure that the Transaction Factory has been specified correctly in JIRA's `entityengine.xml` file. For more information, see Configuring the Entity Engine for JIRA.

- In the `entityengine.xml` file (located in `edit-webapp/WEB-INF/classes/` of the JIRA Installation Directory), ensure the `<transaction-factory>...</transaction-factory>` tag contains:

```
<transaction-factory class="org.ofbiz.core.entity.transaction.JNDIFactory">
    <user-transaction-jndi jndi-server-name="default"
jndi-name="java:comp/env/UserTransaction"/>
    <transaction-manager-jndi jndi-server-name="default"
jndi-name="java:comp/env/UserTransaction"/>
</transaction-factory>
```

  See Configuring the Entity Engine for JIRA for more information about configuring JIRA's database access layer.

2.2 Set JIRA Home

- Use the JIRA Configuration Tool to change the location of your JIRA Home Directory.
  The JIRA Configuration Tool is not available in JIRA WAR distributions.
- Edit the `jira-application.properties` file and set the value of the '`jira.home`' property to the desired location for your JIRA Home Directory. If you are specifying this location's path on Windows, use double back-slashes ("\") between subdirectories. For example, `X:\\path\\to\\JIRA\\Home`
  See the JIRA Installation Directory page to find where this file is located.
- Set an environment variable named `JIRA_HOME` in your operating system whose value is the location of your JIRA Home Directory. To do this:
  - On Windows, do one of the following:
    - Configure this environment variable through the Windows user interface (typically through 'My Computer' or 'Computer')
    - At the command prompt, enter the following command (with your own JIRA Home path) before running JIRA from the command prompt:
      - `set JIRA_HOME=X:\path\to\JIRA\Home`
        Please set your `JIRA_HOME` environment variable value using this format, where:
        - X is the drive letter where your JIRA Home Directory is located and
        - no spacing has been added around the equal sign ('=')
    - Specify the command above in a batch file used to start JIRA.
  - On Linux/Solaris, do one of the following:
    - Enter the following command at a shell/console prompt (with your own JIRA Home path) before running JIRA:
      - `export JIRA_HOME=/path/to/jira/home`
    - Specify the command above in a script used to start JIRA.

You can specify any location on a disk for your JIRA home directory. Please be sure to specify an absolute path.

Please note that you cannot use the same JIRA home directory for multiple instances of JIRA. We recommend locating your JIRA Home Directory completely independently of the JIRA Installation Directory (i.e. not nesting one within the other) as this will minimise information being lost during major operations (e.g. backing up and restoring instances).

For more information about setting up your JIRA Home Directory, please see Setting your JIRA Home Directory.

3. Build JIRA

Now build JIRA by running `build.bat` (Windows) or `./build.sh` (Linux/Solaris) on the command line in the JIRA Installation Directory. This will produce the deployable WAR file in the `dist-tomcat` subdirectory of the JIRA Installation Directory.

4. Update Your Tomcat Installation's Libraries for JIRA

4.1 JDBC Drivers

Your Tomcat installation requires an appropriate JDBC driver to allow JIRA to communicate with the database. To add this JDBC driver to Tomcat, refer to the appropriate instructions:

- Copy the PostgreSQL JDBC Driver to Tomcat
- Copy the MySQL JDBC Driver to Tomcat
- Copy the Oracle JDBC Driver to Tomcat
- Copy the SQL Server 2005 JDBC Driver to Tomcat
- Copy the SQL Server 2008 JDBC Driver to Tomcat
- Copy the HSQL JDBC Driver to Tomcat

  If you intend to use the HSQL database, the HSQL JDBC driver is already included with the other JIRA library files that will be added to Tomcat in the following step.

4.2 Other JIRA Libraries for Tomcat

Tomcat does not come with some libraries required to run JIRA. To fix this, download http://www.atlassian.com/software/jira/downloads/binary/jira-jars-tomcat-distribution-5.0-rc2-tomcat-5x.zip extract and copy the `.jar` library files from this archive to the `common/lib` subdirectory of your Tomcat installation directory.

Please Note:

- Be sure to remove existing versions of these `.jar` library files before copying over new ones.
- To prevent exceptions related to logging, please ensure that the following files are present in Tomcat's `lib` directory. Also ensure that these files *are not present* in the `webapp/jira/WEB-INF/lib` subdirectory of the JIRA Installation Directory. If any of the following files are present in the `webapp/jira/WEB-INF/lib` subdirectory, remove them and rebuild the deployable JIRA WAR file as described in the previous step (above).

File
`jcl-over-slf4j-x.y.z.jar`
`jul-to-slf4j-x.y.z.jar`
`log4j-x.y.z.jar`
`slf4j-api-x.y.z.jar`
`slf4j-log4j12-x-y-z.jar`

5. Configure JIRA's Context in Tomcat

A JIRA 'context' now needs to be set up in Tomcat. To do this:

1. Copy the `jira.xml` file from the `dist-tomcat/tomcat-5.5` subdirectory of your JIRA Installation Directory to the `conf/Catalina/localhost` subdirectory of your Tomcat installation directory.
2. If necessary, customise the copied `jira.xml` file within your Tomcat installation — in particular, the value of the `docBase` attribute of the `Context`element:

```
<Context path="/jira" docBase="path/to/atlassian-jira-4.x.war" debug="0"
useHttpOnly="true">

  <Resource name="UserTransaction" auth="Container"
type="javax.transaction.UserTransaction"
    factory="org.objectweb.jotm.UserTransactionFactory" jotm.timeout="60"/>
  <Manager pathname=""/>

</Context>
```

The paths (denoted as `path/to/`) will be correct by default, assuming you want to deploy the `.war` from the `dist-tomcat` subdirectory of your JIRA Installation Directory.

If you are installing in Windows, make sure that the paths you specify for the location of the WAR file and database are full paths with drive letters (e.g. `C:\path\to\atlassian-jira-4.x.war`).

6. Modify Tomcat's `server.xml` to Handle Internationalised Characters Correctly

In order for JIRA to correctly display internationalised characters in user and group names, you need to modify the `conf/server.xml` file in your Tomcat installation directory by specifying the `URIEncoding="UTF-8"` property within the connector definition for your HTTP protocol.

The connector definition is specified by the following element in your `server.xml` file:

```
<Connector port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"/>
```

You should modify this element by specifying the `URIEncoding="UTF-8"` attribute:

```
<Connector port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true" URIEncoding="UTF-8"/>
```

Please Note:

- Since this property must be specified at the connector level for your application server, this setting will effect all other web applications deployed to the same application server installation running JIRA. While this setting should not adversely effect these other web applications, you should be aware of this point.
- JIRA will run fine without this property set. However, you will run into issues if a user or group is created which contains international characters. Hence, it is recommended that you set this property.

7. Fix Memory and Mail Handling Settings in Tomcat

Memory and mail handling settings need to be modified in Tomcat to avoid the following issues:

- Tomcat effectively leaks memory by caching JSPs. This can result in `OutOfMemoryError` errors if large pages (such as RSS or Excel pages) are requested.
- JIRA requires more memory than what Tomcat provides by default. This may lead to `OutOfMemory` errors when running JIRA if these memory settings are not increased.
- For JIRA's mail handler to avoid problems with RFC 2231-compliant mail clients, set the `mail.mime.decodeparameters` startup parameter in Tomcat to `true`.

To prevent these issues, follow the appropriate instructions for your operating system below.

For Windows

If Tomcat is not installed as a service:

- Edit Tomcat's `bin/setenv.bat`file (or create this file if it does not exist) and set:

```
set CATALINA_OPTS=%CATALINA_OPTS%
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true
-Dmail.mime.decodeparameters=true -Xms128m -Xmx512m -XX:MaxPermSize=256m
```

If you wish to run Tomcat as a service:

- Run the following command from Windows command prompt:

```
tomcat5 //US//JIRA
++JvmOptions="-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true
-Dmail.mime.decodeparameters=true -Xms128m -Xmx512m -XX:MaxPermSize=256m"
```

For Linux/Solaris

Edit Tomcat's `bin/setenv.sh` file (or create this file if it does not exist) and set:

```
export CATALINA_OPTS="$CATALINA_OPTS
-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true -Dmail.mime.decodeparameters=true
-Xms128m -Xmx512m -XX:MaxPermSize=256m"
```

For other environments and more information on memory settings, see Increasing JIRA Memory.

8. Start Tomcat

(Only required if Tomcat is not running as a service.)

JIRA should now be ready to run in Tomcat. To start up JIRA, start (or restart) the Tomcat server with Tomcat's `bin/startup.sh` or `bin/startup.bat` scripts.

9. Run the Setup Wizard

Point your browser to

http://localhost:8080/jira

You should now see the Setup Wizard, which will take you through the JIRA's setup procedure, including a configuration step for your database connection.

Troubleshooting

It is easy to make a mistake in this process. First, check that you have followed the process described above:

- Have you have made changes to `edit-webapp/WEB-INF/classes/entityengine.xml` in your JIRA Installation Directory (step 2 above) and re-run the build script (step 3 above), but your `entityengine.xml` changes were not picked up? If so, delete the `webapps/jira` subdirectory of your Tomcat installation directory and then restart JIRA. (In some circumstances, Tomcat does not correctly re-expand the web application.)
- If you are using an external database, did you copy the correct JDBC driver jar file to the `lib` subdirectory of your Tomcat installation directory? (Refer to step 4 above.)
- Have you updated your Tomcat installation's libraries for JIRA by copying across the additional jar files downloaded in step 4 above? Check if `objectweb-datasource-x.y.z.jar` present in the `lib` subdirectory of Tomcat's installation directory.
- Is the path to your built `.war` file within `conf/Catalina/localhost/jira.xml` of your Tomcat installation directory correct? (Refer to step 5 above.)
- Have you copied your built `.war` file to Tomcat's `webapps` directory? This is almost guaranteed to cause problems - please move this `.war` file elsewhere and delete any JIRA subdirectories created in Tomcat's `webapps` directory which Tomcat may have created (after Tomcat is initially started).
- Have you configured JIRA's context and other custom settings centrally in Tomcat's `conf/server.xml` file instead of the `conf/Catalina/localhost/jira.xml` file of your Tomcat installation directory? Although this is fine, be sure that you *do not* also have a `conf/Catalina/localhost/jira.xml` file present.
- The log files are usually vital to debugging problems. On Windows, these will appear in the console window that loads when running `startup.bat`, or in one of the log files in Tomcat's `logs` directory. On Linux/Solaris, logs will appear in a log file in `logs`, usually `logs/*` (not just `logs/catalina.out`). Check the log file for errors after startup.
- If you experience high memory usage / memory leaks (e.g. `OutOfMemoryError`), you may wish to set the system property `-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true` in the `setenv.sh` / `setenv.bat` file. For more information please see JRA-10145. (Refer to step 7 above.)
- If the connection to your database is dropping out (in particular with MySQL), you will need to set up Tomcat to survive connection closures.
- ⚠ **Please be aware:** The `build.xml` file is an Ant file, which when invoked with the `build.sh` / `build.bat` script, will construct deployable web application archive (`.war`) files (for supported application servers). The `build.xml` file copies the contents of the `webapp` subdirectory of your JIRA Installation Directory and overwrites it with the contents of the sibling `edit-webapp` directory, when constructing its `.war` files. Thus, unless otherwise requested, never edit files within this `webapp` directory!
  If a file needs editing, first copy it from `webapp/path/to/file` to `edit-webapp/path/to/file` subdirectories of your JIRA Installation Directory and edit it in the latter location.

If you're stuck, please raise a support request, and attach your logs, configuration files, plus anything else relevant, and we'll get back to you as soon as possible.

User-contributed notes

Have experiences to share with Tomcat 5.5.x and JIRA? We welcome your thoughts. Please see the user-contributed Tomcat 5.5.x notes.

Tomcat 5.5 notes

This page has general notes on installing JIRA on Tomcat 5.5.x. It supplements the official Tomcat installation docs.

A user writes:

There is a readme file shipped with JIRA. It said that you need to change port 8080 in order to get Tomcat(say a) to start up. **BUT**, if you

have a **CATALINA_HOME** environment variable already set for an existing Tomcat(say b) running on the same server, running startup from the bin directory will start up Tomcat(b). For Tomcat(a) to start one needs to delete the existing **CATALINA_HOME** environment variable as well.

Switching Application Servers to Apache Tomcat

To move JIRA from a non-Tomcat application server to Apache Tomcat, use one of the following methods:

> **Note**
>
> Regardless of which method you use, back up your data first and make sure you test JIRA on the new server before deploying it in production.

Method 1. Export and import the database

Follow the Migrating JIRA to Another Server instructions, installing the new version of JIRA on your new application server.

Method 2. Use your existing database

If you are using the same version of JIRA on the old and new (Apache Tomcat) application server, you do not have to export and re-import your JIRA database (as described in the instructions for Migrating JIRA to Another Server). You can use your existing database with the new application server.

However, you cannot simply copy the WAR file or expanded WAR directory from an existing JIRA WAR installation in the old application server to the new application server. **This will not work.**

To switch to a new (Apache Tomcat) application server, follow these instructions:

1. Install JIRA on the new application server. (Refer to the instructions for your version of Apache Tomcat in the Installing JIRA WAR section.)
2. Check that the JNDI location of the `UserTransaction` as declared in the `entityengine.xml` file is correct for Apache Tomcat.
3. From System Info, check the modified files to see what customisations, if any, exist from the original installation. Consider these changes in your new server.
4. Make sure you shut down the old server before you start up the new one.
5. If you are running the new application server on a different machine to the old one, carry out the following actions as soon as you start the new server:
    - Re-index your data.
    - Make sure that the attachment path is valid for the new server.

Deploying Multiple Atlassian Applications in a Single Tomcat Container

Deploying multiple Atlassian applications in a single Tomcat container is **not supported**. We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration:

- You may not be able to start up all of the applications in the container, due to class conflicts (in 3rd party libraries bundled with our application) that result from the Atlassian applications sharing a single JVM in the Tomcat container.
- You will not be able to determine the startup order of the applications. Hence, you may experience problems such as JIRA starting before Crowd, rather than vice versa.
- Memory problems are also common as one application may allocate all of the memory in the Tomcat JVM to itself, starving the other applications.

We also do not support deploying multiple Atlassian applications to a single Tomcat container for a number of practical reasons. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

Finally, we recommend not deploying *any other applications* to the same Tomcat container that runs the Atlassian application, especially if these other applications have large memory requirements or require additional libraries in Tomcat's `lib` subdirectory.

Configuring Your JIRA Installation

The pages listed below contain information on how to configure and fine-tune your JIRA installation:

- Using the JIRA Configuration Tool
- Running JIRA as a Service
- Starting JIRA Automatically on Linux
- Starting JIRA automatically on FreeBSD

Using the JIRA Configuration Tool

About the JIRA Configuration Tool

The **JIRA Configuration Tool** is an application (included with all JIRA distributions except JIRA WAR) that offers server-level JIRA

configuration through a convenient GUI. This tool allows you to conveniently configure:

- Your JIRA Home Directory
- Your Database Connection
- The TCP ports that JIRA runs through.

Whenever you configure or reconfigure JIRA's server-level settings using this tool, JIRA must be restarted so it can recognise these changes

Please Note:

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. If you need to install a Java platform to run this tool, we recommend using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to perform these server-level configurations manually.

On this page:

- About the JIRA Configuration Tool
- Starting the JIRA Configuration Tool
- Configuring the JIRA Home Directory
- Configuring the Database Connection
- Configuring JIRA's TCP Ports
- Saving your settings

Starting the JIRA Configuration Tool

To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

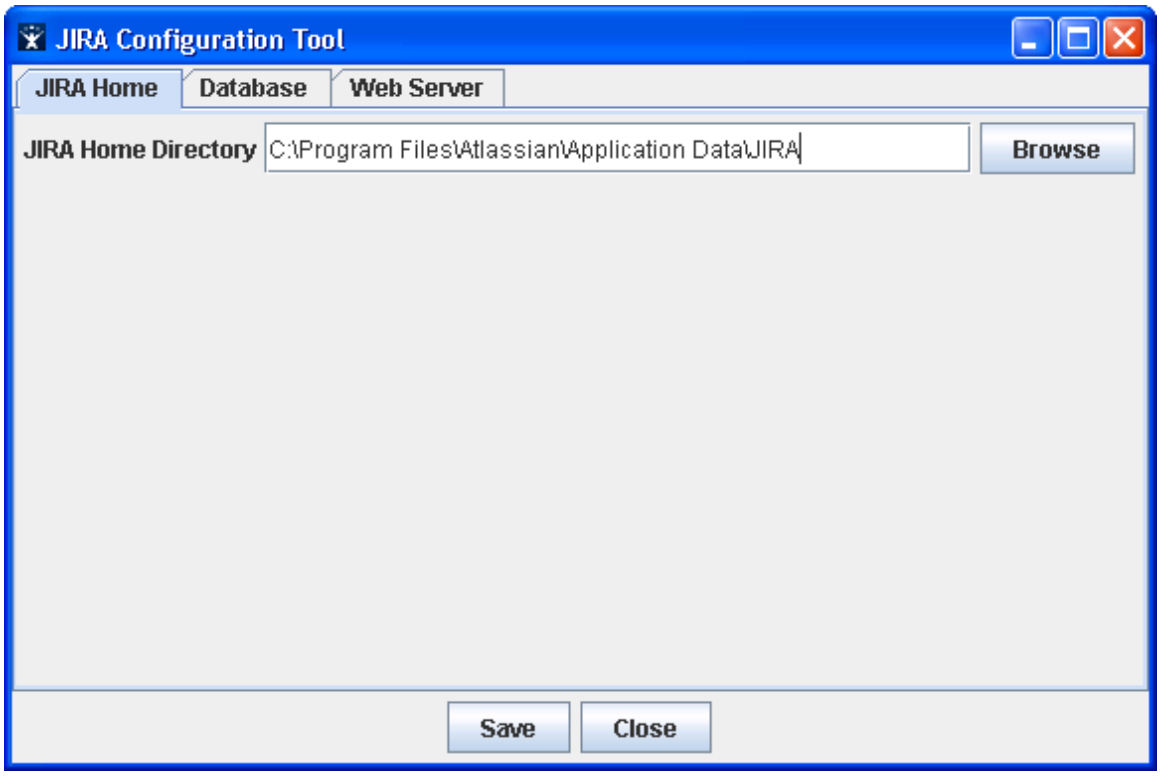
Please Note: You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

Configuring the JIRA Home Directory

Your JIRA Home Directory allows you to set the folder that JIRA uses to store its various data files.

To set your JIRA Home Directory using the JIRA Configuration Tool:

1. Click the '**JIRA Home**' tab.
2. In the 'JIRA Home Directory' field, type the full file path into the text field, or click the '**Browse**' button to browse for the location of your JIRA Home Directory.



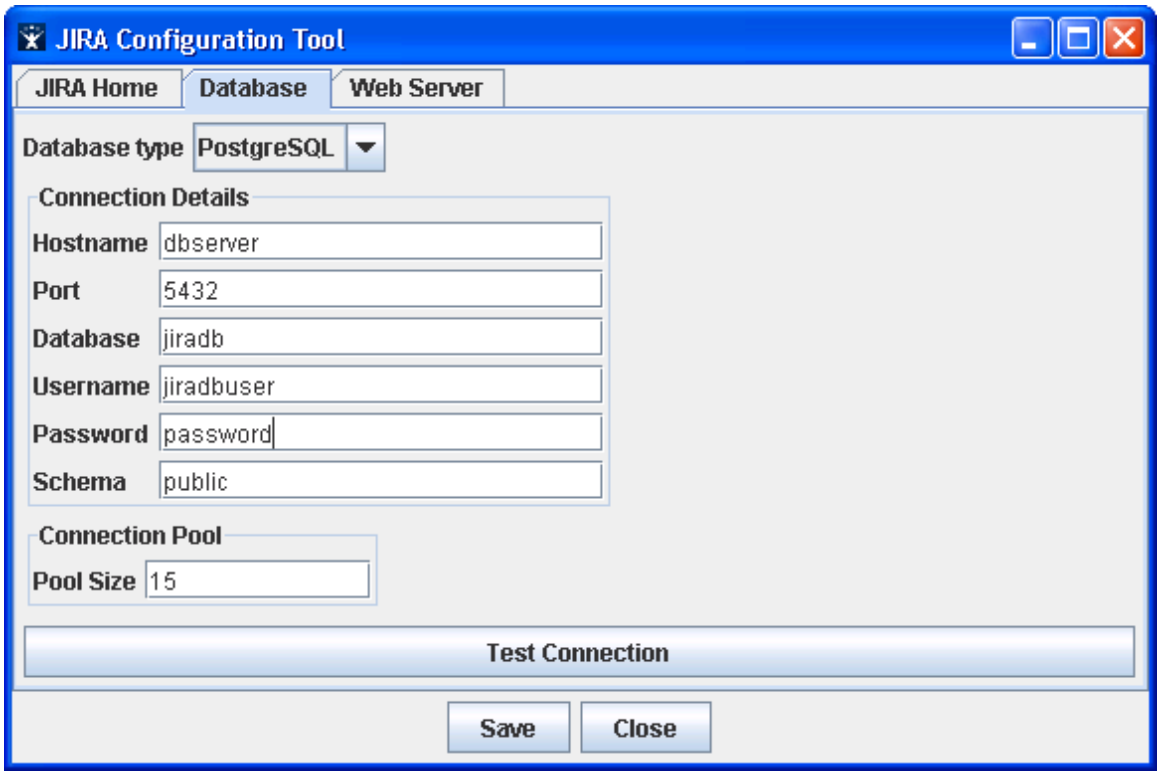
Upon clicking the '**Save**' button, your changes are saved to the `jira-application.properties` file located in the `<jira-application-dir>` subdirectory of your JIRA Installation Directory.

For more information, please see Setting your JIRA Home Directory.

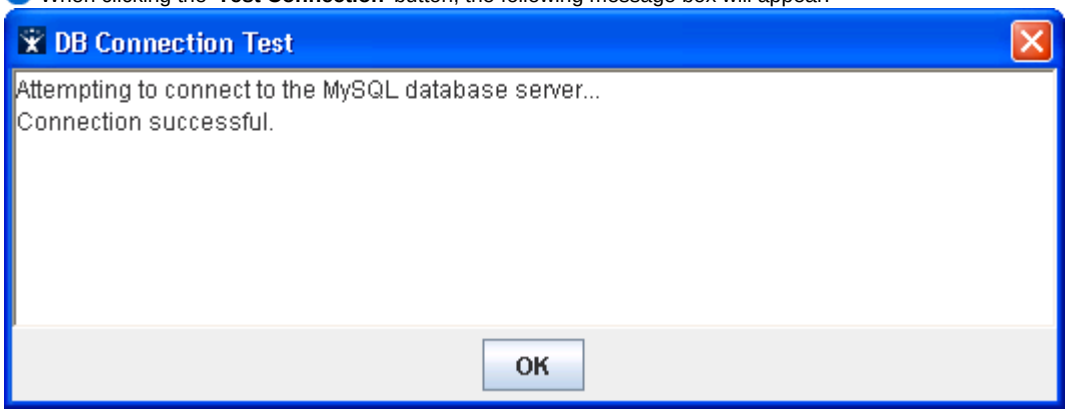
Configuring the Database Connection

To configure JIRA's database connection using the JIRA Configuration Tool, follow the appropriate procedure for your database type:

- Connecting JIRA to PostgreSQL
- Connecting JIRA to MySQL
- Connecting JIRA to Oracle
- Connecting JIRA to SQL Server 2005
- Connecting JIRA to SQL Server 2008
- Connecting JIRA to HSQLDB



When clicking the '**Test Connection**' button, the following message box will appear:



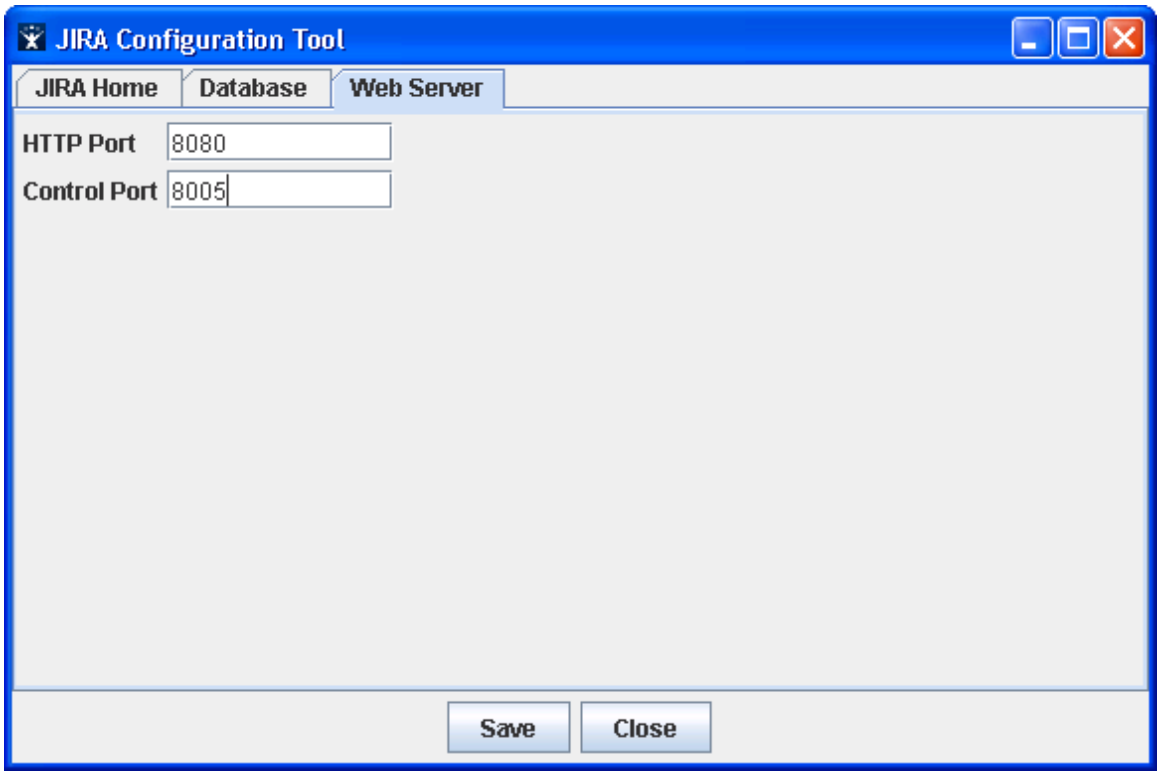
('Connection Successful' will appear if JIRA connected to your database successfully.)

Configuring JIRA's TCP Ports

JIRA runs on TCP listening (HTTP) port 8080 and shutdown (Control) port 8005 by default. However, you can change these ports if they are already being used by an existing service running on the same machine.

To change JIRA's TCP ports using the JIRA Configuration Tool:

1. Click the '**Web Server**' tab.
2. In the 'HTTP Port' field, enter the new TCP listening port number.
3. In the 'Control Port' field, enter the new TCP shutdown port number.



Upon clicking the '**Save**' button, your changes are saved to the `server.xml` file located in the `conf` subdirectory of your JIRA Installation Directory.

For more information, please see Changing JIRA's TCP Ports.

Saving your settings

Click the '**Save**' button when you are done. This saves all configuration changes made in all tabs of the JIRA Configuration Tool.

JIRA must be restarted for your new settings to take effect.

Running JIRA as a Service

For long-term use, JIRA should be configured to automatically restart when the operating system restarts. For Windows servers, this means configuring JIRA to run as a **Windows service.**

If you are running JIRA on Linux and want to start it automatically, please refer to Starting JIRA Automatically on Linux instead.

Running JIRA as a Windows service has other advantages. When started manually a console window opens and there is a risk of someone accidentally shutting down JIRA by closing this window. Also, the JIRA logs are properly managed by the Windows service (found in `logs\stdout*.log` in your JIRA Home Directory, and rotated daily).

There are two ways to install JIRA as a service: via the installer, and manually.

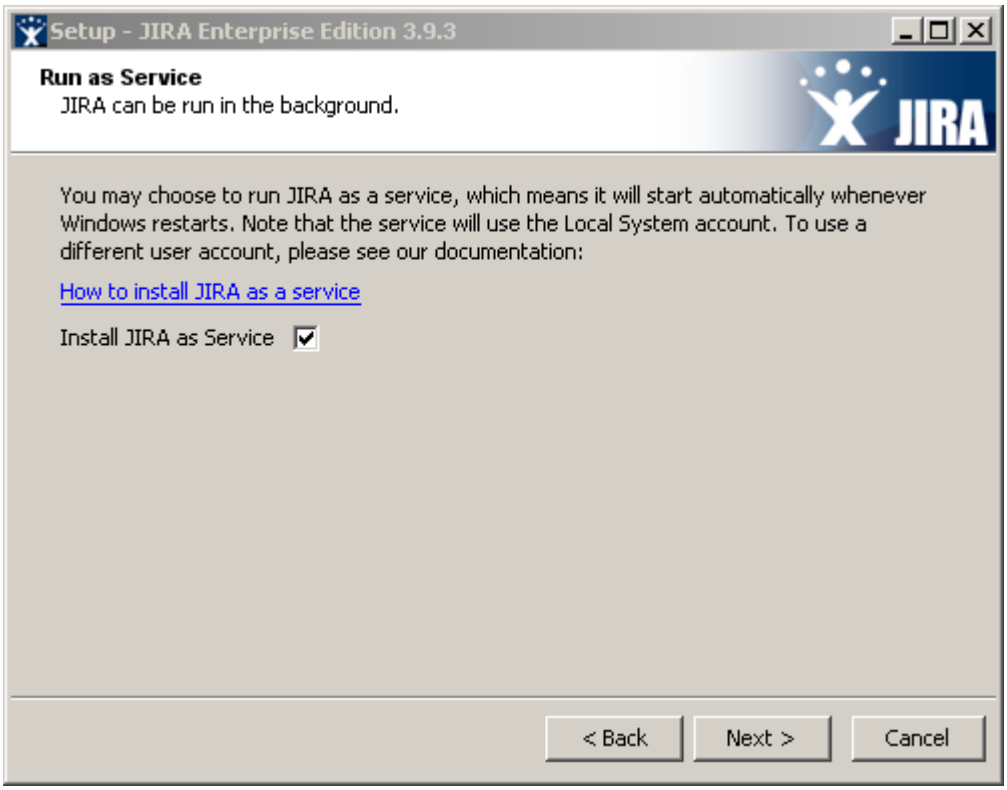
On this page:

- Installing JIRA as a Service
  - Installing as a Service with the Installer
  - Manually Setting up JIRA to Run as a Service
- Removing the JIRA service
- Changing the Windows user that the JIRA service uses
- Specifying the startup order of multiple services
- Locating the name of a service
- Troubleshooting

Installing JIRA as a Service

Installing as a Service with the Installer

The easiest way to get JIRA installed as a Windows service is by clicking the '**Install JIRA as Service**' check box when running the Windows Installer:



You will need full Administrator rights on your Windows operating system for this installation process to complete successfully.

Manually Setting up JIRA to Run as a Service

You can still set up JIRA to run as a service, if any of the following situations apply to you:

- You did not use the Windows Installer.
- You used the Windows Installer, but did not initially install JIRA as a service.

Please Note:

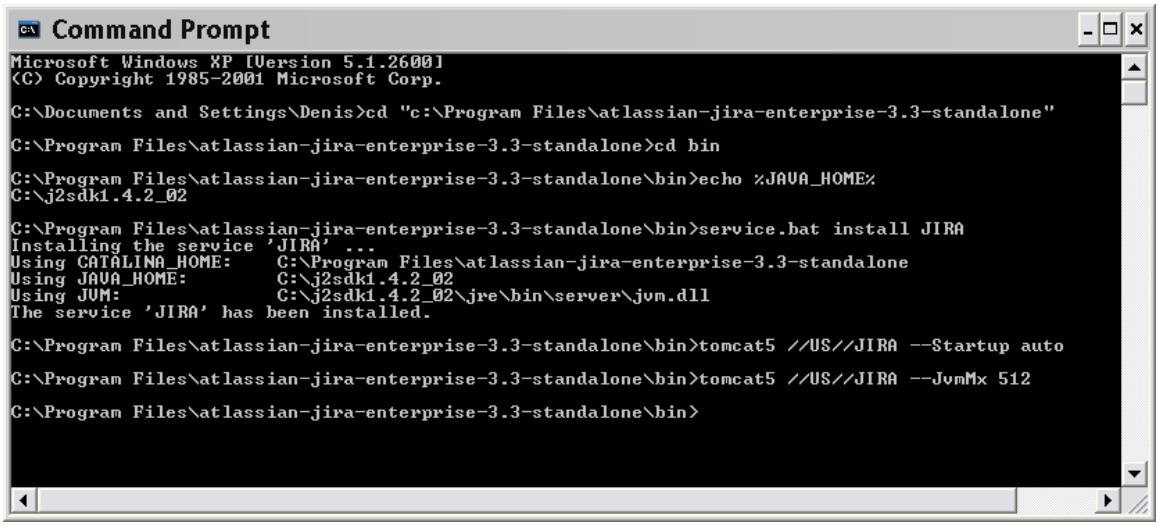
- These instructions do not apply to installations of the JIRA WAR distribution. To run a JIRA WAR installation as a service, refer to the relevant JIRA WAR installation instructions for Apache Tomcat 6.0 or 5.5.
- If you are running a 64-bit version of Windows, please note that Apache Tomcat cannot run as a Windows service if you are using a 64-bit JDK/JRE (see JRA-12965). Please ensure that you are using a 32-bit JDK/JRE.
- If you used the Windows Installer, a 32-bit JDK will already have been installed for you.
- On any Windows operating system with User Account Control (UAC) such as Windows Vista or Windows 7, you must either disable UAC or run 'cmd.exe' as an administrator (e.g. by right-clicking on 'cmd.exe' and selecting "Run as administrator") in order to execute the script in the procedure below. If UAC is enabled, simply logging in to Windows with an Administrator account will not be sufficient.

To set up JIRA to run as a service:

1. Open a Command Prompt.
2. Change directory ('cd') to the JIRA installation directory and then into this directory's '`bin`' subdirectory.
   If a directory in the path has spaces (e.g. '`C:\Program Files\..`'), please convert it to its eight-character equivalent (e.g. '`C:\Progra~1\..`').
3. Ensure the **JAVA_HOME** variable is set to the root of your Java platform's installation directory.
   To find out the current value of the **JAVA_HOME** variable, enter `echo %JAVA_HOME%` at the command prompt.
4. Run the following command:

   ```
   service.bat install JIRA
   ```

   Here is a screenshot of the process:



JIRA should now be set up to run as a service.

5. In addition, to have the JIRA service start automatically when the operating system starts, run:

```
tomcat6 //US//JIRA --Startup auto
```

The JIRA service will automatically start up the next time the operating system reboots. The JIRA service can be manually started with the command **'net start JIRA'** and stopped with **'net stop JIRA'.**

To see what parameters the JIRA service is starting with, go to **Start** -> **Run** and run '`regedt32.exe`' and then:
* For Windows 32 bit edition navigate to HKEY_LOCAL_MACHINE -> SOFTWARE -> Apache Software Foundation -> Procrun 2.0 -> JIRA<time stamp>
* For Windows 64 bit edition navigate to HKEY_LOCAL_MACHINE -> SOFTWARE -> Wow6432Node -> Apache Software Foundation -> Procrun 2.0 -> JIRA<time stamp>

6. Additional JIRA setup options (optional):
   - To increase the maximum memory JIRA can use (the default will already be 256MB), run:

     ```
     tomcat6 //US//service_name --JvmMx 512
     ```

     where **service_name** is the name of your JIRA service, e.g. JIRA123487934298.

   - To add a JVM parameter, for example pass a parameter to enable JIRA's Jelly support, run:

     ```
     tomcat6 //US//service_name ++JvmOptions="-Djira.jelly.on=true"
     ```

     where **service_name** is the name of your JIRA service, e.g. JIRA123487934298.

   - If you are running JIRA and Confluence in the same JVM, increase the MaxPermSize size to 128 MB:

     ```
     tomcat6 //US//service_name ++JvmOptions="-XX:MaxPermSize=128m"
     ```

     where **service_name** is the name of your JIRA service, e.g. JIRA123487934298.

   - Occasionally, it may be useful to view JIRA's Garbage Collection information. This is especially true when investigating memory issues.To turn on the Verbose GC (garbage collection) logging, execute the following command in the command prompt:

     ```
     tomcat6 //US//service_name ++JvmOptions="-Xloggc:path\to\logs\atlassian-gc.log"
     ```

     where **service_name** is the name of your JIRA service, e.g. JIRA123487934298.

The path (denoted by **\path\to**) refers to the directory in which JIRA is currently installed. For example:

```
tomcat6 //US//service_name ++JvmOptions="-Xloggc:c:\jira\logs\atlassian-gc.log"
```

where **service_name** is the name of your JIRA service, e.g. JIRA123487934298.

See the Tomcat documentation for further service options.

Removing the JIRA service

If JIRA was installed through the Windows installer, go to the '**Control Panel**' in Windows, click '**Add or Remove Programs**' and remove JIRA. This will remove the service too.

If you installed the service manually (see above) it can be uninstalled with:

```
service.bat remove JIRA
```

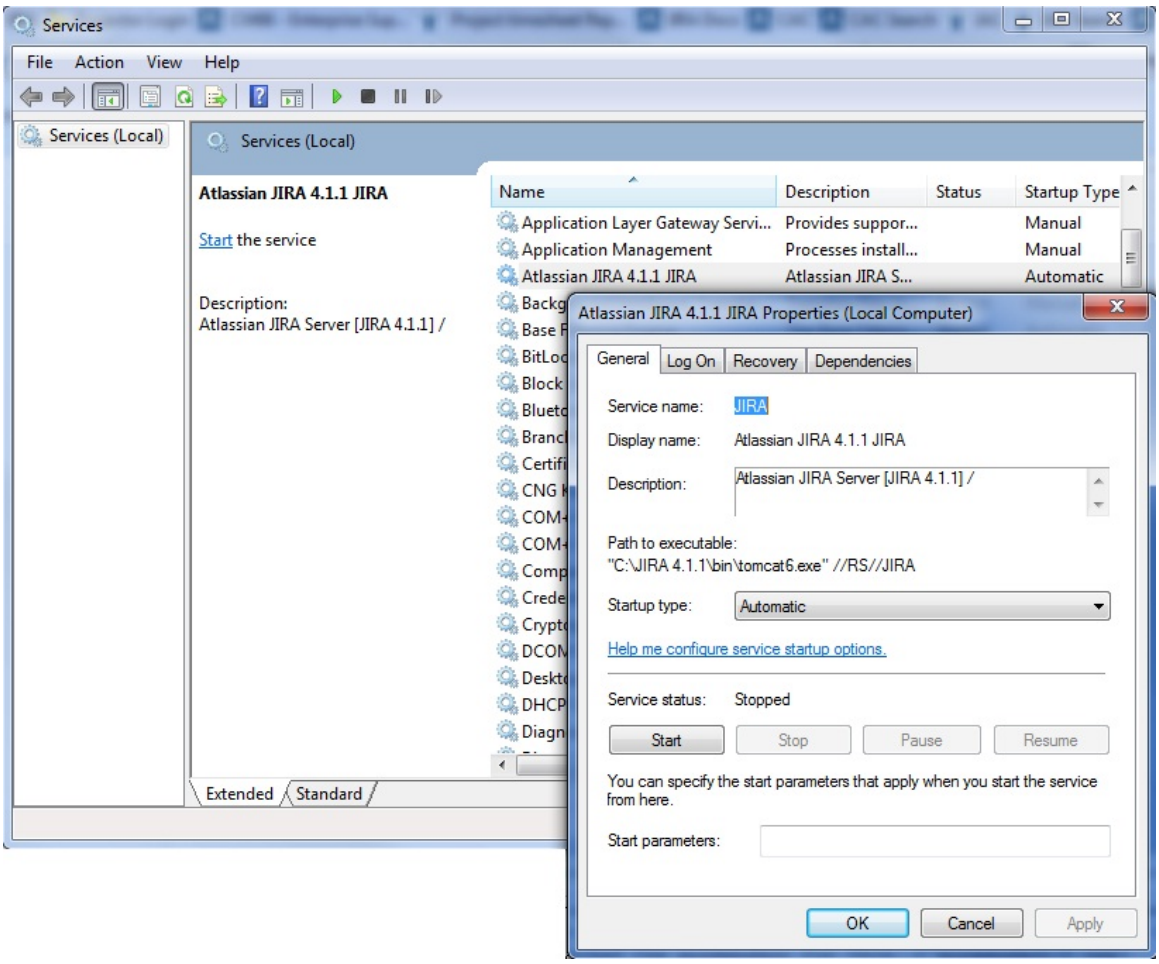
Alternatively, if the above does not work, use `tomcat6 //DS//JIRA`.

Changing the Windows user that the JIRA service uses

If you are using mapped network drives for JIRA's backup directory, attachments directory, index directory or the %CATALINA_HOME%* directory, you need to ensure that JIRA can write to these drives. That is, these directories all need to be writeable by the user which the JIRA service is running as. This may mean that you need to change the Windows user that the JIRA server uses.

Note that you must also specify these network drives by UNC and not letter mappings, e.g. **\\backupserver\jira** not **z:\jira**

To change the Windows user that the JIRA service uses, navigate to the service in Windows, i.e. **'Control Panel' -> 'Administrative Tools' -> 'Services'**. Locate the 'Atlassian JIRA' service, right-click and view the 'Preferences'.



Go to the '**Log On**' tab and change the user as desired.

Specifying the startup order of multiple services

If you have services that depend on each other, it is important that they are started in the correct order. Common examples include:

- If you are running both JIRA and Crowd, it is important to start Crowd first, so that Crowd is running before people try to login to JIRA.
- If the database JIRA connects to is hosted on the same server as JIRA, and is started via a Windows service, the JIRA service will only start successfully if the database service has already started first.

To set up start up dependency rules, open a command prompt and enter the following command:
`C:\Documents and Settings\Developer>sc config [JIRA service] depend=[database service]`
Please note the space character after `'depend='`.

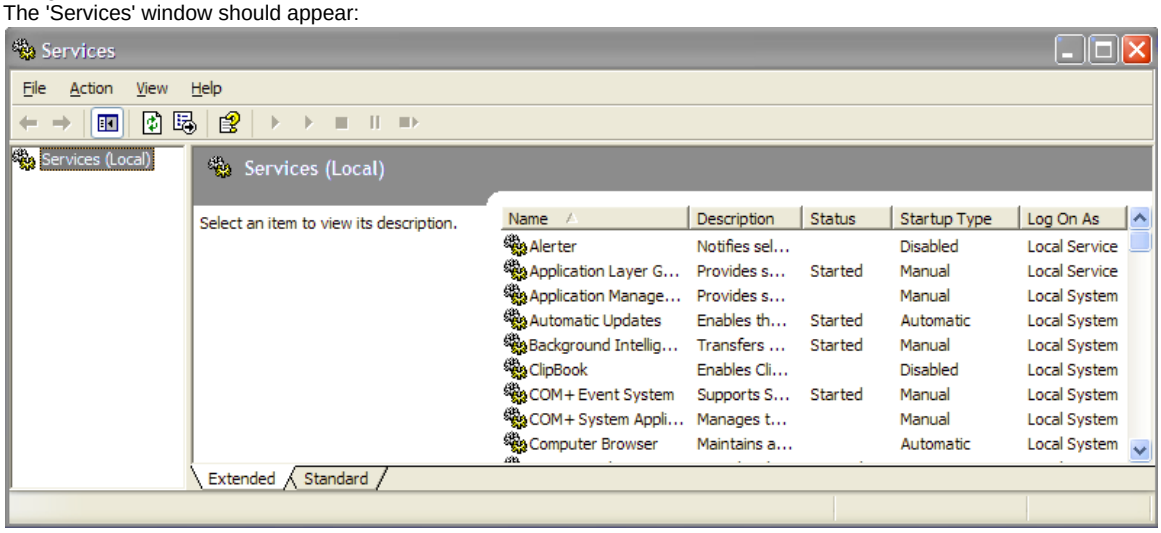
- **[JIRA service]** is the name of the JIRA service you are running, e.g. JIRA051007111904.
- **[database service]** is the name of the database service you are running, e.g. MSSQLSERVER.

If you wish, you can also set up dependency rules by editing the system registry. Please see http://support.microsoft.com/kb/193888 for details on how to do this.

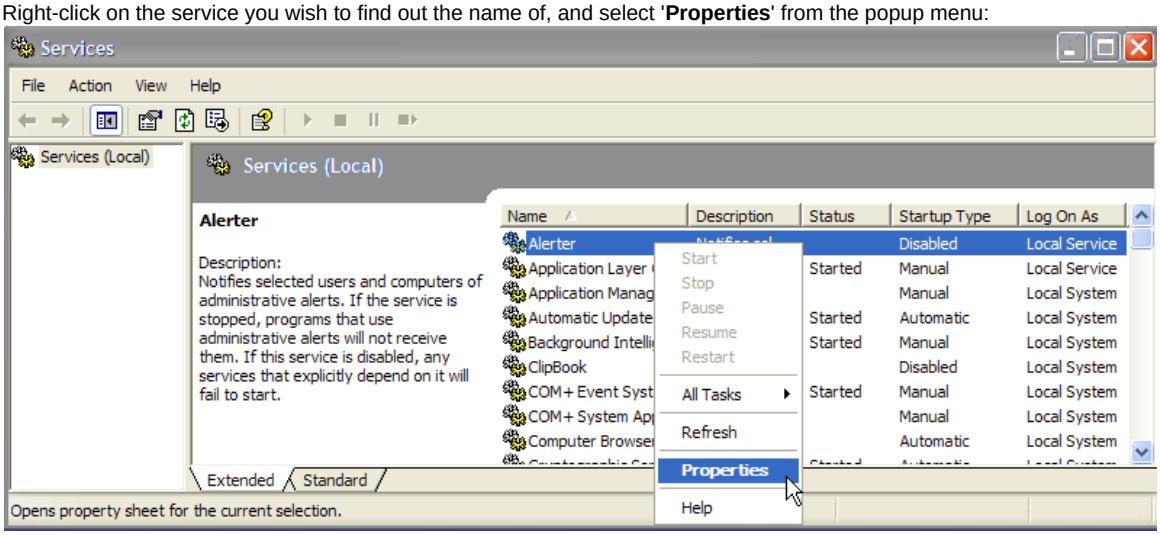
Locating the name of a service

If you do not know the exact name of your JIRA service or your database service, you can find out what they are by following the steps below:

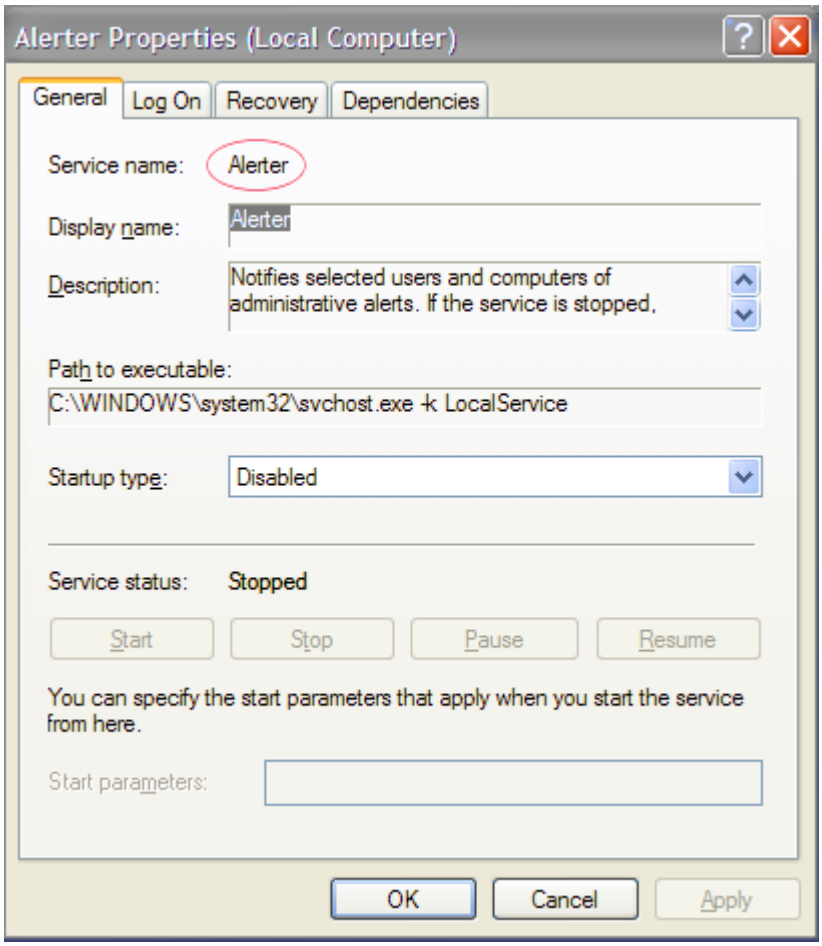
1. Navigate to **'Control Panel' > 'Administrative Tools' > 'Services'**.
2. The 'Services' window should appear:



3. Right-click on the service you wish to find out the name of, and select '**Properties**' from the popup menu:



4. The 'Service name' should appear in the 'General' tab:



Troubleshooting

- Problems may occur when trying to setup JIRA to run as a Windows service with JDK 1.6. The problem is due to failure to locate "MSVCR71.DLL", which can be found in `%JAVA_HOME%/bin`. There are two options to resolve this problem:
  - Add %JAVA_HOME/bin to PATH, then restart the JIRA server.
  - Copy MSVCR71.DLL to system path, C:\WINDOWS\SYSTEM32 or C:\WINNT\SYSTEM32
- Take note of the username that the service is running as, and be sure to modify the `/temp` and `/work` directories in your install directory so that this user has read and write permissions.
- You cannot run JIRA as a service on a 64-bit operating system if you require allocating more than 1.5GB of memory, due to 32-bit JDK memory limitations and 64-bit JDK/Tomcat service issues.

Starting JIRA Automatically on Linux

Linux/Solaris system administration is outside the scope of Atlassian support. This page is provided for your information only.

On Linux/Solaris, the best practice is to install, configure and run each service (including JIRA) as a dedicated user with only the permissions they require.

To install, configure and get JIRA to start automatically on Linux/Solaris:

1. Create a `jira` user account which will be used to run JIRA. For example, enter the following at a Linux/Solaris console:

```
sudo useradd --create-home -c "JIRA role account" jira
```

2. Create a directory into which JIRA will be installed. For example:

```
sudo mkdir /opt/atlassian/jira
sudo chown jira: /opt/atlassian/jira
```

3. Log in as the `jira` user to install JIRA:

```
sudo su - jira
```

4. Assuming you downloaded the JIRA from a 'tar.gz' archive, you need to extract it:

```
cd /opt/atlassian/jira
tar zxvf /tmp/atlassian-jira-X.Y.tar.gz
ln -s atlassian-jira-X.Y/ current
```

5. Edit `current/atlassian-jira/WEB-INF/classes/jira-application.properties` and set `jira.home=/var/atlassian/application-data/jira`
6. Then back as root, create the file `/etc/init.d/jira` (code shown below), which will be responsible for starting up JIRA after a reboot (or when manually invoked).

```
#!/bin/sh -e
# JIRA startup script
#chkconfig: 2345 80 05
#description: JIRA

# Define some variables
# Name of app ( JIRA, Confluence, etc )
APP=jira
# Name of the user to run as
USER=jira
# Location of application's bin directory
BASE=/opt/atlassian/jira/current
# Location of Java JDK
export JAVA_HOME=/usr/lib/jvm/java-6-sun

case "$1" in
  # Start command
  start)
    echo "Starting $APP"
    /bin/su -m $USER -c "cd $BASE/logs && $BASE/bin/startup.sh &> /dev/null"
    ;;
  # Stop command
  stop)
    echo "Stopping $APP"
    /bin/su -m $USER -c "$BASE/bin/shutdown.sh &> /dev/null"
    echo "$APP stopped successfully"
    ;;
   # Restart command
   restart)
        $0 stop
        sleep 5
        $0 start
        ;;
  *)
    echo "Usage: /etc/init.d/$APP {start|restart|stop}"
    exit 1
    ;;
esac

exit 0
```

7. Make the init script executable:

```
chmod \+x /etc/init.d/jira
```

8. Place symlinks in the run-level directories to start and stop this script automatically.

    a. For Debian-based systems:

```
update-rc.d jira defaults
```

The following commands will be executed to place symlinks in the run-level directories:

```
Adding system startup for /etc/init.d/jira ...
   /etc/rc0.d/K20jira -> ../init.d/jira
   /etc/rc1.d/K20jira -> ../init.d/jira
   /etc/rc6.d/K20jira -> ../init.d/jira
   /etc/rc2.d/S20jira -> ../init.d/jira
   /etc/rc3.d/S20jira -> ../init.d/jira
   /etc/rc4.d/S20jira -> ../init.d/jira
   /etc/rc5.d/S20jira -> ../init.d/jira
```

b. For RedHat-based systems:

the init.d script contains chkconfig settings

```
sudo /sbin/chkconfig --add jira
```

9. Ensure the script is executed in the correct order, in particular after the database startup script.

Thank you for this information

Thank you to Matthew Block and Pete Toscano for the original comments that we based this information on.

Starting JIRA automatically on FreeBSD

*nix-based operating system administration is outside the scope of Atlassian support. This document is provided for information-purposes only.

On *nix-based BSD operating systems, the best practice is to install, configure and run each service (including JIRA) as a dedicated user with only the permissions they require.

To run JIRA automatically on FreeBSD:

1. As root, create the file `/usr/local/etc/rc.d/jira.sh` (code shown below), which will be responsible for starting up JIRA after a reboot (or when manually invoked). If you are not using postgresql for your database, change the REQUIRE line to whatever is in the PROVIDE line in your database init script.

```
#!/bin/sh
#
# Startup script for JIRA on FreeBSD
#
# This goes in /usr/local/etc/rc.d and gets run at boot-time.

# PROVIDE: jira
# REQUIRE: postgresql
# KEYWORD: shutdown

#
# Add the following lines to /etc/rc.conf to enable jira:
#
#jira_enable="YES"
#
jira_enable="${jira_enable-NO}"

. /etc/rc.subr

name="jira"
rcvar=`set_rcvar`
start_cmd="${name}_start"
stop_cmd="${name}_stop"

jira_start()
{
    echo -n " Starting JIRA"
    su - atlassian -c '/home/atlassian/jira/bin/startup.sh'
}

jira_stop()
{
    echo -n " Stopping JIRA"
    su - atlassian -c '/home/atlassian/jira/bin/shutdown.sh'
}

load_rc_config $name
run_rc_command "$1"
```

2. Make the init script executable:

```
chmod \+x /usr/local/etc/rc.d/jira.sh
```

3. Make the init script readonly:

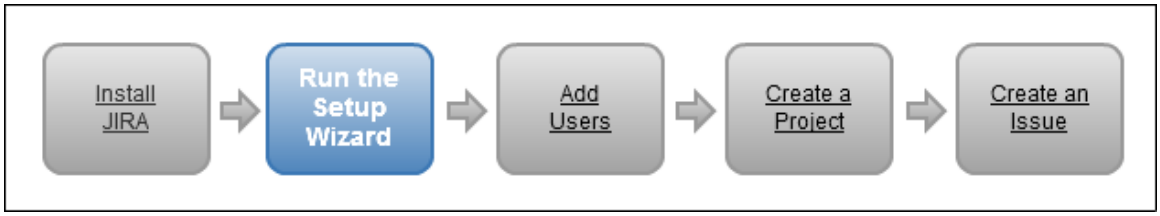
```
chmod \-w /usr/local/etc/rc.d/jira.sh
```

4. Add the following line to `/etc/rc.conf`

```
jira_enable="YES"
```

More information can be found in this article.

Running the Setup Wizard



After you have installed JIRA (or have deployed JIRA WAR to your application server) and have accessed its URL (eg. `http://<jira-server-name>:8080` or `http://<jira-server-name>:8080/jira`) for the first time, you will be presented with a brief Setup Wizard to configure JIRA.

On this page:

- Step 1 of 4: Database Configuration
  - Connecting to an External Database
- Step 2 of 4: Application Properties
  - License Key
- Step 3 of 4: Administrator Account
- Step 4 of 4: Email Notification
- You're done!

Step 1 of 4: Database Configuration

The first page of the Setup Wizard is shown below.

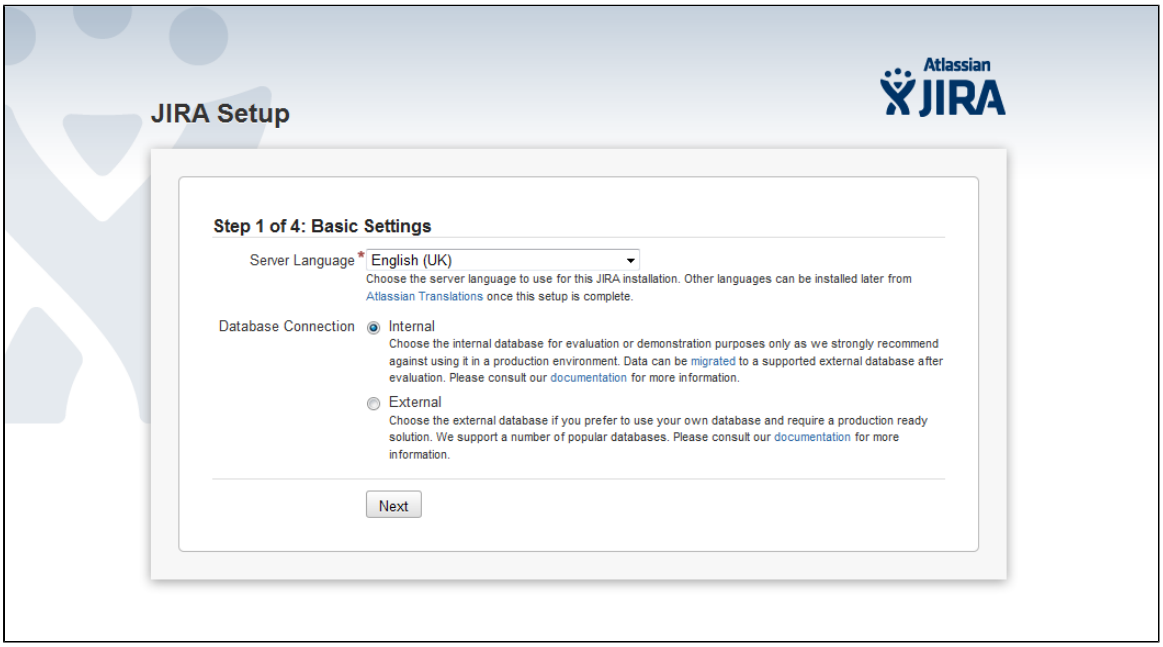
Choose the language you would like the JIRA user interface to appear in by selecting the preferred **Server Language**.

Please Note:

- As soon as you choose a language from the **Server Language** dropdown list, the JIRA user interface will switch to that language.
- Be aware that some languages may have more comprehensive translations than others.

On this page you can choose between connecting JIRA to:

- The **Internal** database — used for evaluating JIRA only or
- An **External** database — used for running JIRA in a production environment.

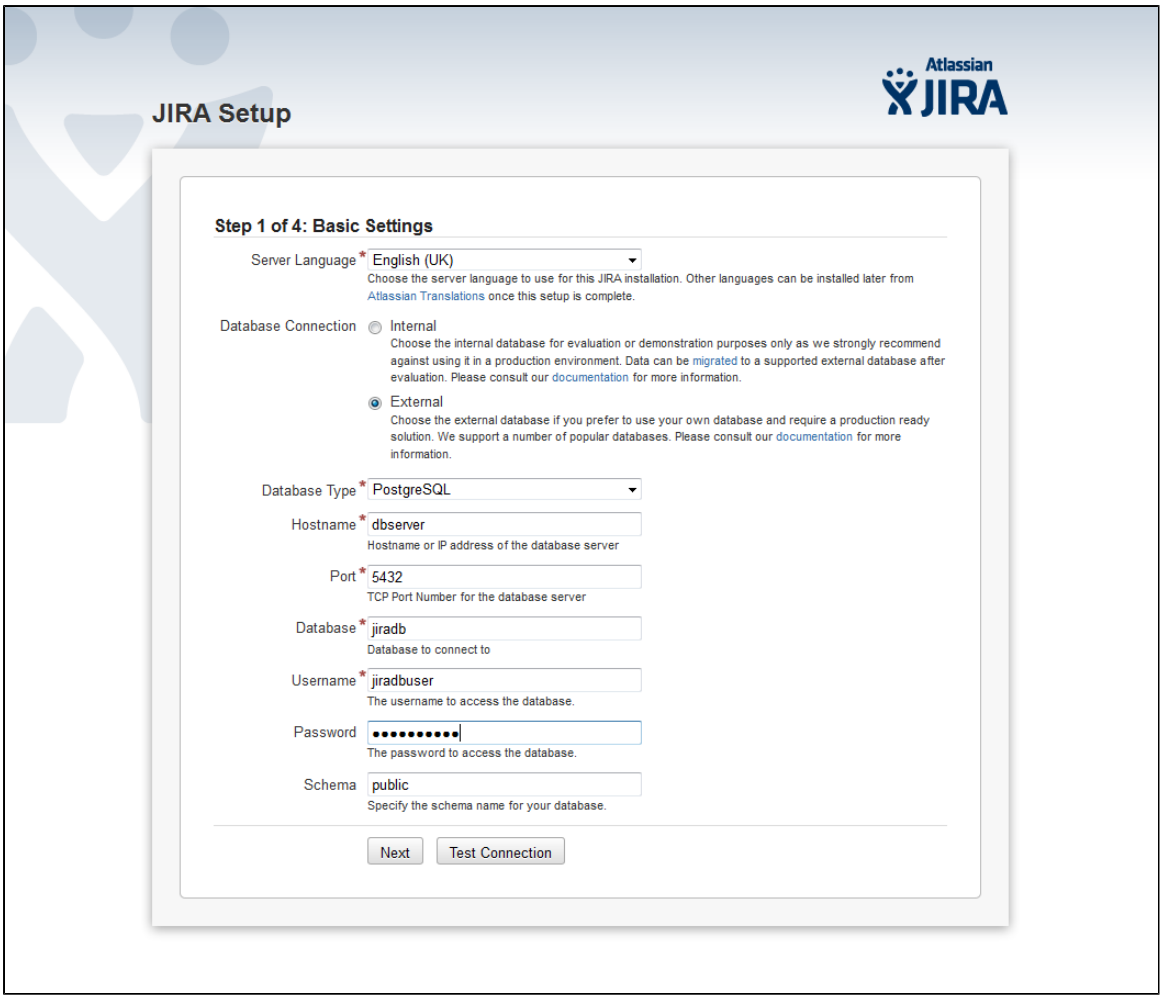


Connecting to an External Database

If you choose to connect JIRA to an external database (by clicking the **External** radio button), you will be presented with a number of database configuration options.

Your external database must be a newly-created (or empty) database. For more information on creating a database, refer to step 2 (for creating/configuring a database) of the appropriate documentation for your database:

- Create and Configure the PostgreSQL Database
- Create and Configure the MySQL Database
- Configure the Oracle Database
- Create and Configure the SQL Server 2005 Database
- Create and Configure the SQL Server 2008 Database



To configure your external database connection:

1. Choose the type of database server (running your new JIRA database) in the **Database Type** field. Your database configuration options will change depending on your chosen **Database Type**.
2. The fields on this page of the Setup Wizard are identical to those on the JIRA Configuration Tool (included with all JIRA distributions except JIRA WAR) when configuring your database connection. For details on configuring these fields for your external database, refer to step 3 of section 4.1 (Using the JIRA Configuration Tool) of the appropriate documentation:
    - Connecting JIRA with PostgreSQL
    - Connecting JIRA with MySQL
    - Connecting JIRA with Oracle
    - Connecting JIRA with SQL Server 2005
    - Connecting JIRA with SQL Server 2008

Please Note:

- In the Setup Wizard, JIRA expects that the external database it connects to is empty. You will not be able to proceed if JIRA detects that your external database contains any existing data.
- You cannot configure your database connection pool size through the Setup Wizard. You can do this subsequently using the JIRA Configuration Tool (included with all JIRA distributions except JIRA WAR) or by manually configuring your database connection (described on each specific database configuration guide). See Connecting JIRA to a Database for details.

Step 2 of 4: Application Properties

On this page you can configure some of JIRA's settings and enter your license key. For more details on the settings and what they mean, see Configuring JIRA Options.

JIRA will store your automated backups, file attachments and indexes within your JIRA Home Directory.

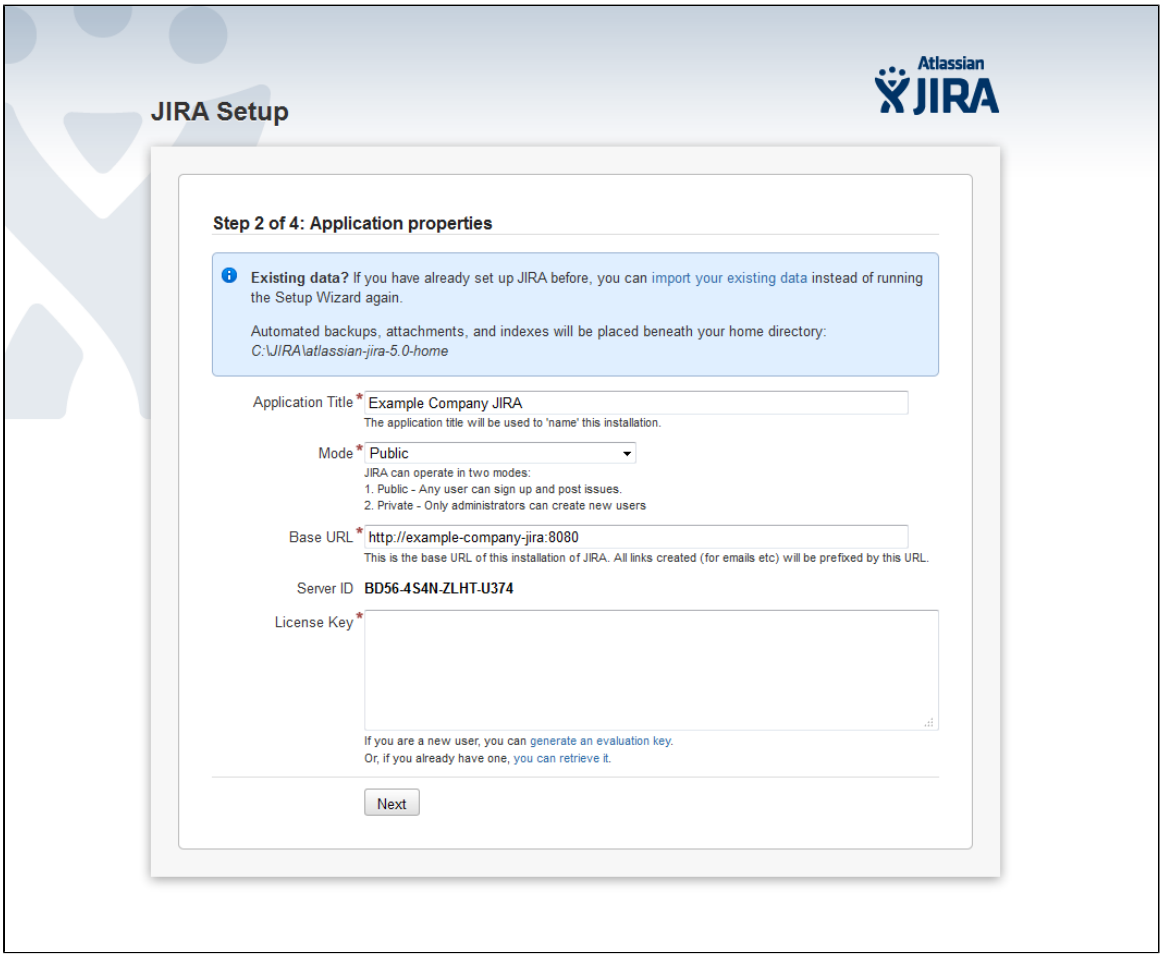
If you have an existing XML backup of a JIRA site, you can import it into your new JIRA installation at this point by clicking the top **import your existing data** link. If you choose this option, **please note the following points:**

- Your XML backup file must be located in the `import` subdirectory of your JIRA Home Directory.
- Your existing JIRA license details will be restored from your XML backup file, unless you specify different one in the **License** field of the **Import Existing Data** page.
- See Restoring Data for more details on using this feature.

License Key

You are required to enter a valid license key before you can use JIRA. You can obtain an evaluation license key which will allow JIRA to run unrestricted for 30 days. To use your existing license key or obtain a new license key, follow the steps below:

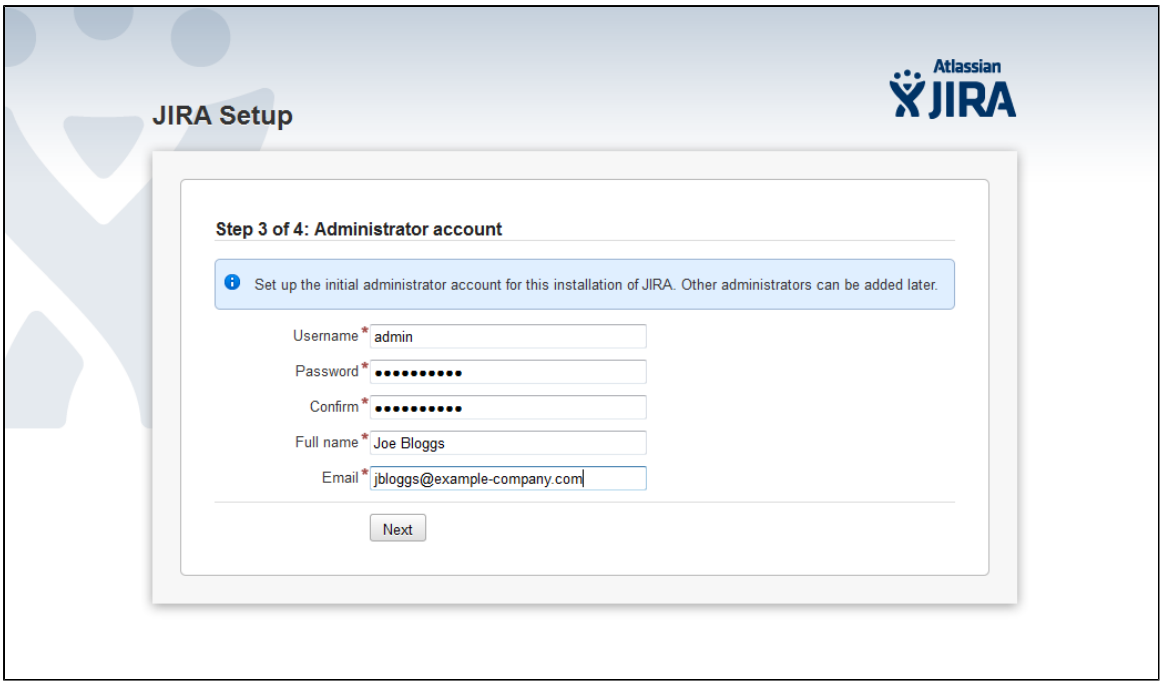
1. Do either of the following:
    - If you are a new user/customer, click the **generate an evaluation key** link to do just that.
      OR
    - If you are an existing customer and have an existing JIRA license, click the **you can retrieve it** link to retrieve your existing license.
2. Once you have either created an account or logged in with an existing account, the **Licenses** page will appear.
   If you logged in with an existing account, a list of your existing Atlassian product licenses will be shown. If you do not have any existing JIRA licenses, generate a new one by:
    - Clicking on the **New Evaluation License** link,
    - Selecting **JIRA** for the 'Product' field,
    - Completing the remaining fields and
    - Clicking the **Generate License** button to generate your new evaluation license.
      You will be taken back to the **License** page again.
3. On the **Licenses** page, copy the new evaluation license key to your clipboard and paste it into the **License Key** field of the Setup Wizard.



Step 3 of 4: Administrator Account

After completing the second step, the third page of the wizard sets up an administrator account.

Once this initial administrator account is created, that administrator can then create other administrators.

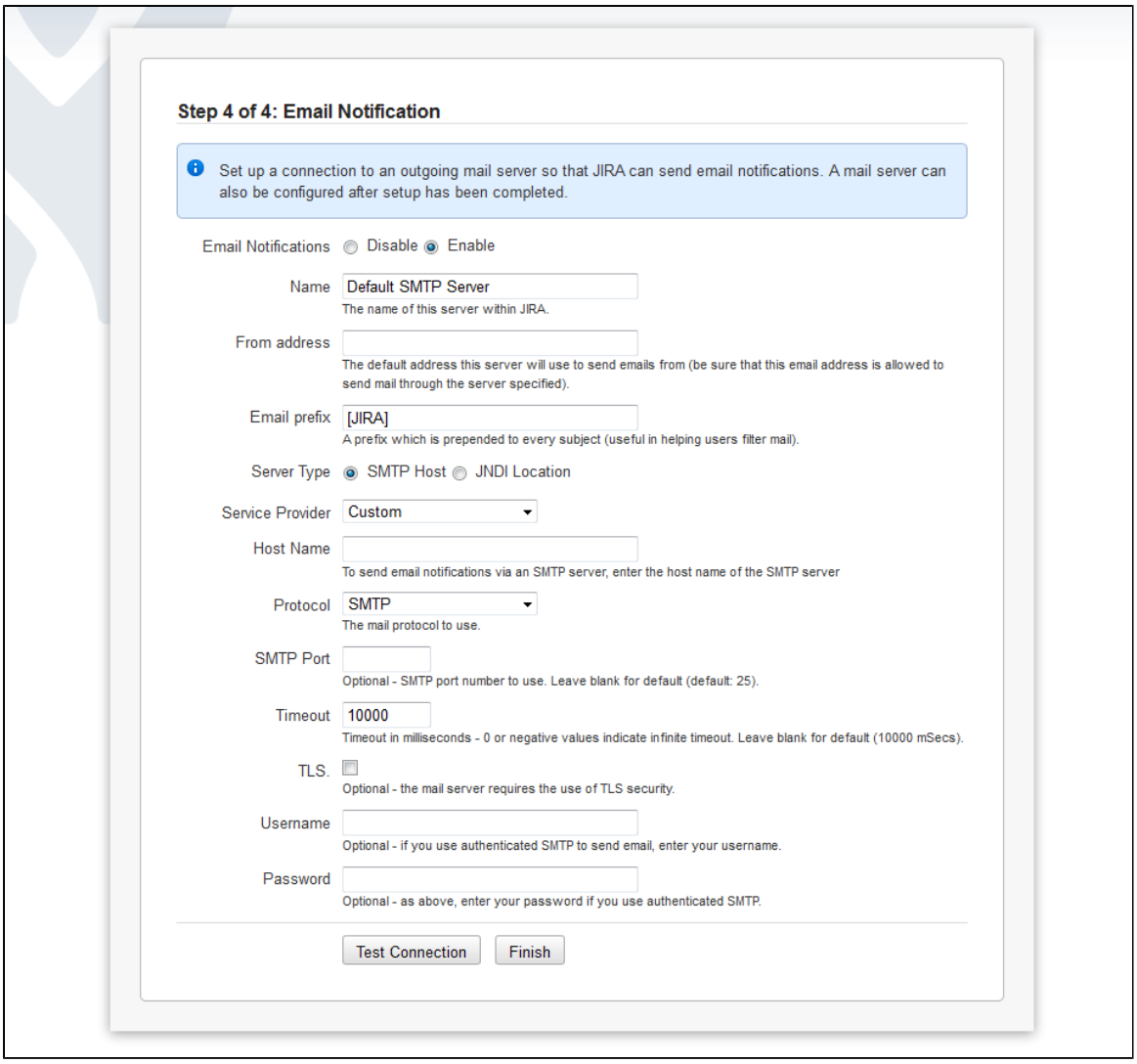


Step 4 of 4: Email Notification

The last page of the Setup Wizard allows you to configure the outgoing emails from JIRA.

To enable or disable email notifications (you can always enable them later), select the respective **Enable** or **Disable** radio buttons.

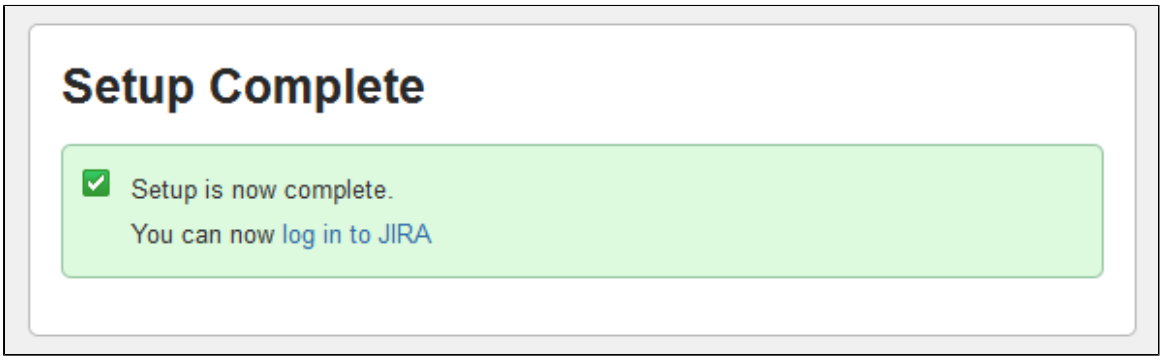
If you enable email notifications, you will need to set up a connection to a mail server, as indicated in the following screenshot.



You're done!

Once you complete step 4, JIRA should be set up and ready for use.

Click '**log in to JIRA**' to start using JIRA!

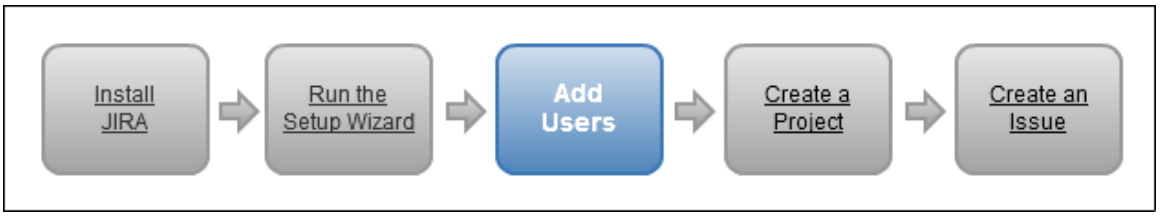


For information on getting started with JIRA, please refer to the following guides:

- JIRA User's Guide
- JIRA Administrator's Guide

Also note that you can click the help icon (yellow question-mark) in the upper right-hand corner at any time for further information.

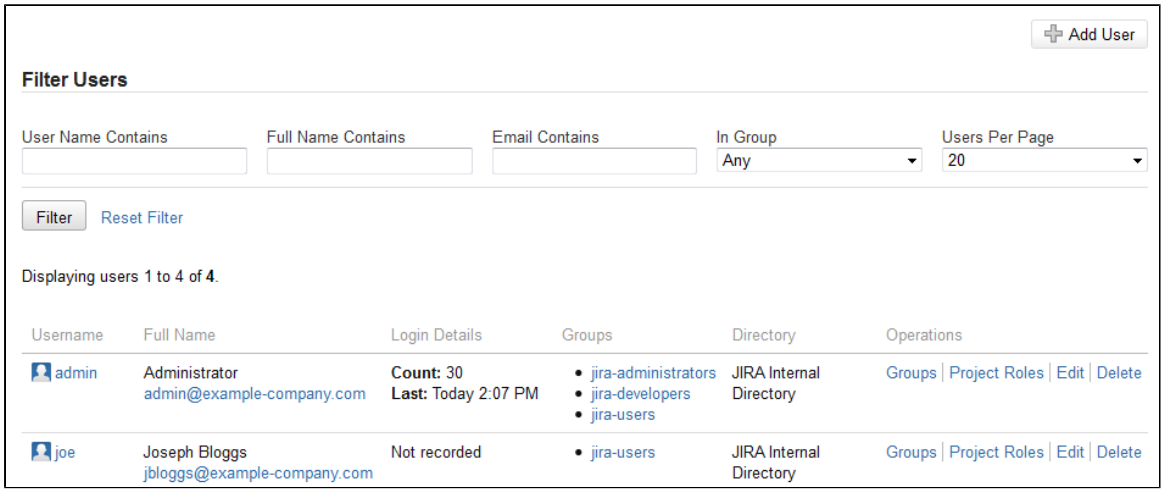
Next Steps - Adding Users



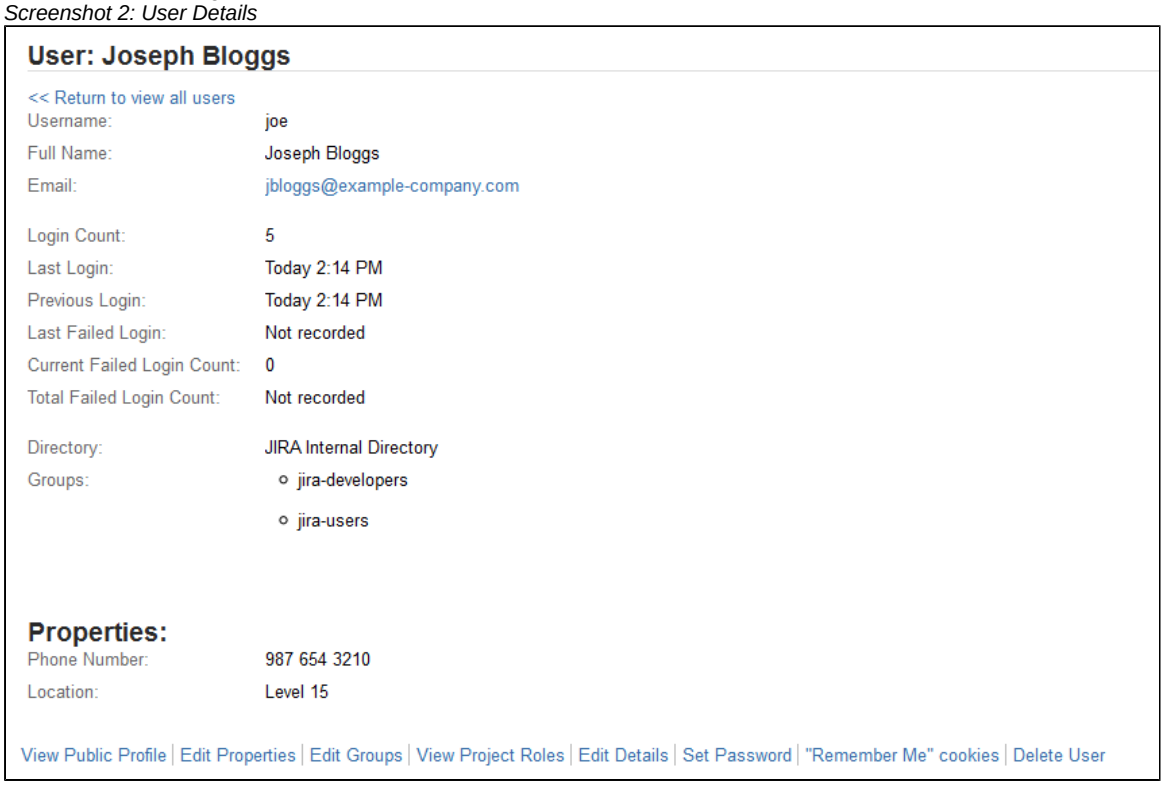
Viewing Users

To view a list of JIRA users:

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Select **Administration > Users > Users** to open the 'User Browser' page.
   Keyboard shortcut: **g + g +** start typing **users**
   Screenshot 1: The User Browser



3. To restrict the list of users shown in the **User Browser**, use the Filter form at the top of the User Browser.
   Specifying (part of) the user's username, full name, email address and/or group membership, then clicking the **Filter** button, will reduce the list to only those users who match those criteria.
4. To view details and login information about a user in the list, click their **Username** or **Email** Address.
   Screenshot 2: User Details

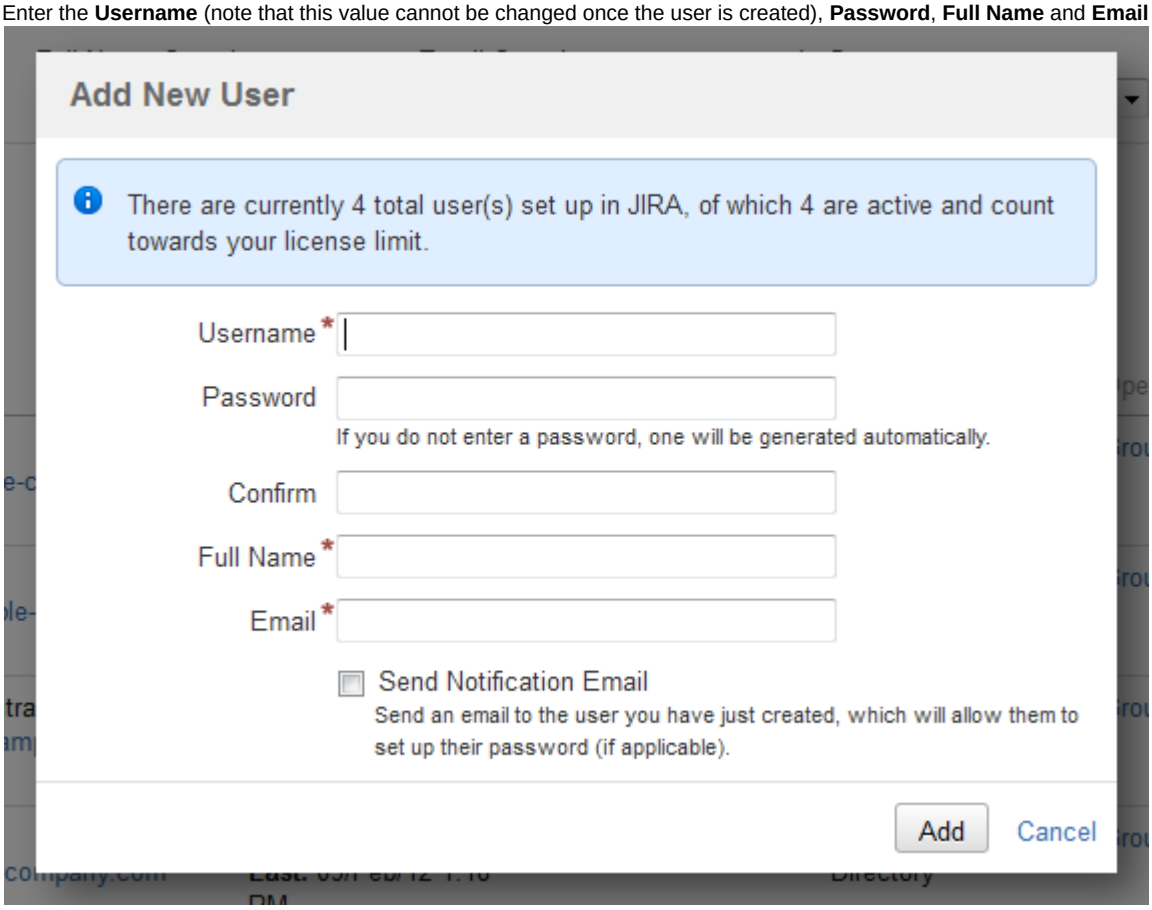


On this page:

- Viewing Users
- Adding a User
- Assigning a User to a Group
- Assigning a User to a Project Role
- Changing a User's Name or Email Address
- Changing a User's Password
- Adding a Property to a User
- Deactivating a User
- Deleting a User
- Notes

Adding a User

1. Open the **User Browser** (see Viewing Users above) and click the **Add User** button to open the 'Add New User' dialog box.
2. Enter the **Username** (note that this value cannot be changed once the user is created), **Password**, **Full Name** and **Email** address.



3. Optionally, select the **Send Notification Email** check box to send the user an email containing:
   - their login name; and
   - a link from which to set their password (this link is valid for 24 hours).
4. Click the **Create** button.

Users can also be created via:

- **Signup** — see Enabling Public Signup.
- **Email** — e.g. you can use a mail handler to allow JIRA to create new users based on the sender's email address (of email messages processed by JIRA). See Creating Issues and Comments from Email for more information about how to configure mail handlers for this purpose.
- **Connecting to an Internal Directory with LDAP Authentication** — see Copying Users on First Login.

Please Note: If you have a user limited license (e.g. starter license) and have reached your user limit, any further users added will not have permission to log in to JIRA.

Assigning a User to a Group

When a user is created, they will be added to any groups that are set up to have new users automatically added to them.

To change a user's group membership:

1. Locate the user in the **User Browser** (see Viewing Users above) and click the **Groups** link in the **Operations** column.

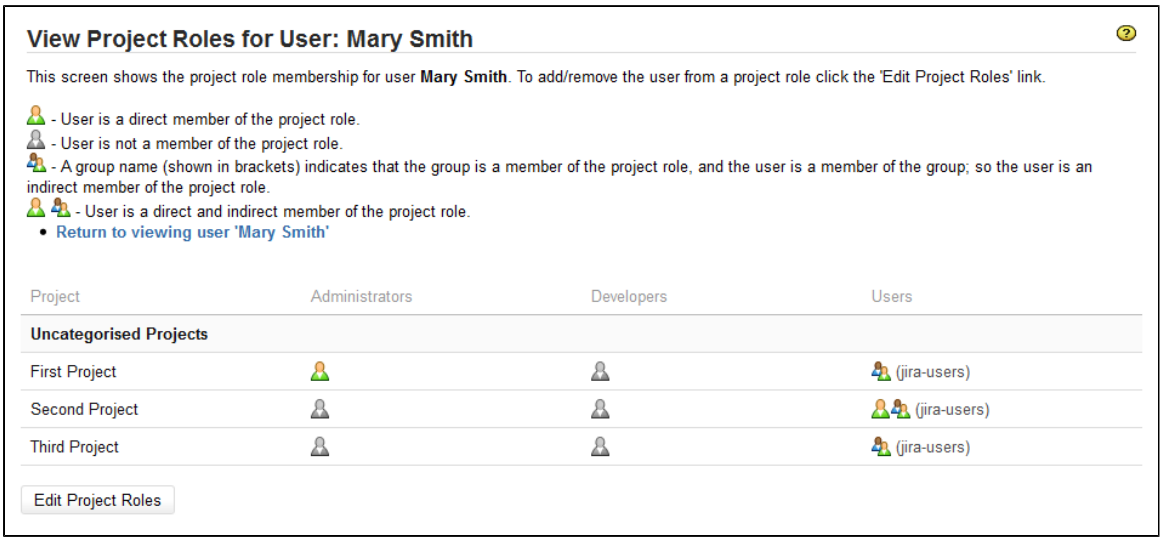
2. This will display two lists; the one on the left shows all available groups, and the one on the right shows all groups to which the user currently belongs. Use the **Join** and **Leave** buttons to add the user to or remove them from your selected group.

Please Note: If you have a user limited license (e.g. starter license) and have reached your user limit, you will not be able to assign any further users to groups with login permissions (i.e. jira-users permission) without first reducing the number of users with login permissions.

Assigning a User to a Project Role

Assigning a user to a project role enables them to fulfil a particular function in a particular project.

To view a user's project role membership, locate the user in the **User Browser** (see Viewing Users above) and click the **Project Roles** link in the **Operations** column. This will display a table showing all the projects and project roles that exist in JIRA, and the user's current project role membership for each project:



e.g. this screenshot shows that, for the 'First Project' project:

- Mary is a member of the 'Administrators' project role.
- Mary is not a member of the 'Developers' project role.
- Mary is indirectly a member of the 'Users' project role, through being a member of the 'jira-users' group.
  (Also note that, for the 'Third Project' project, Mary is both a direct *and* an indirect member of the 'Users' project role.)

Click the **Edit Project Roles** button. The check boxes will then be available for you to select (to add the user to a project role) or clear (to remove the user from a project role).

Changing a User's Name or Email Address

1. Locate the user in the **User Browser** (see Viewing Users above) and click their **Edit** link in the **Operations** column. This displays a form where you can change the user's Full Name or Email Address.
2. Click **Update** to confirm the change.

Changing a User's Password

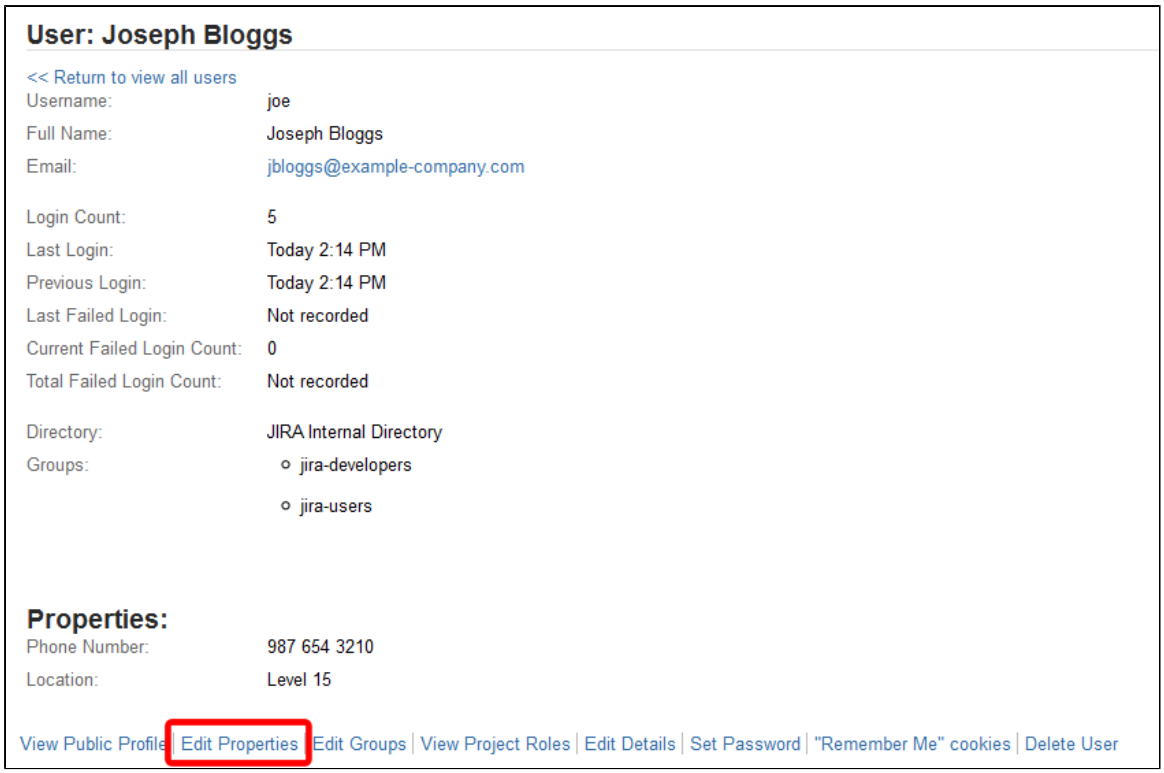
1. Locate the user in the **User Browser** (see Viewing Users above) and click their **Username**. This displays the user's details, below which are several links.
2. Click the **Set Password** link. This displays the **Set Password** screen.
3. Enter and confirm the new password.
4. Click the **Update** button.

Adding a Property to a User

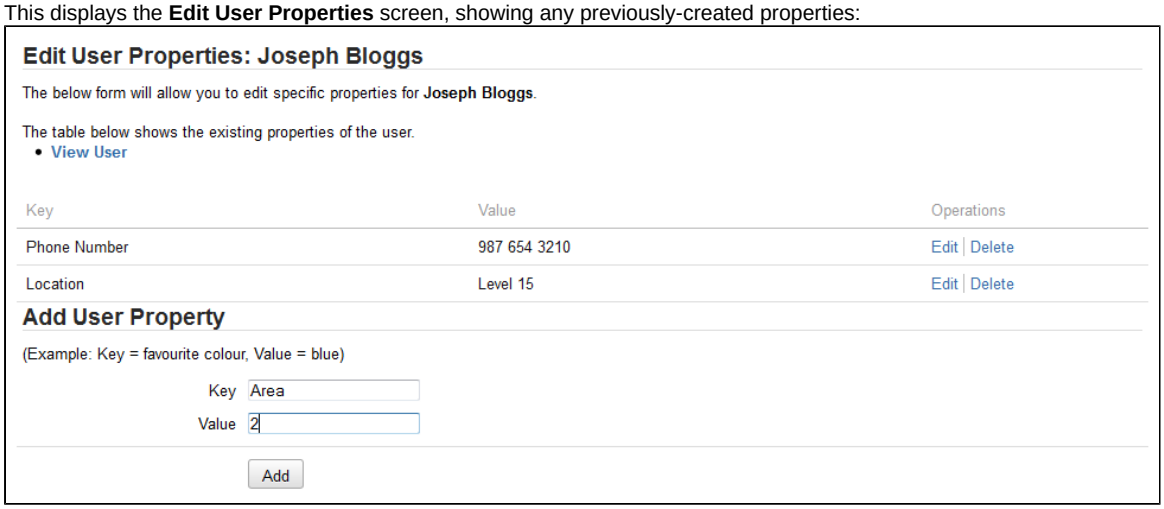
A **'Property'** is an extra piece of information that you can store regarding a user. A Property consists of a **Key** of your choice (eg. 'Phone number', 'Location') plus a corresponding **Value** (eg. '987 654 3210', 'Level Three').

To create a new Property for a user:

1. Locate the user in the **User Browser** (see Viewing Users above) and click their **Username**. This displays the user's details in a box.
2. Click the **Edit Properties** link towards the end of the page.



3. This displays the **Edit User Properties** screen, showing any previously-created properties:



4. Enter the new **Key** and its **Value**, then click the **Add** button.

Deactivating a User

Before you deactivate a user, you should:

- **Reassign any open issues assigned to that user**. You will need the '**Assign Issue**' permission to change the assignee for the issues.
- **Make sure the user is not the 'Default Assignee' for any project(s)**. You will need the '**Administer Project**' permission to change the Default Assignee for the project(s).

To deactivate a user account:

1. Remove the user from all groups. Read Managing Groups for more information.
2. Remove the user from all project roles. Read Managing Project Role Membership for more information.

Deactivating the user account will result in the following:

- the user will not count towards your **license limit**.
- **work log entries** associated with the user will remain.
- **filter subscriptions** will continue to be sent to the user — If this is a problem, you can change the user's email address in JIRA to an imaginary address, e.g. `user@example.com`.

There is currently no *"disabled user"* user type in JIRA.

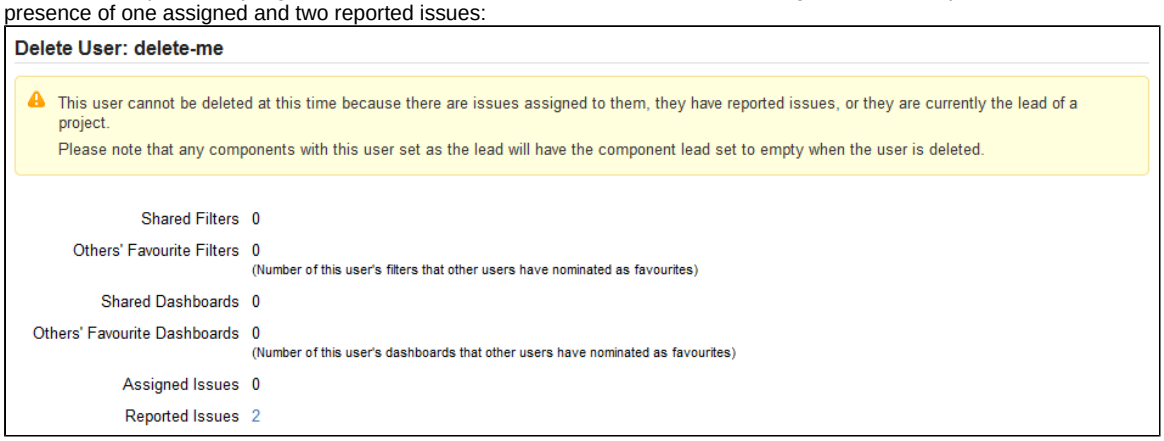
Deleting a User

Rather than deleting a user, we recommend that you **deactivate their account** instead (as described above). Deactivating a user's account will prevent that account from being used and prevent anyone from being able to log in to JIRA using that account. However, it will preserve that user's issues history.

Please Note: Before you delete a user, you should bulk-edit the issues involved and **change the reporter** to someone else. You will need the **'Modify Reporter'** permission to change the reporter for the issues. You will also need to allow editing of closed issues if some of the issues the user created are closed and you do not wish to reopen them.

To delete a user:

1. Locate the user in the **User Browser** (see Viewing Users above) and click the **Delete** link in the **Operations** column.
2. The confirmation screen that follows will summarise any involvement of that user in the system by showing current issues assigned to and reported by that user, etc. These connections between the user and other parts of the system may prevent the deletion of that user. For example, attempting to delete a user called *test-user* results in the following screen, which prevents deletion due to the presence of one assigned and two reported issues:



   As well as reassigning any issues, you may need to bulk-edit the issues created by the user and change the 'Reporter' to someone else. You'll need the 'Modify Reporter' permission to do this.
3. If there are no issues assigned to, or reported by the user, the confirmation screen will display a **Delete** button; click this to proceed with the deletion.

Please Note:

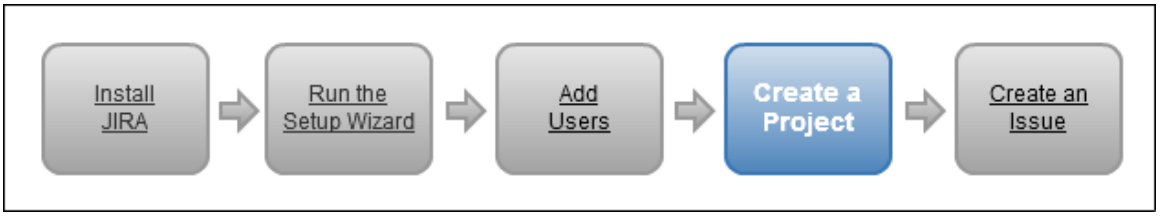
- Please note that the filters and dashboards of a user will be deleted when the user is deleted, regardless of whether the filters or dashboards are shared with other users.
- Any numbers of issues which have been reported by or assigned to the user you are attempting to delete, are respectively hyperlinked to a list of the individual issues (in the Issue Navigator).

Notes

- If you are using External User Management, you will not be able to create, edit or delete users from within JIRA; but you can still assign users to project roles, and create/edit/delete user properties.
- If you have JIRA connected to either a delegated LDAP directory or an LDAP directory set to 'Read Only' (see Connecting to an LDAP Directory for details), you will not be able to change a user password from within JIRA.
- **Multiple user directories:** You may define multiple user directories in JIRA, so that JIRA looks in more than one place for its users and groups. For example, you may use the default JIRA **internal directory** and also connect to an **LDAP** directory server. In such cases, you can define the **directory order** to determine where JIRA looks first when processing users and groups. Here is a summary of how the directory order affects the processing:
  - The order of the directories is the order in which they will be searched for users and groups.
  - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

  See Managing Multiple Directories.

Next Steps - Creating a Project



Creating a Project

To add a new project in JIRA:

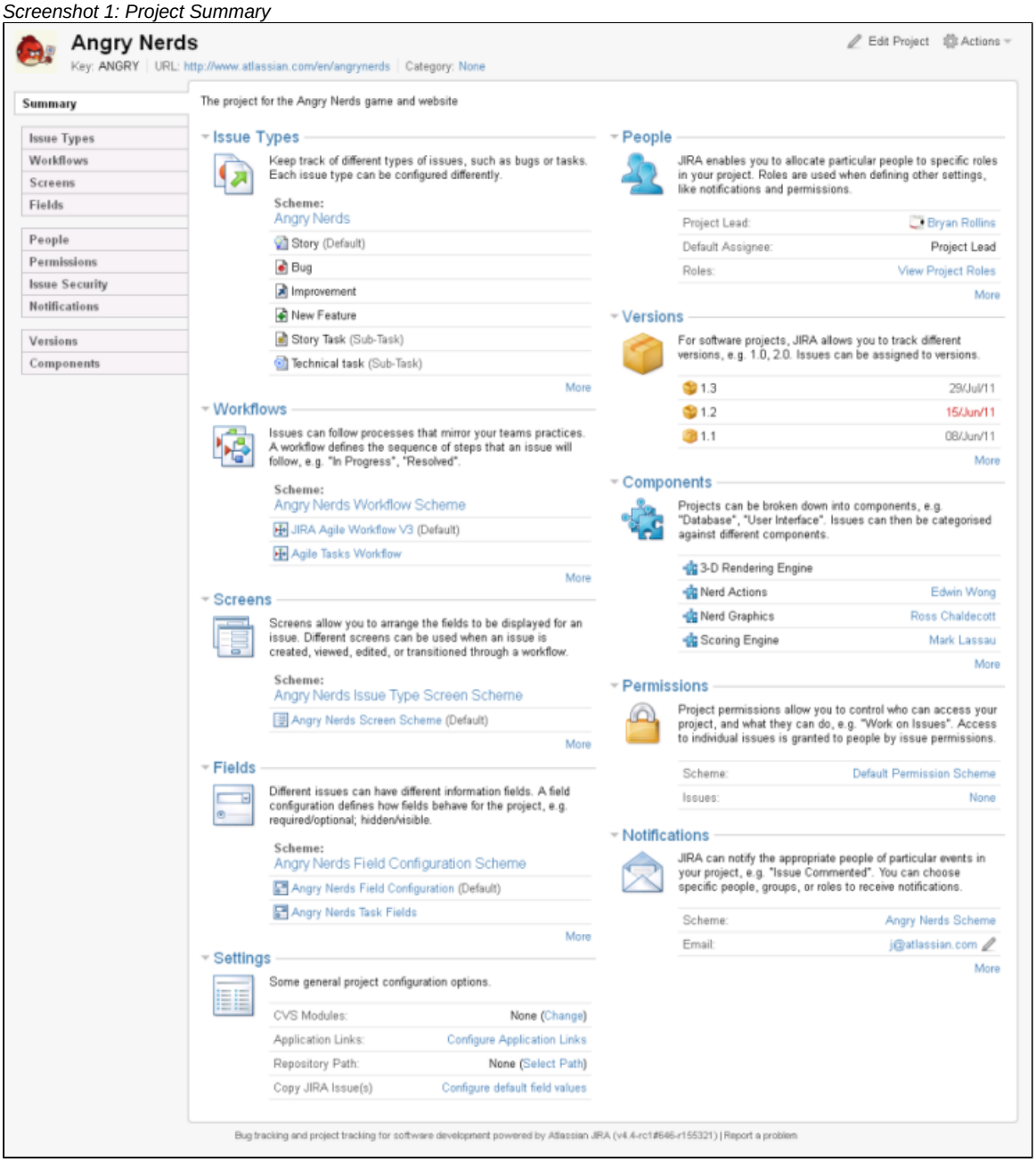
1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click the '**Administration**' link at the top of the screen.

3. Select **'Projects' > 'Projects'** then click **'Add Project'** to display the 'Add Project' screen (see Screenshot 2below).
    - **Name** — type a descriptive name. This can be changed later if you wish.
    - **Key** — type a 'key' unique to this project (eg. 'WEB'). This will be used as the prefix of this project's issue keys (e.g. 'WEB-100'). We recommend that you define a key that describes the project and is easy to type. Please note that the key cannot be changed once the project is created.
    - **Project Lead** — choose the person who will manage this project. You can change the Project Lead later if you wish. Note that issues can be automatically assigned to the Project Lead (see 'People' below).

      If there is only one user in your JIRA system, the Project Lead will default to that person and this field will not be available.
4. Once created, the 'Project Summary' screen for your new project will be displayed (see Screenshot 1 below). You can then configure other details of your project as described below.

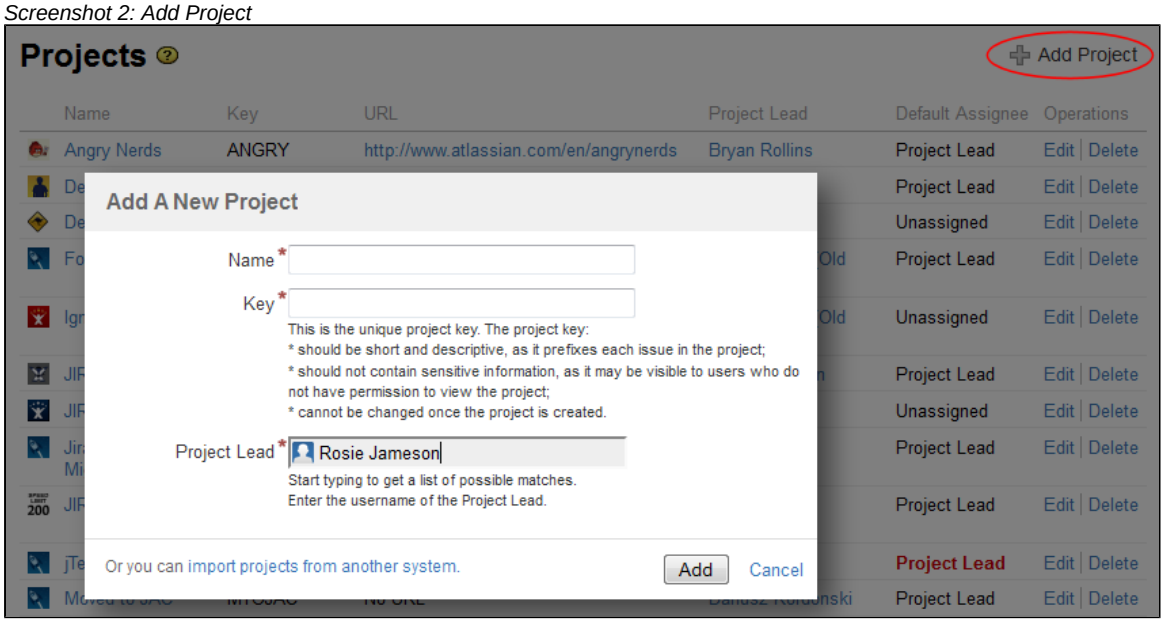
    Here is what a project looks like once created:

    Screenshot 1: Project Summary



(View full size)

Screenshot 2: Add Project



On this page:

- Creating a Project
- Configuring a Project
  - Project Details
  - Issue Types
  - Workflows
  - Screens
  - Fields
  - Settings
  - People
  - Versions
  - Components
  - Permissions
  - Notifications
- A note about Project Administrators

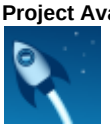
Configuring a Project

To configure a project in JIRA:

1. Log in as a user with the **'JIRA Administrators'** global permission.
2. Click the '**Administration**' link at the top of the screen.
3. Click **'Projects'** and select the project of interest. The 'Project Summary' screen will be displayed (see Screenshot 1 above).
   Keyboard shortcut: **'g' + 'g' +** start typing **'project'**

You can then edit the project's configuration settings as follows:

Project Details

- **Name** — type a descriptive name. This can be changed later if you wish.
- **URL** — an optional URL associated with this project, eg. pointing to project documentation.
- **Project Avatar** — an image (48x48 pixels) that represents the project. You can either use the default image, i.e.:
  
  or choose a different image. If you prefer not to use an image, simply upload a transparent pixel.
- **Description** — an optional description of this particular project. You can include HTML, but make sure all your tags are closed.
  Warning: Please be aware that this is completely unfiltered HTML and as such, it is susceptible to cross site scripting attacks.

Click the link next to the 'Category' field under the project name to assign the project into a logical category/group. This is useful for managing multiple related projects. If no categories exist, click the '**Add**' link on the following page to add a new category. New categories can also be created via **'Administration' > 'Projects' > 'Project Categories'**.

Issue Types

JIRA enables you to keep track of different types of things — bugs, tasks, helpdesk tickets, etc — by using different *issue types*. You can also configure each issue type to act differently, e.g. to follow a different process flow or track different pieces of information.

- **Issue Type Scheme** — the project's issue type scheme determines which issue types apply to this project.

Workflows

Your JIRA issues can follow a process that mirrors your team's practices. A *workflow* defines the sequence of steps (or statuses) that an issue will follow, e.g. Open, In Progress, Resolved. You can configure how issues will transition between statuses, e.g. who can transition them, under what conditions, and which screen will be displayed for each transition.

- **Workflow Scheme** — the project's workflow scheme determines which workflows (issue state transitions) apply to issue types in this project.

Screens

JIRA allows you to display particular pieces of issue information at particular times, by defining *screens*. A screen is simply a collection of fields. You can choose which screen to display when an issue is being created, viewed, edited, or transitioned through a particular step in a workflow.

- **Screen Scheme** — the project's screen scheme determines which screens are displayed for different issue operations (view, edit, create);
  OR
  Issue Type Screen Scheme — the project's issue type screen scheme determines which screens are displayed for different issue operations (view, edit, create), for different issue types.

Fields

JIRA enables you to define field behaviour: each field can be required/optional, rich text/plain text, hidden/visible. You define this behaviour by using a *field configuration*.

- **Field Configuration Scheme** — the project's field configuration scheme determines which field configuration applies to issue types in this project. (A field configuration determines each field's overall visibility, requiredness, formatting (wiki/rich-text or plain) and help-text).

Settings

- **CVS Modules** — configures CVS integration for this project.
- **Application Links** — projects or other entities on other applications or sites to which this JIRA project has been linked via application links. New project/entity links can be created by clicking the '**Configure Application Links**' link. See Adding Project Links between Applications for details.

People

Different people may play different roles in different projects — the same person may be a leader of one project but an observer of another project. JIRA enables you to allocate particular people to specific roles in your project.

- **Project Lead** — user fulfilling the role of project leader. Used as the 'Default Assignee' (see below), and potentially elsewhere in JIRA (e.g. in permission schemes, notification schemes, issue security schemes and workflows).
- **Default Assignee** — the user to whom issues in this project are initially assigned when created. Can be either the 'Project Lead' (above), or, if **Allow unassigned issues** is set to 'On' in JIRA's general configuration, 'Unassigned'. There are also default component assignees.
- **Project Roles** — members are users/groups who fulfil particular functions for this project. Project roles are used in permission schemes, notification schemes, issue security schemes and workflows.

Versions

If you are using JIRA to manage the development of a product, you may want to define different *versions* to help you track which issues relate to different releases of your product (e.g. 1.0, 1.1, 1.2, 2.0 beta, 2.0). JIRA can help you manage, release and archive your versions. Versions can also have a Release Date, and will automatically be highlighted as "overdue" if the version is unreleased when this date passes.

- **Versions** — versions defined in the project. See the version management page for details.

Components

You may want to define various *components* to categorise and manage different issues. For a software development project, for example, you might define components called "Database", "Usability", "Documentation" (note that issues can belong to more than one component). You can choose a Default Assignee for each component, which is useful if you have different people leading different sub-teams in your project.

- **Components** — logical groups that this project's issues can belong to. See the component management page for details.

Permissions

JIRA allows you to control who can access your project, and exactly what they can do (e.g. "Work on Issues", "Comment on Issues", "Assign Issues"), by using *project permissions*. You can also control access to individual issues by using *security levels*. You can choose to grant access to specific users, or groups, or roles (note that roles are often the easiest to manage).

- **Permission Scheme** — the project's permission scheme determines who has permission to view or change issues in this project.
- **Issue Security Scheme** — the project's issue security scheme determines what visibility levels issues in this project can have (see issue-level security).

Notifications

JIRA can notify the appropriate people when a particular event occurs in your project (e.g. "Issue Created", "Issue Resolved"). You can choose specific people, or groups, or roles to receive *email notifications* when different events occur. (Note that roles are often the easiest to manage.)

- **Notification Scheme** — the project's notification scheme determines who receives email notifications of changes to issues in this project.
- **Email** — specifies the 'From' address for emails sent from this project. Only available if an SMTP email server has been configured in JIRA.

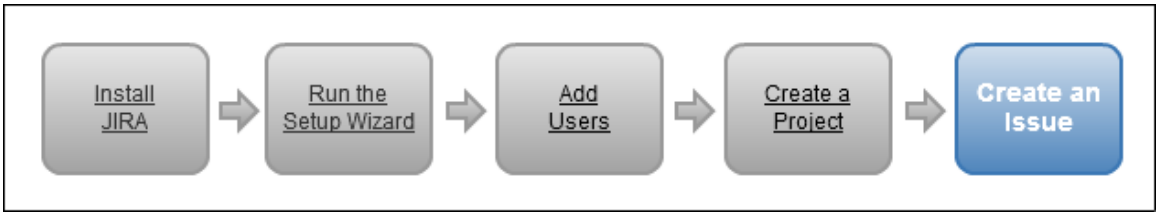
A note about Project Administrators

A JIRA project administrator is someone who has the project-specific 'Administer Project' permission, but not necessarily the global 'JIRA Administrator' permission.

A project administrator can:

- Edit the project name
- Edit the project description
- Edit the project avatar image
- Edit the URL
- Edit the Project Lead
- Edit project role membership
- Configure entities for application links
- Define project components
- Define project versions
- View — but not edit — the project's schemes (notification scheme, permission scheme, etc)

Next Steps - Creating an Issue



To create a JIRA issue, you need the **Create Issue** project permission for the issue's relevant project. If you do not have this permission, please contact your JIRA administrator.

To create a new JIRA issue:

1. Click the **Create Issue** link (at the top-right of the JIRA user interface) to open the **Create Issue** dialog box.
   Keyboard shortcut: **c**
   While browsing a project, you can click one of the **issue type** icons such as **Bug**, **New Feature**, **Task** or **Improvement**.
2. Select the relevant **Project** and **Issue Type** on the **Create Issue** dialog box.
3. Type a **Summary** for the issue and complete any appropriate fields — at least required ones which are marked by an asterisk.
   If you want to access fields which are not shown on this dialog box or you want to hide existing fields:
   a. Click the **Configure Fields** button.
   b. Click **Custom** and select the fields you want to show or hide by selecting or clearing the relevant check boxes, respectively, or click **All** to show all fields.
      When you next create an issue, JIRA remembers your last choice of selected fields.
4. *(Optional)* If you want to create a series of similar issues (with the same **Project** and **Issue Type**), select the **Create another** check

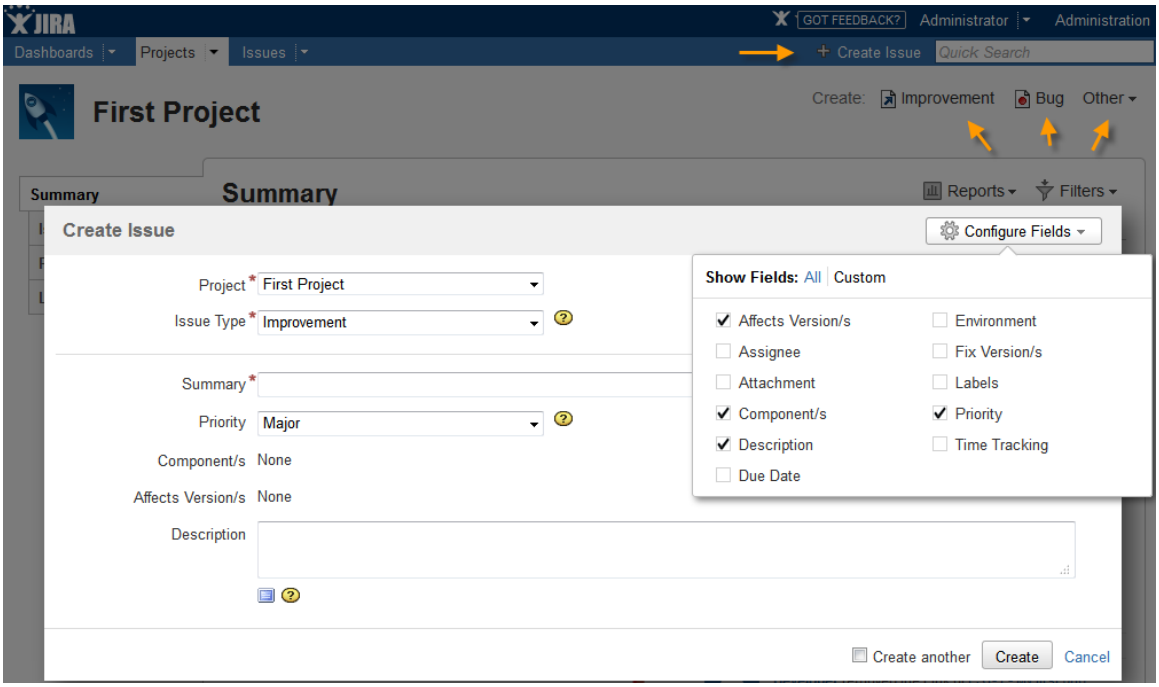
box.

5. Click the **Create** button to create the issue.
   If you selected the **Create another** check box (above), a new **Create Issue** dialog box will appear after your issue is created, automatically pre-populated with your previous issue details, while leaving the **Summary** field blank.

Tips:

- You can mention other users in the **Description** or **Comment** field so that an email message will be sent to the user's email address (registered with their JIRA account) upon clicking the **Update** button. See Emailing an issue to users by mentioning them for details.
- To see a list of all issues that you have created, which have not yet been resolved, go to your user name and select **Profile** and on your profile, click **Filters** > **Reported & Open**.
- With appropriate configuration by your JIRA administrator, it is also possible to create issues via email.

Screenshot: Example 'Create Issue' dialog box



Related topics

Sharing a Search Result

Connecting JIRA to a Database

JIRA requires a relational database to store its issue data.

If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure a database connection for you to either JIRA's internal HSQL database or an external database.

JIRA's internal HSQL database is suitable for evaluation purposes. However, HSQL databases are prone to corruption. For production installations of JIRA, we strongly recommend that you connect JIRA to another supported database. This allows you to take advantage of your database system's own backup and recovery features.

The following are more detailed instructions for configuring a connection to a JIRA database:

- Connecting JIRA to PostgreSQL
- Connecting JIRA to MySQL
- Connecting JIRA to Oracle
- Connecting JIRA to SQL Server 2005
- Connecting JIRA to SQL Server 2008
- Connecting JIRA to HSQLDB

Which Database?

Your choice of database can significantly affect your subsequent experience of JIRA administration. If you have a choice of databases, please first read our list of supported databases.

If you are looking for a low-cost solution, consider using PostgreSQL or MySQL as both of these are open source (free) software.

Upgrading JIRA or Migrating JIRA to Another Server?

If you are upgrading JIRA manually or migrating JIRA to another server and do not have access to a pre-existing `dbconfig.xml` file, you will need to re-configure your database connection. This results in a `dbconfig.xml` file (being created in the JIRA Home Directory of your new JIRA installation), whose content defines your JIRA database connection.

The options for re-configuring your database connection depend on what JIRA distribution you are using:

'Recommended' distributions:	If you installed JIRA using the 'Windows Installer', 'Linux Installer' or from an 'Archive File', you can re-configure your database connection either with the JIRA Configuration Tool or manually.
WAR distribution:	If you have set up a JIRA WAR installation, you need to manually configure your database connection.

Specific instructions for configuring database connections either using the JIRA Configuration Tool or manually are provided in the specific instructions for each database (listed above).

Data Migration

To transfer your issue data from one database to another, please refer to the instructions for Switching databases.

Connecting JIRA to PostgreSQL

These instructions will help you connect JIRA to a PostgreSQL 8.2+ database.

Please Note:

- If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure a PostgreSQL database connection for you.
- A version of these instructions specific to Linux and JIRA is available.

1. Before You Begin

1.1 Are You Migrating JIRA to Another Server?

If you are Migrating JIRA to Another Server, create an export of your data as an XML backup. You will then be able to transfer data from you old database to your new database, as described in Switching databases.

1.2 Shut Down JIRA

On this page:

- 1. Before You Begin
  - 1.1 Are You Migrating JIRA to Another Server?
  - 1.2 Shut Down JIRA
- 2. Create and Configure the PostgreSQL Database
- 3. Copy the PostgreSQL JDBC Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your PostgreSQL Database
  - 4.1 Connecting JIRA to PostgreSQL Using the JIRA Configuration Tool
  - 4.2 Connecting JIRA to PostgreSQL Manually
- 5. Next steps
- Installation notes

2. Create and Configure the PostgreSQL Database

1. Create a database user which JIRA will connect as (e.g. **`jiradbuser`**). Remember this database user name, as it will be used to configure JIRA's connection to this database in subsequent steps.
2. Create a database for JIRA to store issues in (e.g. **`jiradb`**) with Unicode collation.

```
CREATE DATABASE jiradb WITH ENCODING 'UNICODE';
```

Or from the command-line:

```
$ createdb -E UNICODE jiradb
```

3. Ensure that the user has permissions to connect to the database, and to create and write to tables in the database.
   Remember this database name, as it will be used to configure JIRA's connection to this database in subsequent steps.

3. Copy the PostgreSQL JDBC Driver to Your Application Server (JIRA WAR Only)

Skip this step if you installed a 'Recommended' distribution of JIRA, which (unlike JIRA WAR) includes the PostgreSQL JDBC driver.

1. Download the PostgreSQL JDBC driver from http://jdbc.postgresql.org/download.html. (Obtain version 8.4 of the **JDBC 4** driver.)
2. Add the PostgreSQL JDBC driver jar to the `lib/` directory of your application server.

4. Configure Your JIRA Server to Connect to Your PostgreSQL Database

There are two ways to configure your JIRA server to connect to your PostgreSQL database:

- Using the JIRA Configuration Tool (except JIRA WAR distributions) or
- Manually

Please Note:

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. We recommend using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to configure JIRA with PostgreSQL manually.

4.1 Connecting JIRA to PostgreSQL Using the JIRA Configuration Tool

The JIRA Configuration Tool is not available with JIRA WAR distributions.

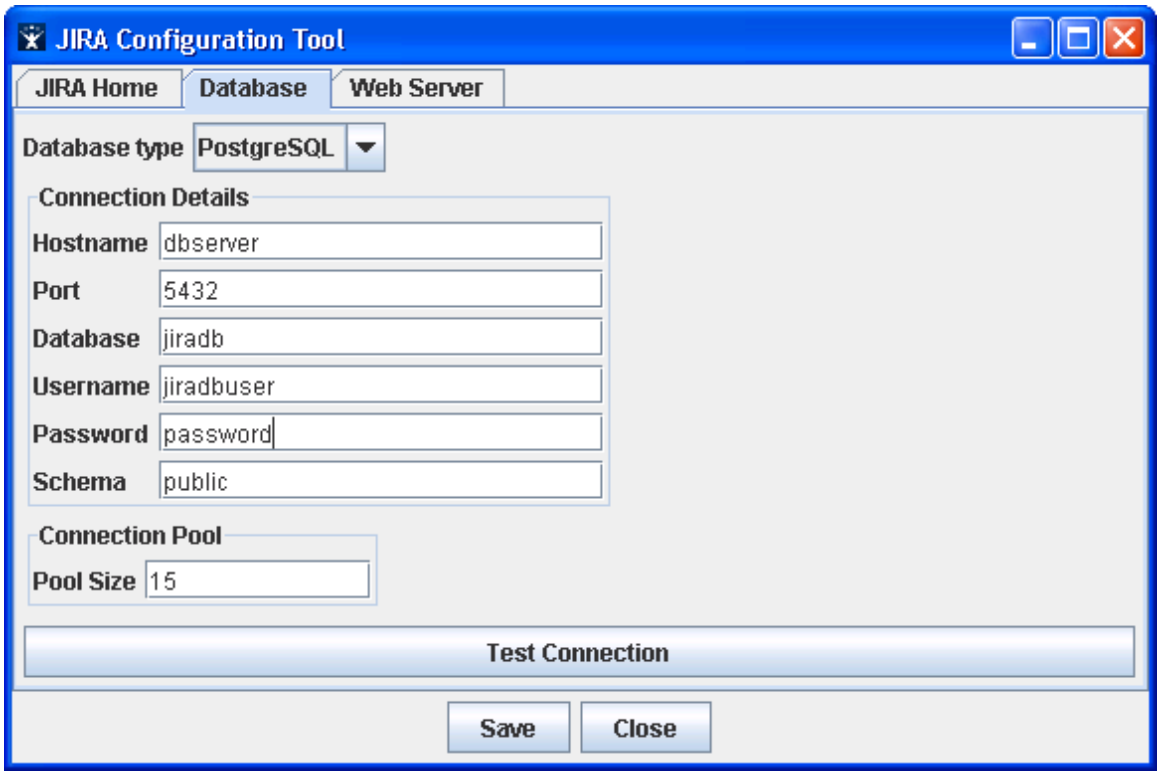
To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

Please Note: You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

To connect JIRA to PostgreSQL using the JIRA Configuration Tool:

1. Click the '**Database**' tab.
2. From the 'Database type' drop-down choose '**PostgreSQL**'.
   The JIRA Configuration Tool will display your current database configuration settings if any are already set.



3. Fill in the connection details for your PostgreSQL database.
    - 'Hostname' — The name or IP address of the machine that the PostgreSQL server is installed on.
    - 'Port' — The TCP/IP port that the PostgreSQL server is listening on. You can leave this blank to use the default port.
    - 'Database' — The name of your PostgreSQL database (into which JIRA will save its data).
    - 'Username' — The user that JIRA uses to connect to the PostgreSQL server.
    - 'Password' — The user's password to authenticate with the PostgreSQL server.
    - 'Schema' — The name of the schema that your PostgreSQL database uses.

      Please ensure your database's schema name is lower-case as JIRA cannot work with PostgreSQL databases whose schema names contain upper-case characters.
4. JIRA keeps a pool of database connections open to the database server. You can set the maximum size of this pool in the 'Pool Size' text field.
5. After typing in your settings, click the '**Test Connection**' button to test the connection settings. The tool will attempt to connect to the database, and give a message with the results.
6. Click '**Save**' to save your settings when you are done.

   Please Note:
    - The JIRA Configuration Tool will save your database configuration to a `dbconfig.xml` file in your JIRA Home Directory.
    - JIRA must be restarted for your new settings to take effect.

Congratulations — you have finished! Proceed to 'Next Steps' below.

4.2 Connecting JIRA to PostgreSQL Manually

1. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.

   If this file does not exist, create the file, copy and paste the example XML code below into this file and edit the pasted XML as required.

   Please Note:

   Ensure that the `<database-type/>` element's content specifies your type of database, as shown below. If you forget to do this and you start JIRA, your database tables may be created incorrectly. Refer to our Incorrect database type specified documentation if this happens to you.
   PostgreSQL 7.2 and later requires a schema to be specified in the `<schema-name/>` element. If your PostgreSQL database uses the default '`public`' schema, this should be specified in the `<schema-name/>` element as shown below. Please ensure your database's schema name is lower-case as JIRA cannot work with PostgreSQL databases whose schema names contain upper-case characters.

   When editing your `dbconfig.xml` file, escape any '`&`' characters by adding '`amp;`' to the end of each one.

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>postgres72</database-type>
  <schema-name>public</schema-name>
  <jdbc-datasource>
    <url>jdbc:postgresql://dbserver:5432/jiradb</url>
    <driver-class>org.postgresql.Driver</driver-class>
    <username>jiradbuser</username>
    <password>[enter db password]</password>
    <pool-size>15</pool-size>
  </jdbc-datasource>
</jira-database-config>
```

2. Save your edited `dbconfig.xml` file (at the root of your JIRA Home Directory).
   JIRA must be restarted for your new settings to take effect.

5. Next steps

You should now have JIRA configured to connect to your PostgreSQL database.

- If you are using a 'recommended' distribution of JIRA, start it up and watch the logs for any errors.
- If you are using the JIRA WAR distribution, rebuild and redeploy the webapp in your application server.

Installation notes

Please see JIRA and PostgreSQL.

Connecting JIRA to MySQL

These instructions will help you connect JIRA to a MySQL 5.x database.

Please Note:

- If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure a MySQL database connection for you
- A Linux-specific version of these instructions is available.

1. Before You Begin

1.1 Are You Migrating JIRA to Another Server?

If you are Migrating JIRA to Another Server, create an export of your data as an XML backup. You will then be able to transfer data from you old database to your new database, as described in Switching databases.

1.2 Shut Down JIRA

1.3 Known Issues with MySQL

If you are using a MySQL database with any of the following:

- JIRA 3.13 or above,
- version 5.5.25 or higher of Tomcat 5,
- version 6.0.13 or higher of Tomcat 6,

you may experience problems with your connections dropping out (see JRA-15731 for details). Please read Surviving Connection Closures for information on the changes required to your application server to address this.

If you experience any other issues with using JIRA and MySQL, please refer to our additional notes for more information.

On this page:

- 1. Before You Begin
  - 1.1 Are You Migrating JIRA to Another Server?
  - 1.2 Shut Down JIRA
  - 1.3 Known Issues with MySQL
- 2. Create and Configure the MySQL Database
- 3. Copy the MySQL JDBC Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your MySQL Database
  - 4.1 Connecting JIRA to MySQL Using the JIRA Configuration Tool
  - 4.2 Connecting JIRA to MySQL Manually
  - 4.3. Additional Notes
- 5. Next Steps
- Installation Notes

2. Create and Configure the MySQL Database

1. Create a database user which JIRA will connect as (e.g. **`jiradbuser`**). ✅ Remember this database user name, as it will be used to configure JIRA's connection to this database in subsequent steps.
2. Create a database for JIRA to store issues in (e.g. **`jiradb`**). The database must have a character set of UTF8. Enter the following command from within the MySQL command client:

   ```
   create database jiradb character set utf8;
   ```

   (if you want your database to be named **`jiradb`**). ✅ Remember this database name, as it will be used to configure JIRA's connection to this database in subsequent steps.
3. Ensure that the user has permission to connect to the database, and permission to create and populate tables.

3. Copy the MySQL JDBC Driver to Your Application Server (JIRA WAR Only)

⚠ Skipthis step if you installed a 'Recommended' distribution of JIRA, which (unlike JIRA WAR) includes the MySQL JDBC driver.

1. Download the MySQL Connector/J JDBC driver v5.1
   ⚠ A user had reported experiencing problems with the Resin JDBC driver for MySQL. However, the Connector/J driver from MySQL is known to work correctly.
2. Add the MySQL JDBC driver jar (**`mysql-connector-java-5.x.x-bin.jar`**) to the `lib/` directory.

4. Configure Your JIRA Server to Connect to Your MySQL Database

There are two ways to configure your JIRA server to connect to your MySQL database:

- Using the JIRA Configuration Tool (except JIRA WAR distributions) or
- Manually

ℹ **Please Note:**

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. We recommend using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to configure JIRA with MySQL manually.

4.1 Connecting JIRA to MySQL Using the JIRA Configuration Tool

⚠ The JIRA Configuration Tool is not available with JIRA WAR distributions.

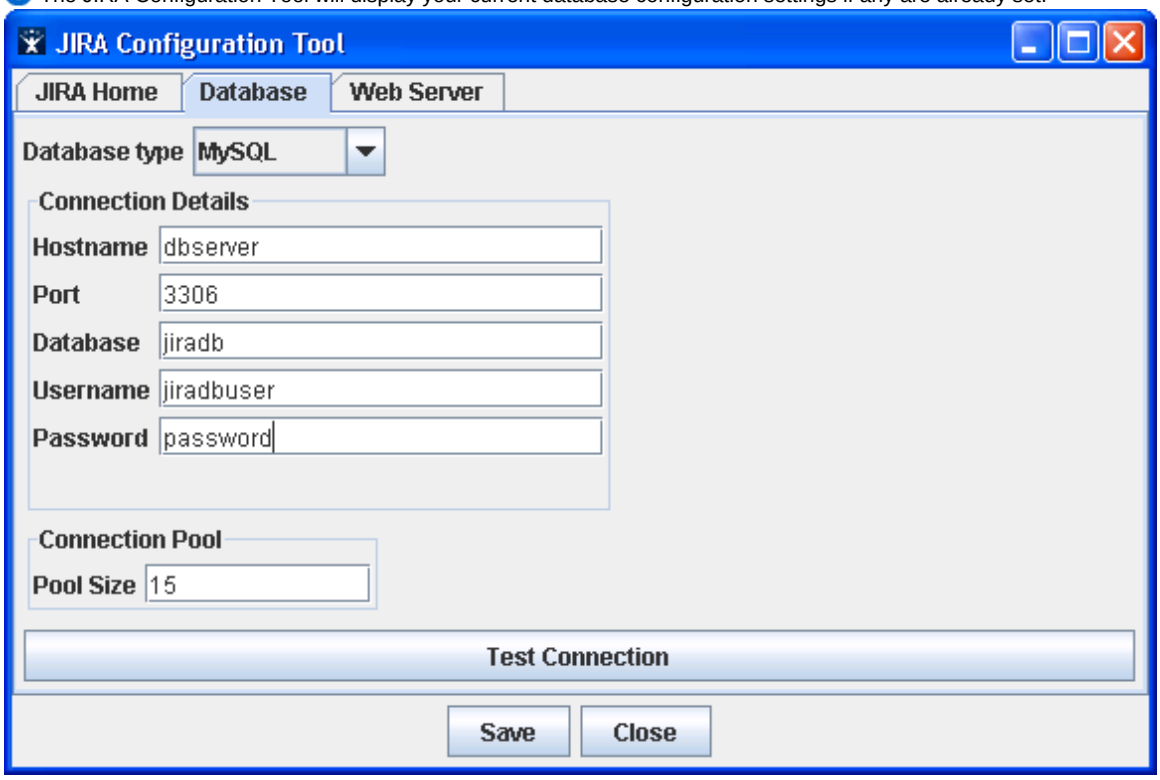
To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

ℹ **Please Note:** You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

To connect JIRA to MySQL using the JIRA Configuration Tool:

1. Click the '**Database**' tab.
2. From the 'Database type' drop-down choose '**MySQL**'.
   The JIRA Configuration Tool will display your current database configuration settings if any are already set.



3. Fill in the connection details for your MySQL database.
   - 'Hostname' — The name or IP address of the machine that the MySQL server is installed on.
   - 'Port' — The TCP/IP port that the MySQL server is listening on. You can leave this blank to use the default port.
   - 'Database' — The name of your MySQL database (into which JIRA will save its data).
   - 'Username' — The user that JIRA uses to connect to the MySQL server.
   - 'Password' — The user's password to authenticate with the MySQL server.
4. JIRA keeps a pool of database connections open to the database server. You can set the maximum size of this pool in the 'Pool Size' text field.
5. After typing in your settings, click the '**Test Connection**' button to test the connection settings. The tool will attempt to connect to the database, and give a message with the results.
6. Click '**Save**' to save your settings when you are done.
   Please Note:
   - The JIRA Configuration Tool will save your database configuration to a `dbconfig.xml` file in your JIRA Home Directory. This tool also adds the element `<validation-query>select 1</validation-query>` to this file, which is usually required when running JIRA with default MySQL installations. See Surviving Connection Closures for more information.
   - JIRA must be restarted for your new settings to take effect.

Congratulations — you have finished! Proceed to 'Next Steps' below.

4.2 Connecting JIRA to MySQL Manually

1. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.
   If this file does not exist, create the file, copy and paste the example XML code below into this file and edit the pasted XML as required.
   When editing your `dbconfig.xml` file, escape the '&' characters by adding 'amp;' to the end of each one, as shown in the `<url/>`element below:

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>mysql</database-type>
  <schema-name></schema-name>
  <jdbc-datasource>

<url>jdbc:mysql://dbserver:3306/jiradb?useUnicode=true&amp;characterEncoding=utf8&amp;sessior
<driver-class>com.mysql.jdbc.Driver</driver-class>
    <username>jiradbuser</username>
    <password>[enter db password]</password>
    <pool-size>15</pool-size>
    <validation-query>select 1</validation-query>
  </jdbc-datasource>
</jira-database-config>
```

Please Note:

- The database URL in the example `dbconfig.xml` file content above assumes a UTF-8 database — i.e. that your database was created using a command similar to `create database jiradb character set utf8;`
- If you do not specify `character set utf8` when creating this database, you risk getting '`Data truncation: Data too long for column`' errors when importing data or corruption of non-supported characters. See storing non-ASCII characters in MySQL for details.
- The `<validation-query>select 1</validation-query>` element is usually required when configuring JIRA with default MySQL installations. See Surviving Connection Closures for more information.
- Adding the `sessionVariables=storage_engine=InnoDB` parameter to the database URL is strongly recommended when configuring JIRA with MySQL, to avoid data corruption. See the Additional Notes section below for more information.

2. Save your edited `dbconfig.xml` file (at the root of your JIRA Home Directory).
   JIRA must be restarted for your new settings to take effect.

Proceed to 'Next Steps' below.

4.3. Additional Notes

The default storage engine used by MySQL Server versions prior to 5.5 is MyISAM. Hence, a JIRA database running on a default configuration of a MySQL Server earlier than version 5.5, could experience table creation problems (JRA-24124), which may result in data corruption in JIRA.

Hence, specifying the `sessionVariables=storage_engine=InnoDB` parameter in your database URL (by following the procedures above) is *strongly recommended*. Doing so ensures that tables written to JIRA's MySQL database will use the InnoDB storage engine, which supports 'database transactions' required by JIRA.

Be aware that JIRA uses the 'READ-COMMITTED' transaction isolation level with MySQL, which currently only supports row-based binary logging. If you require MySQL's binary logging features, you must configure MySQL's binary logging format to be 'row-based'. If not, you may encounter problems when creating issues in JIRA. For more information, please refer to JIRA Cannot Create Issues when Using MySQL with Binary Logging.

5. Next Steps

You should now have JIRA configured to connect to your MySQL database.

- If you are using a 'recommended' distribution of JIRA, start it up and watch the logs for any errors.
- If you are using the JIRA WAR distribution, rebuild and redeploy the webapp in your application server.

Installation Notes

Please see JIRA and MySQL.

Connecting JIRA to Oracle

These instructions will help you connect JIRA to an Oracle 11g database.

Please Note:

- If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure a Oracle database connection for you.
- Oracle 9i is no longer a supported database for use with JIRA and the 11.2.x drivers from Oracle do not support 9i.

1. Before You Begin

1.1 Are You Migrating JIRA to Another Server?

If you are Migrating JIRA to Another Server, create an export of your data as an XML backup. You will then be able to transfer data from you old database to your new database, as described in Switching databases.

1.2 Shut Down JIRA

On this page:

- 1. Before You Begin
  - 1.1 Are You Migrating JIRA to Another Server?
  - 1.2 Shut Down JIRA
- 2. Configure Oracle
- 3. Copy the Oracle JDBC Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your Oracle Database
  - 4.1 Connecting JIRA to Oracle Using the JIRA Configuration Tool
  - 4.2 Connecting JIRA to Oracle Manually
- 5. Next steps
- Installation notes

2. Configure Oracle

1. Ensure that you have a database instance available for JIRA (either create a new one or use an existing one).
2. Within that database instance, create a user which JIRA will connect as (e.g. **`jiradbuser`**). Remember this database user name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   When you create a user in Oracle, Oracle will create a 'schema' automatically.
   When you create a user, the tablespace for the table objects must be specified.

   ```
   create user <user> identified by <user_pass> default tablespace <tablespace_name> quota
   unlimited on <tablespace_name>;
   ```

3. Ensure that the user has the following permissions:

   ```
   grant connect to <user>;
   grant create table to <user>;
   grant create sequence to <user>;
   grant create trigger to <user>;
   ```

3. Copy the Oracle JDBC Driver to Your Application Server (JIRA WAR Only)

Skip this step if you installed a 'Recommended' distribution of JIRA, which (unlike JIRA WAR) includes the Oracle JDBC driver.

1. Download the Oracle JDBC driver (from Oracle's site).
2. Add the appropriate Oracle JDBC driver jar (**`ojdbc5.jar`** for JDK 1.5, **`ojdbc6.jar`** for JDK 1.6) to the `lib/` directory.

Please note that a number of the Oracle JDBC driver versions cannot be used with JIRA or are inherently unstable. The known issues with Oracle drivers are as follows:

- We **recommend** that you use the 11.2.x versionof the driver for all versions of Oracle (it is backwards compatible). Many other versions of the driver have been noted to have problems, such as:
  - Version 10.2.0.3.0 of the 10g Release 2 JDBC driver has been noted to produce occurrences of error ORA-01461. The Oracle Support site has further details on this Oracle server issue, although you will need an Oracle support account to access this site.
  - Version 10.2.0.1.0 of the 10g Release 2 JDBC driver hangs with some databases.
  - The 10g Release 1 JDBC driver (10.1.0.4) does not hang, but throws ArrayIndexOutOfBoundsExceptions.

4. Configure Your JIRA Server to Connect to Your Oracle Database

There are two ways to configure your JIRA server to connect to your Oracle database:

- Using the JIRA Configuration Tool (except JIRA WAR distributions) or
- Manually

Please Note:

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. We recommend using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to configure JIRA with Oracle manually.

4.1 Connecting JIRA to Oracle Using the JIRA Configuration Tool

The JIRA Configuration Tool is not available with JIRA WAR distributions.

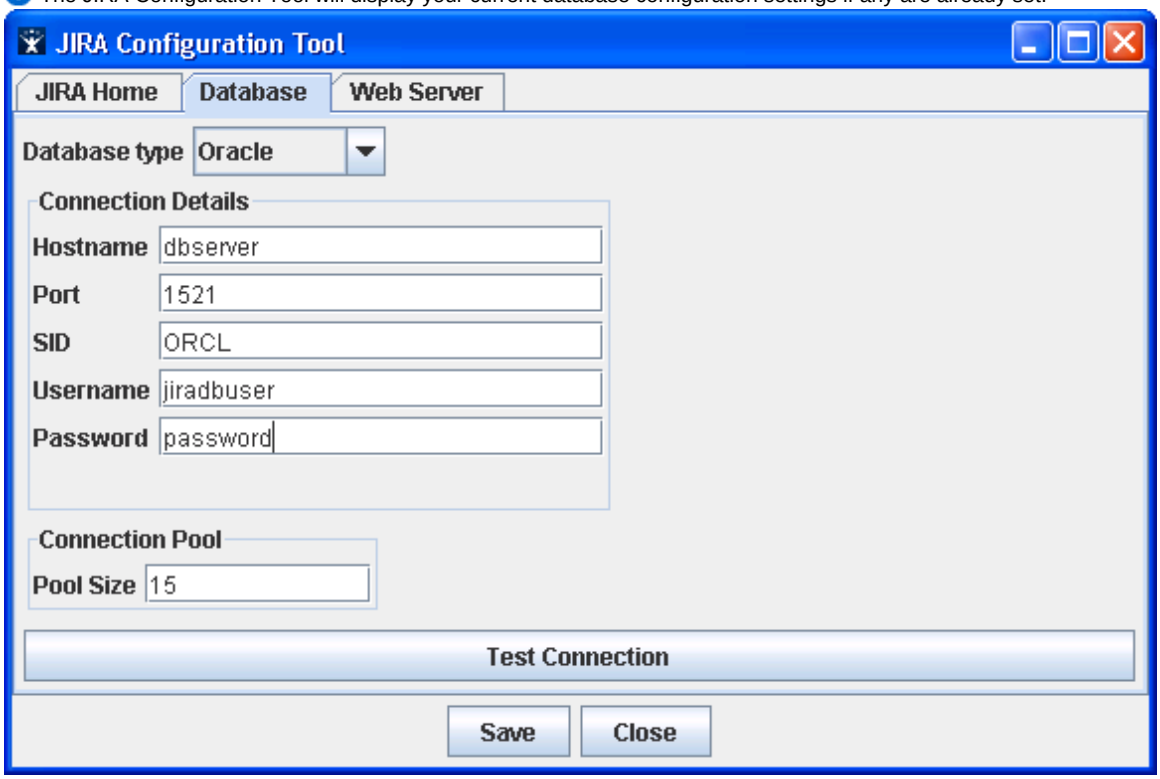
To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

Please Note: You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

To connect JIRA to Oracle using the JIRA Configuration Tool:

1. Click the '**Database**' tab.
2. From the 'Database type' drop-down choose '**Oracle**'.
   The JIRA Configuration Tool will display your current database configuration settings if any are already set.



3. Fill in the connection details for your Oracle database.
   - 'Hostname' — The name or IP address of the machine that the Oracle server is installed on.
   - 'Port' — The TCP/IP port that the Oracle server is listening on. The default port number for Oracle is '1521'.
   - 'SID' — The Oracle 'System Identifier'. The default value for most Oracle servers is 'ORCL'. If you are using the Oracle Express Edition, this will be 'XE'.
   - 'Username' — The user that JIRA uses to connect to the Oracle server.
   - 'Password' — The user's password to authenticate with the Oracle server.
4. JIRA keeps a pool of database connections open to the database server. You can set the maximum size of this pool in the 'Pool Size' text field.
5. After typing in your settings, click the '**Test Connection**' button to test the connection settings. The tool will attempt to connect to the database, and give a message with the results.
6. Click '**Save**' to save your settings when you are done.
   Please Note:
   - The JIRA Configuration Tool will save your database configuration to a `dbconfig.xml` file in your JIRA Home Directory.
   - If you had previously specified any additional custom settings while manually configuring JIRA with Oracle below (for example, adding the `<connection-properties>SetBigStringTryClob=true</connection-properties>` element to your `dbconfig.xml` file), these custom settings will be deleted upon clicking the '**Save**' button and you will need to reinstate them manually.
   - JIRA must be restarted for your new settings to take effect.

Congratulations — you have finished! Proceed to 'Next Steps' below.

4.2 Connecting JIRA to Oracle Manually

1. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.
   If this file does not exist, create the file, copy and paste the example XML code below into this file and edit the pasted XML as required.

   Please Note:

   Ensure that the `<database-type/>` element's content specifies your type of database, as shown below. If you forget to do this and you start JIRA, your database tables may be created incorrectly. Refer to our Incorrect database type specified documentation if this happens to you.

   When editing your `dbconfig.xml` file, escape any '`&`' characters by adding '`amp;`' to the end of each one.

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>oracle10g</database-type>
  <schema-name></schema-name>
  <jdbc-datasource>
    <url>jdbc:oracle:thin:@<hostname>:<port number>:<SID></url>
    <driver-class>oracle.jdbc.OracleDriver</driver-class>
    <username>jiradbuser</username>
    <password>[enter db password]</password>
    <pool-size>15</pool-size>
  </jdbc-datasource>
</jira-database-config>
```

   On a default Oracle server installation, your `<url/>` element's content might look similar to — `jdbc:oracle:thin:@dbserver:1521:ORCL`

2. Save your edited `dbconfig.xml` file (at the root of your JIRA Home Directory).
   JIRA must be restarted for your new settings to take effect.

Please Note:

- If you start experiencing problems when dealing with custom workflows or working with issues that have long descriptions, comments or custom field values, try adding the element `<connection-properties>SetBigStringTryClob=true</connection-properties>` as a child of the `</jdbc-datasource>` element in your `dbconfig.xml` file. Adding this connection property may overcome these problems. Be aware that you will need to restart JIRA for this setting to take effect.

5. Next steps

You should now have JIRA configured to connect to your Oracle database. The next step is to start it up!

- If you are using a 'recommended' distribution of JIRA, start it up and watch the logs for any errors.
- If you are using the JIRA WAR distribution, rebuild and redeploy the webapp in your application server.

Installation notes

Please see JIRA and Oracle.

Connecting JIRA to SQL Server 2005

These instructions will help you connect JIRA to a Microsoft SQL Server 2005 database.

Please Note:

- SQL Server Express is not one of our recommended databases; however, it is possible to set up JIRA to work with this database. This external blog post contains instructions on installing JIRA on SQL Server Express 2005 that may be helpful, if you want to try this.
- Due to numerous reported performance issues with SQL Server 2000, it is recommended that you use SQL Server 2005 instead.

The following instructions apply only to SQL Server 2005, not to SQL Server 2000, which is no longer supported.

If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure a SQL Server 2005 database connection for you.

1. Before You Begin

1.1 Are You Migrating JIRA to Another Server?

If you are Migrating JIRA to Another Server, create an export of your data as an XML backup. You will then be able to transfer data from you old database to your new database, as described in Switching databases.

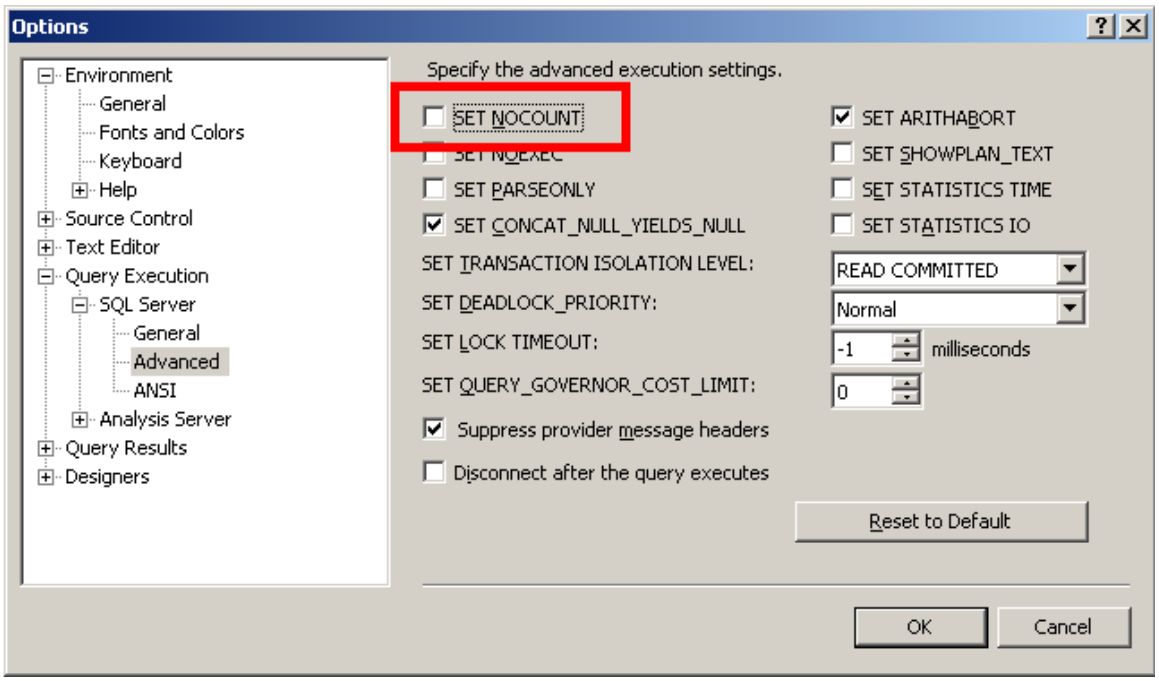
1.2 Shut Down JIRA

On this page:

- 1. Before You Begin
  - 1.1 Are You Migrating JIRA to Another Server?
  - 1.2 Shut Down JIRA
- 2. Create and Configure the SQL Server Database
- 3. Copy the SQL Server JDBC Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your SQL Server 2005 Database
  - 4.1 Connecting JIRA to SQL Server 2005 Using the JIRA Configuration Tool
  - 4.2 Connecting JIRA to SQL Server 2005 Manually
- 5. Next steps
- Installation notes

2. Create and Configure the SQL Server Database

1. Create a database for JIRA to store issues in (e.g. **`jiradb`**). Remember your database name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   Please Note:
   - Collation type must be case-insensitive, for example, 'SQL_Latin1_General_CP437_CI_AI' is case-insensitive collation type. If your SQL Server installation's collation type settings have not been changed from their defaults, check the collation type settings.
   - SQL Server uses Unicode encoding to store characters. This is sufficient to prevent any possible encoding problems.
2. Create a database user which JIRA will connect as (e.g. **`jiradbuser`**). Remember your database user name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   This database user should **not** be the database owner, but **should** be in the `db_owner` role. (See SQL Server Startup Errors for details.)
3. Create an empty 'schema' in the database (e.g. **`jiraschema`**) for the JIRA tables. Remember this database schema name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   A 'schema' in SQL Server 2005 is a distinct namespace used to contain objects and is **different** from a traditional database schema. You are not required to create any of JIRA's tables, fields or relationships (JIRA will create these objects in your empty schema when it starts for the first time). You can read more on SQL Server 2005 schemas in the relevant Microsoft documentation.
4. Ensure that the database user has permission to connect to the database, and create and populate tables in the newly-created schema.
   If you are having difficulties setting up your JIRA database for SQL Server, additional information is available in the Setting Up a SQL Server 2005 database for JIRA document.
5. Ensure that TCP/IP is enabled on SQL Server and listening on the correct port (which is 1433 for a default SQL Server installation).
   Remember this port number, as it will be used to configure JIRA's connection to this database in subsequent steps.
   Read the Microsoft documentation for information on how to enable a network protocol (TCP/IP) and how to configure SQL server to listen on a specific port.
6. Ensure that SQL Server is operating in the appropriate authentication mode. By default, SQL Server operates in 'Windows Authentication Mode'. However, if your user is not associated with a trusted SQL connection, i.e. 'Microsoft SQL Server, Error: 18452' is received during JIRA startup, you will need to change the authentication mode to 'Mixed Authentication Mode'. Read the Microsoft documentation on authentication modes and changing the authentication mode to 'Mixed Authentication Mode'
7. Turn off the SET NOCOUNT option. (The JIRA on MS SQL Server document provides details on the errors that occur if SET NOCOUNT is set.) To turn off SET NOCOUNT:
   - Open **SQL Server Management Studio** and navigate to **Tools -> Options -> Query Execution -> SQL Server -> Advanced**. The following screenshot displays the configuration panel for this setting in MSSQL Server 2005. Ensure that the **SET NOCOUNT** option is **not selected:**



3. Copy the SQL Server JDBC Driver to Your Application Server (JIRA WAR Only)

Skip this step if you installed a 'Recommended' distribution of JIRA, which (unlike JIRA WAR) includes the SQL Server JDBC driver.

1. Download the SQL Server JDBC driver (v1.2.4) from JTDS.
   Microsoft have their own JDBC driver but we **strongly recommend avoiding it** after receiving many reports of intermittent disconnections (JRA-5760 and JRA-6872), workflow problems (JRA-8443) and Chinese character problems (JRA-5054).
2. Add the SQL Server JDBC driver jar (jtds-1.2.4.jar) to the `<Tomcat install>/lib/` directory.

4. Configure Your JIRA Server to Connect to Your SQL Server 2005 Database

There are two ways to configure your JIRA server to connect to your SQL Server 2005 database:

- Using the JIRA Configuration Tool (except JIRA WAR distributions) or
- Manually

Please Note:

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. We recommend using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to configure JIRA with SQL Server 2005 manually.

4.1 Connecting JIRA to SQL Server 2005 Using the JIRA Configuration Tool

The JIRA Configuration Tool is not available with JIRA WAR distributions.

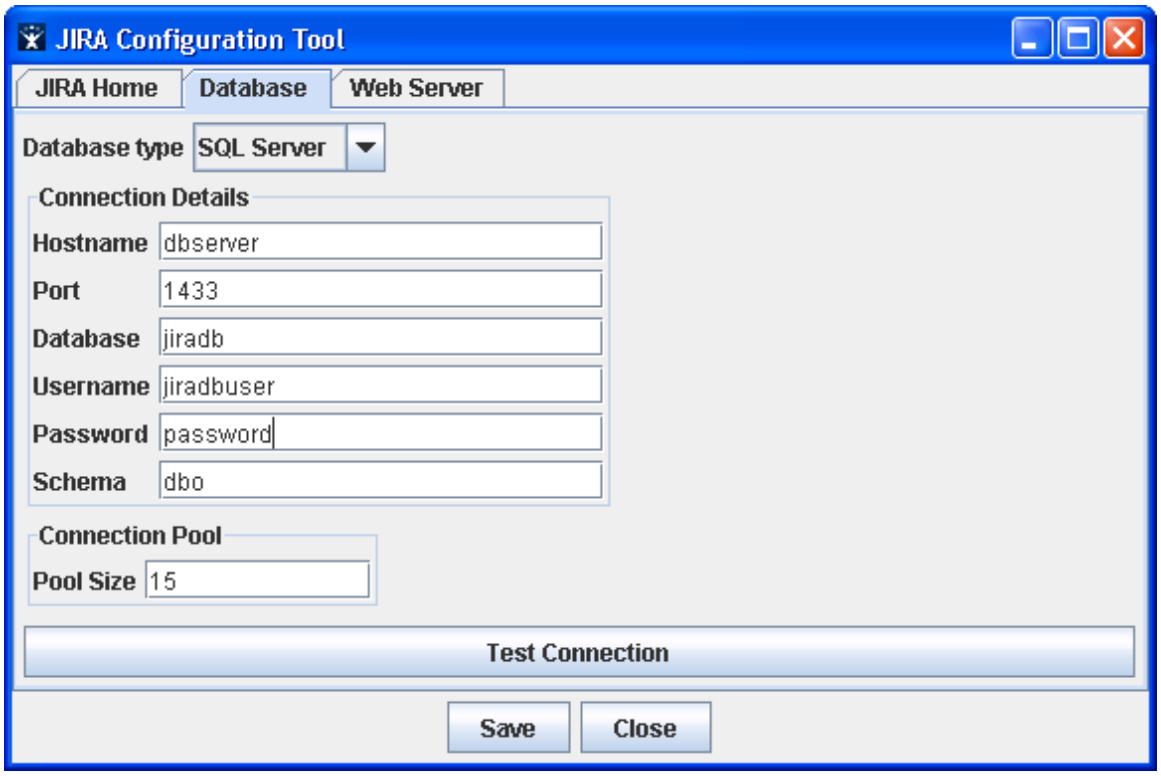
To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

Please Note: You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

To connect JIRA to SQL Server 2005 using the JIRA Configuration Tool:

1. Click the '**Database**' tab.
2. From the 'Database type' drop-down choose '**SQL Server**'.
   The JIRA Configuration Tool will display your current database configuration settings if any are already set.



3. Fill in the connection details for your SQL Server 2005 database.
    - 'Hostname' — The name or IP address of the machine that the SQL Server 2005 server is installed on.
    - 'Port' — The TCP/IP port that the SQL Server 2005 server is listening on. You can leave this blank to use the default port.
    - 'Database' — The name of your SQL Server 2005 database (into which JIRA will save its data).
    - 'Username' — The user that JIRA uses to connect to the SQL Server 2005 server.
    - 'Password' — The user's password to authenticate with the SQL Server 2005 server.
    - 'Schema' — The name of the schema that your SQL Server 2005 database uses.

      You might need to change this from the default 'dbo'.
4. JIRA keeps a pool of database connections open to the database server. You can set the maximum size of this pool in the 'Pool Size' text field.
5. After typing in your settings, click the '**Test Connection**' button to test the connection settings. The tool will attempt to connect to the database, and give a message with the results.
6. Click '**Save**' to save your settings when you are done.

Please Note:

- The JIRA Configuration Tool will save your database configuration to a `dbconfig.xml` file in your JIRA Home Directory.
- JIRA must be restarted for your new settings to take effect.

Congratulations — you have finished! Proceed to 'Next Steps' below.

4.2 Connecting JIRA to SQL Server 2005 Manually

1. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.

   If this file does not exist, create the file, copy and paste the example XML code below into this file and edit the pasted XML as required.

> **Please Note:**
>
> Ensure that the `<database-type/>` element's content specifies your type of database, as shown below. Also ensure that the `<schema-name/>` element's content matches the name of the SQL Server's schema specified when creating your JIRA database (above).

When editing your `dbconfig.xml` file, escape any '`&`' characters by adding '`amp;`' to the end of each one.

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>mssql</database-type>
  <schema-name>jiraschema</schema-name>
  <jdbc-datasource>
    <url>jdbc:jtds:sqlserver://dbserver:1433/jiradb</url>
    <driver-class>net.sourceforge.jtds.jdbc.Driver</driver-class>
    <username>jiradbuser</username>
    <password>[enter db password]</password>
    <pool-size>15</pool-size>
  </jdbc-datasource>
</jira-database-config>
```

2. Save your edited `dbconfig.xml` file (at the root of your JIRA Home Directory).
   JIRA must be restarted for your new settings to take effect.

5. Next steps

You should now have JIRA configured to connect to your SQL Server database. The next step is to start it up!

- If you are using a 'recommended' distribution of JIRA, start it up and watch the logs for any errors.
- If you are using the JIRA WAR distribution, rebuild and redeploy the webapp in your application server.

Installation notes

Please see JIRA and MS SQL Server 2005.

Connecting JIRA to SQL Server 2008

These instructions will help you connect JIRA to a Microsoft SQL Server 2008 database.

If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure a SQL Server 2008 database connection for you.

1. Before You Begin

1.1 Are You Migrating JIRA to Another Server?

If you are Migrating JIRA to Another Server, create an export of your data as an XML backup. You will then be able to transfer data from you old database to your new database, as described in Switching databases.

1.2 Shut Down JIRA

On this page:

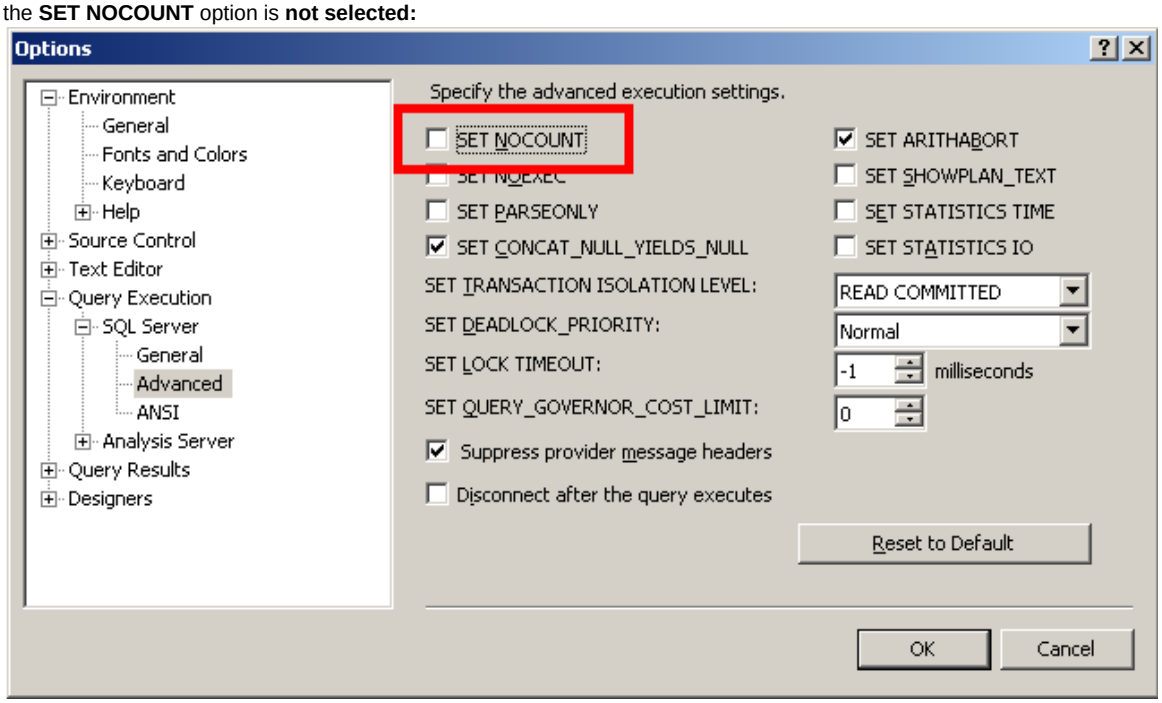
- 1. Before You Begin
  - 1.1 Are You Migrating JIRA to Another Server?
  - 1.2 Shut Down JIRA
- 2. Create and Configure the SQL Server Database
- 3. Copy the SQL Server JDBC Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your SQL Server 2008 Database
  - 4.1 Connecting JIRA to SQL Server 2008 Using the JIRA Configuration Tool
  - 4.2 Connecting JIRA to SQL Server 2008 Manually
- 5. Next steps
- Installation notes

2. Create and Configure the SQL Server Database

1. Create a database for JIRA to store issues in (e.g. **`jiradb`**). Remember this database name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   Please Note:
   - Collation type must be case-insensitive, for example, 'SQL_Latin1_General_CP437_CI_AI' is case-insensitive collation type.

If your SQL Server installation's collation type settings have not been changed from their defaults, check the collation type settings.
- SQL Server uses Unicode encoding to store characters. This is sufficient to prevent any possible encoding problems.

2. Create a database user which JIRA will connect as (e.g. **`jiradbuser`**). Remember this database user name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   This database user should **not** be the database owner, but **should** be in the `db_owner` role. (See SQL Server Startup Errors for details.)
3. Create an empty 'schema' in the database (e.g. **`jiraschema`**) for the JIRA tables. Remember this database schema name, as it will be used to configure JIRA's connection to this database in subsequent steps.
   A 'schema' in SQL Server 2008 is a distinct namespace used to contain objects and is **different** from a traditional database schema. You are not required to create any of JIRA's tables, fields or relationships (JIRA will create these objects in your empty schema when it starts for the first time). You can read more on SQL Server 2008 schemas in the relevant Microsoft documentation.
4. Ensure that the database user has permission to connect to the database, and create and populate tables in the newly-created schema.
5. Ensure that TCP/IP is enabled on SQL Server and listening on the correct port (which is 1433 for a default SQL Server installation).
   Remember this port number, as it will be used to configure JIRA's connection to this database in subsequent steps.
   Read the Microsoft documentation for information on how to enable a network protocol (TCP/IP) and how to configure SQL server to listen on a specific port.
6. Ensure that SQL Server is operating in the appropriate authentication mode. By default, SQL Server operates in 'Windows Authentication Mode'. However, if your user is not associated with a trusted SQL connection, i.e. 'Microsoft SQL Server, Error: 18452' is received during JIRA startup, you will need to change the authentication mode to 'Mixed Authentication Mode'. Read the Microsoft documentation on authentication modes and changing the authentication mode to 'Mixed Authentication Mode'
7. Turn off the SET NOCOUNT option. (The JIRA on MS SQL Server document provides details on the errors that occur if SET NOCOUNT is set.) To turn off SET NOCOUNT:
   - Open **SQL Server Management Studio** and navigate to **Tools -> Options -> Query Execution -> SQL Server -> Advanced**. The following screenshot displays the configuration panel for this setting in MSSQL Server 2008. Ensure that the **SET NOCOUNT** option is **not selected:**



3. Copy the SQL Server JDBC Driver to Your Application Server (JIRA WAR Only)

Skip this step if you installed a 'Recommended' distribution of JIRA, which (unlike JIRA WAR) includes the SQL Server JDBC driver.

1. Download the SQL Server JDBC driver (v1.2.4) from jTDS.
   Microsoft have their own JDBC driver but we have not tested JIRA with it. Previous versions of the MS JDBC driver have been known to cause issues: (JRA-5760 and JRA-6872), workflow problems (JRA-8443) and Chinese character problems (JRA-5054).
2. Add the SQL Server JDBC driver jar (`jtds-1.2.4.jar`) to the `lib/` directory.

4. Configure Your JIRA Server to Connect to Your SQL Server 2008 Database

There are two ways to configure your JIRA server to connect to your SQL Server 2008 database:

- Using the JIRA Configuration Tool (except JIRA WAR distributions) or
- Manually

Please Note:

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. We recommend

using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to configure JIRA with SQL Server 2008 manually.

4.1 Connecting JIRA to SQL Server 2008 Using the JIRA Configuration Tool

The JIRA Configuration Tool is not available with JIRA WAR distributions.

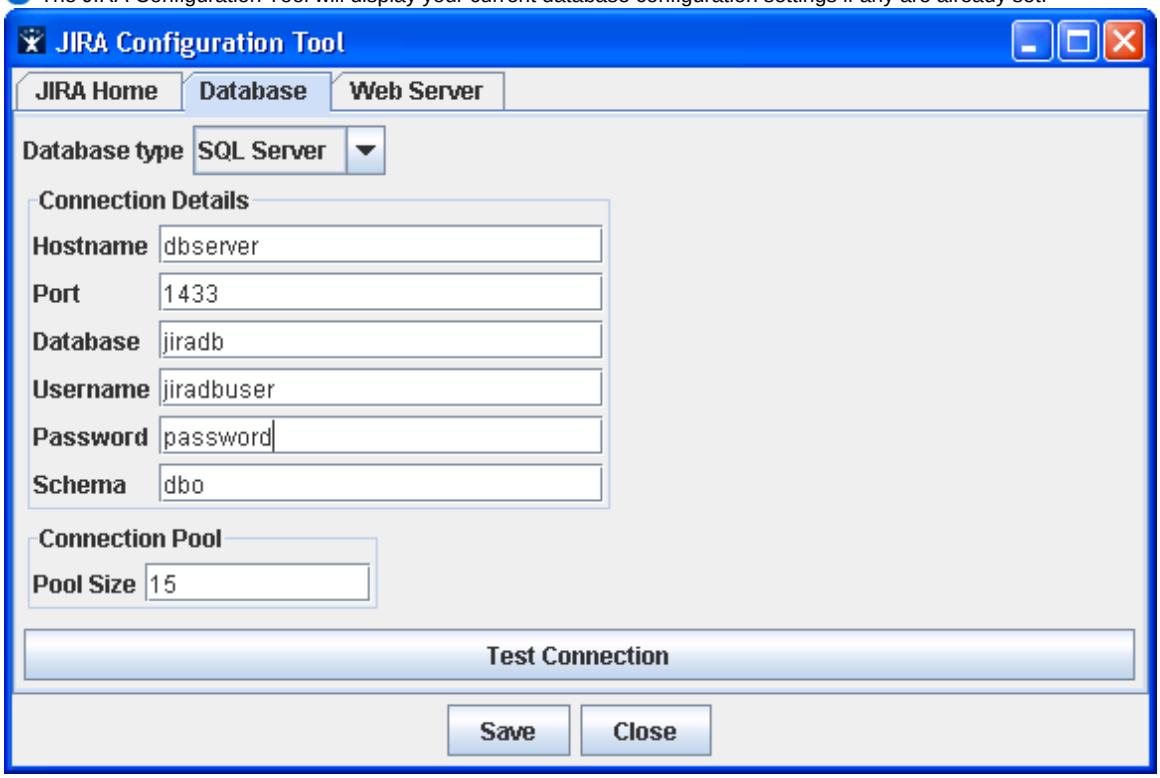
To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

Please Note: You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

To connect JIRA to SQL Server 2008 using the JIRA Configuration Tool:

1. Click the '**Database**' tab.
2. From the 'Database type' drop-down choose '**SQL Server**'.
   The JIRA Configuration Tool will display your current database configuration settings if any are already set.



3. Fill in the connection details for your SQL Server 2008 database.
   - 'Hostname' — The name or IP address of the machine that the SQL Server 2008 server is installed on.
   - 'Port' — The TCP/IP port that the SQL Server 2008 server is listening on. You can leave this blank to use the default port.
   - 'Database' — The name of your SQL Server 2008 database (into which JIRA will save its data).
   - 'Username' — The user that JIRA uses to connect to the SQL Server 2008 server.
   - 'Password' — The user's password to authenticate with the SQL Server 2008 server.
   - 'Schema' — The name of the schema that your SQL Server 2008 database uses.
     You might need to change this from the default 'dbo'.
4. JIRA keeps a pool of database connections open to the database server. You can set the maximum size of this pool in the 'Pool Size' text field.
5. After typing in your settings, click the '**Test Connection**' button to test the connection settings. The tool will attempt to connect to the database, and give a message with the results.
6. Click '**Save**' to save your settings when you are done.
   Please Note:
   - The JIRA Configuration Tool will save your database configuration to a `dbconfig.xml` file in your JIRA Home Directory.
   - JIRA must be restarted for your new settings to take effect.

Congratulations — you have finished! Proceed to 'Next Steps' below.

4.2 Connecting JIRA to SQL Server 2008 Manually

1. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.
   If this file does not exist, create the file, copy and paste the example XML code below into this file and edit the pasted XML as required.

   Please Note:

   Ensure that the `<database-type/>` element's content specifies your type of database, as shown below.
   Also ensure that the `<schema-name/>` element's content matches the name of the SQL Server's schema specified when creating your JIRA database (above).
   If you have a named SQL Server instance, your `<url/>` element's content may look similar to — `jdbc:jtds:sqlserver://dbserver:1433/jiradb;instance=instance_name`
   For details about constructing these URLs please refer to the jTDS JDBC Driver FAQ.

   When editing your `dbconfig.xml` file, escape any '`&`' characters by adding '`amp;`' to the end of each one.

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>mssql</database-type>
  <schema-name>jiraschema</schema-name>
  <jdbc-datasource>
    <url>jdbc:jtds:sqlserver://dbserver:1433/jiradb</url>
    <driver-class>net.sourceforge.jtds.jdbc.Driver</driver-class>
    <username>jiradbuser</username>
    <password>[enter db password]</password>
    <pool-size>15</pool-size>
  </jdbc-datasource>
</jira-database-config>
```

2. Save your edited `dbconfig.xml` file (at the root of your JIRA Home Directory).
   JIRA must be restarted for your new settings to take effect.

5. Next steps

You should now have JIRA configured to connect to your SQL Server database. The next step is to start it up!

- If you are using a 'recommended' distribution of JIRA, start it up and watch the logs for any errors.
- If you are using the JIRA WAR distribution, rebuild and redeploy the webapp in your application server.

Installation notes

Please see JIRA and MS SQL Server 2008.

Connecting JIRA to HSQLDB

Although HSQLDB is bundled with JIRA, we do not recommend it for production use. Please consider using one of the recommended databases instead. See Connecting JIRA to a Database for more information.

If you are setting up a completely new JIRA installation, the JIRA Setup Wizard will configure an HSQL database connection for you, by choosing the **Internal Database Connection** during the first step of the wizard.

1. Before You Begin

Are You Migrating JIRA to Another Server?

If you are Migrating JIRA to Another Server, create an export of your data as an XML backup. You will then be able to transfer data from you old database to your new database, as described in Switching databases.

On this page:

- 1. Before You Begin
  - Are You Migrating JIRA to Another Server?
- 2. Copy the HSQLDB Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your HSQL Database
  - 4.1 Connecting JIRA to HSQLDB Using the JIRA Configuration Tool
  - 4.2 Connecting JIRA to HSQLDB Manually
  - 4.2 Next steps
- Installation notes

2. Copy the HSQLDB Driver to Your Application Server (JIRA WAR Only)

Skip this step if you installed a 'Recommended' distribution of JIRA, which (unlike JIRA WAR) includes the HSQLDB JDBC driver.

1. Download the HSQLDB JDBC driver — hsqldb-1.8.0.5.jar for JIRA 3.7+, or hsqldb-1.7.1-patched.jar for JIRA 3.6.5 and earlier. We **strongly recommend** upgrading to 3.7 if you wish to use hsqldb, as hsqldb 1.7.x is prone to data corruption.
2. Add the HSQLDB JDBC driver jar to the `lib/` directory.

4. Configure Your JIRA Server to Connect to Your HSQL Database

There are two ways to configure your JIRA server to connect to your HSQL database:

- Using the JIRA Configuration Tool (except JIRA WAR distributions) or
- Manually

Please Note:

- The JIRA Configuration Tool requires a Java platform to be installed and configured on your operating system. We recommend using a Java platform supported by JIRA — refer to JIRA Requirements for details.
- If you use the JIRA WAR distribution or have a console-only connection to your JIRA server, you will need to configure JIRA with HSQLDB manually.

4.1 Connecting JIRA to HSQLDB Using the JIRA Configuration Tool

The JIRA Configuration Tool is not available with JIRA WAR distributions.

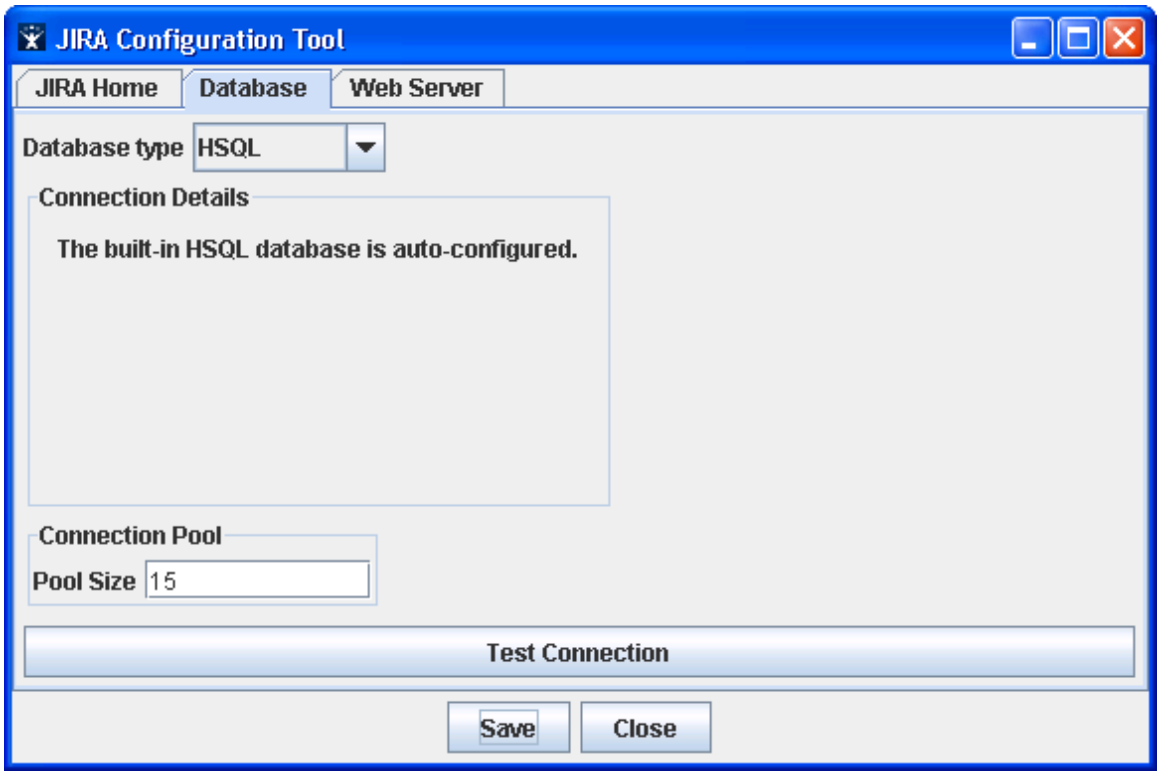
To start the JIRA Configuration Tool:

- **On Windows:**
  Open a command prompt and run `config.bat` in the `bin` subdirectory of the JIRA Installation Directory.
- **On Linux/Unix:**
  Open a console and execute `config.sh` in the `bin` subdirectory of the JIRA Installation Directory.

Please Note: You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See Installing Java for details.

To connect JIRA to HSQLDB using the JIRA Configuration Tool:

1. Click the '**Database**' tab.
2. From the '**Database type**' drop-down, choose '**HSQL**'.



3. JIRA keeps a pool of database connections open to the database server. You can set the maximum size of this pool in the 'Pool Size' text field.
4. Click '**Save**' to save your settings when you are done.

Please Note:

- The JIRA Configuration Tool will save your database configuration to a `dbconfig.xml` file in your JIRA Home Directory. This tool also adds the following elements to this file, which are normally required when running JIRA with HSQLDB:
  `<min-evictable-idle-time-millis>4000</min-evictable-idle-time-millis>`
  `<time-between-eviction-runs-millis>5000</time-between-eviction-runs-millis>`
- JIRA must be restarted for your new settings to take effect.

Congratulations — you have finished! Proceed to 'Next Steps' below.

4.2 Connecting JIRA to HSQLDB Manually

1. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.
   If this file does not exist, create the file, copy and paste the example XML code below into this file and edit the pasted XML as required.

> **Please Note:**
>
> Ensure that the `<database-type/>` element's content specifies your type of database, as shown below. If you forget to do this and you start JIRA, your database tables may be created incorrectly. Refer to our Incorrect database type specified documentation if this happens to you.

When editing your `dbconfig.xml` file, escape any '`&`' characters by adding '`amp;`' to the end of each one.

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>hsql</database-type>
  <schema-name>PUBLIC</schema-name>
  <jdbc-datasource>
    <url>jdbc:hsqldb:/path/to/jira/database/jiradb</url>
    <driver-class>org.hsqldb.jdbcDriver</driver-class>
    <username>sa</username>
    <password></password>
    <pool-size>15</pool-size>
    <min-evictable-idle-time-millis>4000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>5000</time-between-eviction-runs-millis>
  </jdbc-datasource>
</jira-database-config>
```

Please Note:

- The following child elements of the `<jdbc-datasource/>` element are normally required when running JIRA with HSQLDB:
  `<min-evictable-idle-time-millis>4000</min-evictable-idle-time-millis>`
  `<time-between-eviction-runs-millis>5000</time-between-eviction-runs-millis>`

2. Save your edited `dbconfig.xml` file (at the root of your JIRA Home Directory).

JIRA must be restarted for your new settings to take effect.

4.2 Next steps

You should now have JIRA configured to connect to your HSQL database.

- If you are using a 'recommended' distribution of JIRA, start it up and watch the logs for any errors.
- If you are using the JIRA WAR distribution, rebuild and redeploy the webapp in your application server.

Installation notes

Please see JIRA and HSQL.

Switching Databases

JIRA's data can be migrated between different databases on the same database server, to the same database type on a different server (e.g. from one PostgreSQL server to another PostgreSQL server) or to a different type of database server (e.g. from a MySQL server to a PostgreSQL server).

To do this, follow the appropriate procedure:

- Migrating JIRA's Data to the Same Type of Database
- Migrating JIRA's Data to a Different Type of Database Server

Migrating JIRA's Data to the Same Type of Database

Use this procedure to migrate JIRA's data to:

- A different database on the same database server, or
- The same database type on a different database server (e.g. from one PostgreSQL server to another PostgreSQL server).

To migrate JIRA's data to the same type of database:

1. Use your database server's native tools to either:
   - Copy your JIRA database to a new database on the same database server installation, or
   - Copy/migrate your JIRA database to a new database of the same type on a different database server installation.

   Please Note:
   - If you are unable to do either of these tasks, use the Migrating JIRA's Database to a Different Type of Database Server procedure (below) instead.
   - You could use this procedure to migrate JIRA's data to a different type of database server (e.g. MySQL to PostgreSQL). However, you would need to find tools that support these processes. Furthermore, Atlassian does not provide support for this strategy.
2. Once your new database has been populated with JIRA's data, shut down your JIRA server.
3. Make a backup of your JIRA Home Directory and JIRA Installation Directory.
4. Reconfigure your JIRA server's connection to your database:
   - If you installed a 'Recommended' distribution of JIRA, you can use the JIRA Configuration Tool (by running

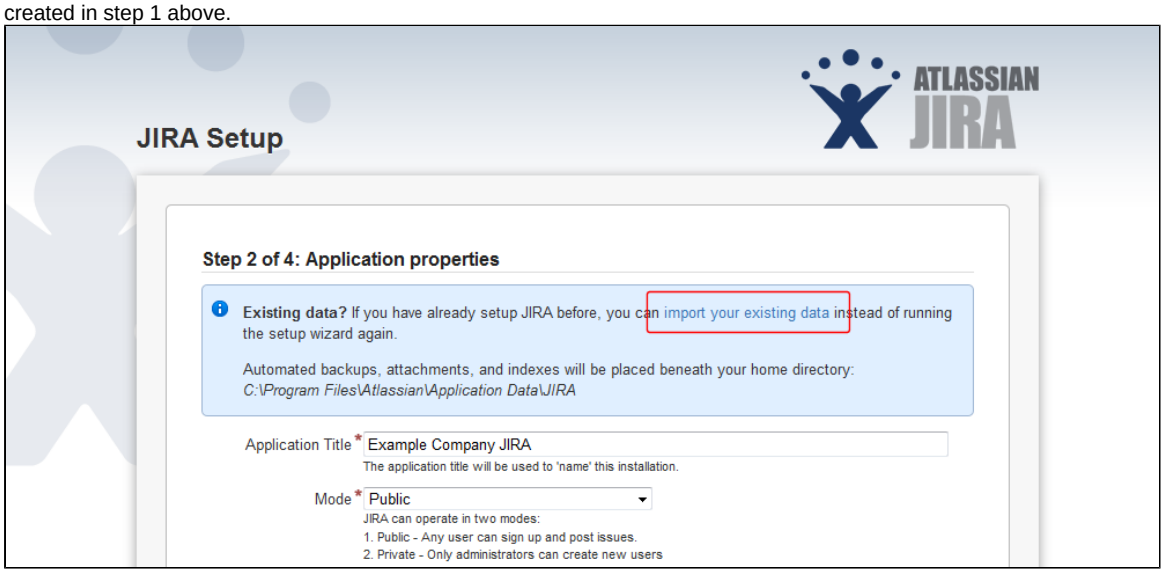
`bin/config.sh` (for Linux/Solaris) or `bin\config.bat` (for Windows) in your JIRA Installation Directory), which provides a convenient GUI that allows you to reconfigure JIRA's database connection settings.

- If any of the following points applies to your situation, you need to manually configure the `dbconfig.xml` file in your JIRA Home Directory. Refer to the appropriate database configuration guide in the Connecting JIRA to a Databasesection for the manual configuration instructions.
  - You are using JIRA WAR
  - You have a console-only connection to your JIRA server
  - You would prefer to configure your database connection manually (for custom configuration purposes).

Migrating JIRA's Data to a Different Type of Database Server

Use this procedure to migrate JIRA's data to a different type of database server (e.g. from a MySQL server to a PostgreSQL server).

1. Create an export of your data as an XML backup. See Backing Up Data for details.
   ⚠ Please note that JIRA's XML backup utility does not back up attachments (if you have attachments enabled).
2. Create a new database on your new database server to house JIRA's data. See the appropriate database configuration guide in the Connecting JIRA to a Database section for the database creation instructions.
3. Shut down your JIRA server.
4. Make a backup of your JIRA Home Directory and JIRA Installation Directory.
5. Delete the `dbconfig.xml` file in your JIRA Home Directory.
6. Restart JIRA and you should see the first step of the JIRA Setup Wizard for configuring your database connection.
7. Configure JIRA's connection to your new database (created in step 2 above) and click the '**Next**' button.
8. On the 'Application Properties' setup page, click the '**import your existing data**' link and restore your data from the XML backup created in step 1 above.



Surviving Connection Closures

When a database server reboots or a network failure has occurred, all connections in the database connection pool are broken. To overcome this issue, JIRA would normally need restarting (or for JIRA WAR distributions, the application server running JIRA would need restarting).

However, the Apache Commons DBCP (DataBase Connection Pool), which is used by JIRA, can validate connections in the connection pool by running a simple SQL query. If a broken connection is detected, a new one is created to replace it.

To do this, you need to specify an optional `<validation-query/>` element (in the `dbconfig.xml` file of your JIRA Home Directory), whose content is the query which validates connections in the connection pool. See the following procedure for details.

Setting the Validation Query for Your JIRA Database

To ensure JIRA validates connections in the connection pool:

1. Shut down JIRA (or the Tomcat installation running JIRA).
2. Edit the `dbconfig.xml` file at the root of your JIRA Home Directory.
3. Add the `<validation-query/>` element with the appropriate validation query for your type of database, as shown in the example below for MySQL. (See Determining the Validation Query below for details.)

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>mysql</database-type>
  <schema-name></schema-name>
  <jdbc-datasource>

<url>jdbc:mysql://dbserver:3306/jiradb?characterEncoding=utf8&amp;useUnicode=true</url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <username>jiradbuser</username>
    <password>mydbpassword</password>
    <pool-size>20</pool-size>

    <validation-query>select 1</validation-query>

  </jdbc-datasource>
</jira-database-config>
```

4. Save your edited `dbconfig.xml` file.
5. Restart JIRA (or the Tomcat installation running JIRA).

Determining the Validation Query

Different database types have slightly different SQL syntax requirements for their validation query. The validation query should be as simple as possible, as this is run every time a connection is retrieved from the pool.

The following validation queries are recommended for the following types of databases:

Database Type	Validation Query
MySQL	`select 1`
Microsoft SQL Server	`select 1`
Oracle	`select 1 from dual`
PostgreSQL	`select version();`

Result

You should now be able to recover from a complete loss of all connections in the database connection pool without the need to restart JIRA or the application server running JIRA.

Performance Considerations:

- Setting this option has a performance impact. The overall decrease in performance should be minimal, as the query itself is quick to run. In addition, the query will only execute when you make a connection. Thus, if the connection is kept for the duration of a request, the query will only occur once per request.
- If you are running a large JIRA installation, you may wish to assess the performance impact of this change before implementing it.

Upgrading JIRA

This document describes the standard, recommended procedure for upgrading to **JIRA 5.0.x** on Linux or Windows.

Use this procedure if:

- You are **upgrading from JIRA version 4.3.0 or later on Linux or Windows (excluding JIRA WAR installations)**.
  AND
- You are **not changing** any of the following:
  - The operating system that will run JIRA.
  - The database that will store JIRA's data.
  - The location of JIRA's index and/or attachments paths.

Otherwise:

- If any of the first three points applies to your situation *AND* the fourth point is true, use the *manual procedure* (see Upgrading JIRA Manually) to upgrade JIRA instead:
  - You are **upgrading from JIRA Standalone versions 4.0.0 – 4.2.x**.
  - You are **upgrading from JIRA WAR version 4.0.0 or later**.
  - You are **upgrading JIRA version 4.3.0 or later on Solaris (excluding JIRA WAR installations)**.
    AND
  - You are **not changing** any of the following:
    - The operating system that will run JIRA.
    - The database that will store JIRA's data.
    - The location of JIRA's index and/or attachments paths.

- If any of these points applies to your situation, use the *migration procedure* (see Migrating JIRA to Another Server) to upgrade JIRA instead:
  - You are **upgrading from a version of JIRA prior to 4.0.0**.
  - You are **changing the location of your index and/or attachments path** for **JIRA 5.0.x**.
  - You are **changing the operating system** that will run **JIRA 5.0.x**.
  - You are **changing the database or database system** that will store JIRA's data.

Upgrading to JIRA 5.0?

If so, please review the JIRA 5.0 Release Notes and JIRA 5.0 Upgrade Notes for important information about this version of JIRA.

On this page:

- 1. Before You Start
- 2. Upgrade Overview
- 3. Performing the Upgrade
  - Upgrading JIRA on Windows
  - Upgrading JIRA on Linux
- Upgrade Check List
  - Back Up Your External Database
  - Check Plugin Compatibility

Please read/perform *all steps and sub-steps* in consecutive order.

1. Before You Start

1. You will need current software maintenance to perform the upgrade. If you are unsure, confirm the following:
   - Your license support period is still valid.
   - If your current license has expired but you have a new license with you, please update your license in JIRA.
     If you forget to do this and your license has expired, you will receive errors during the upgrade process. Refer to the instructions on upgrading beyond current license period.
2. Read the release notes and upgrade guide for the version of JIRA you are upgrading to. The upgrade guide (in particular) contains important information that may be relevant to your JIRA installation.
   - If you plan to skip a few JIRA versions for your next JIRA upgrade, **we strongly recommend** that you read the upgrade guides for all major versions between your current version and the version to which you are upgrading. Refer to Important Version-Specific Upgrade Notes for quick links to these guides.
3. Confirm that your operating system, database, other applicable platforms and hardware still comply with the requirements for **JIRA 5.0.x**. Newer versions of JIRA may have different requirements and supported platforms to previous JIRA versions.
   The End of Support Announcements for JIRA page also has important information regarding platform support for future versions of JIRA.
4. Some anti-virus or other Internet security tools may interfere with the JIRA upgrade process and prevent the process from completing successfully. If you experience or anticipate experiencing such an issue with your anti-virus/Internet security tool, disable this tool first before proceeding with the JIRA upgrade.
5. Check for any known issues in the JIRA Knowledge Base.
6. If you have installed any additional JIRA plugins (i.e. not included with JIRA), please verify that they will be compatible with the version of JIRA you are upgrading to. You can find a plugin's compatibility information from the the plugin's home page on the Atlassian Plugin Exchange. Once you have confirmed the availability of compatible versions, you should upgrade your plugins after successfully upgrading JIRA. This can be done via the '**Plugin Repository**' in your **Administration Console**.
7. **Test first!**— We strongly recommend performing your upgrade in a test environment first. Do not upgrade your production JIRA server until you are satisfied that your test environment upgrade has been successful.
   - If you have any problems with your test environment upgrade which you cannot resolve, create an issue at our support site so that we can assist you.
   - If you have any problems during the upgrade of your production JIRA server, do not allow your users to start using this server. Instead:
     - Continue to use your old JIRA server — this will help ensure that you do not lose production data.
     - Also create an issue at our support site so that we can help you resolve the problems with production JIRA upgrade.

2. Upgrade Overview

The upgrade feature of the Linux and Windows Installers, automates the following tasks for you:

1. Backs up the Installation and Home Directories of the existing JIRA installation to be upgraded.
2. Installs **JIRA 5.0.x** whilst migrating the following from your existing JIRA installation to the new **JIRA 5.0.x**installation:
    - Legacy database configurations defined as a datasource within the application server (used in JIRA 4.3.x and earlier) to the new database configuration used in JIRA 4.4 and later. See JIRA 4.4 Upgrade Notes for details.
    - TCP port values in your existing JIRA installation's `server.xml` file. Be aware that other configurations or customisations in this file are not migrated.
    - Custom values in your existing JIRA installation's `jira-application.properties` and `setenv.sh` / `setenv.bat` files.
      Be aware that in the `setenv.sh` / `setenv.bat`file, only the following values are migrated:
        - `JVM_SUPPORT_RECOMMENDED_ARGS`
        - `JVM_MINIMUM_MEMORY`
        - `JVM_MAXIMUM_MEMORY`
        - `JIRA_MAX_PERM_SIZE`

The upgrade feature detects and notifies you of any files (other than `jira-application.properties` and `setenv.sh` / `setenv.bat`) in the `atlassian-jira` subdirectory of your existing JIRA Installation Directory, which had been deleted, added or modified from a 'default' JIRA installation. This informs you of any customisations you will need to migrate manually over to your upgraded JIRA installation directory.

The upgrade feature also re-uses your existing JIRA Home Directory so that any key data stored in this directory from your previous JIRA installation will be retained after the JIRA upgrade.

Please Note:

- The upgrade process requests that you conduct a backup of your database using your database's backup utilities. If your database does not support online backups, you can stop the upgrade process, shut down JIRA, perform your database backup and then restart the upgrade process to continue on.
- If you have made customisations to your `seraph-config.xml` file or any other file customisations in your JIRA installation directory which are not handled by the upgrade wizard, these must be migrated manually.
- If your attachments and index files are located outside your JIRA Home Directory, then backups of these directories must be performed manually.

3. Performing the Upgrade

Refer to the appropriate upgrade instructions below for your operating system:

- Windows
- Linux

Upgrading JIRA on Windows

1. Download the **JIRA 'Windows Installer'** (.exe) file (for the new version of JIRA) from the JIRA Download page.
2. Run the '.exe' file to start the upgrade wizard.
   If a Windows 7 (or Vista) 'User Account Control' dialog box requests if you want to allow the upgrade wizard to make changes to your computer, specify '**Yes**'. If you do not, the installation wizard will have restricted access to your operating system and any subsequent installation options will be limited.
3. At the 'Upgrading JIRA?' step, choose the '**Upgrade an existing JIRA installation**' option.
4. In the 'Existing JIRA installation directory' field, specify the JIRA Installation Directory of your JIRA installation to be upgraded.
   The upgrade wizard will attempt to find an existing JIRA installation and use its location to pre-populate this field. However, always verify this location, particularly if you have multiple JIRA installations running on the same machine.
5. During subsequent steps of the upgrade wizard, you will be prompted to specify or do the following options:
    a. At the 'Back up JIRA directories' step, ensure the '**Back up these directories**' option is selected. This creates 'zip' archive file backups of your existing JIRA Installation and JIRA Home Directories in their respective parent directory locations.
       Please Note:
        - **Choosing this option is strongly recommended!**
        - At this point, the upgrade wizard notes any customisations in your existing JIRA Installation Directory which it cannot automatically migrate to your upgraded JIRA installation. If you are informed of any files containing such customisations, please make a note of these files as you will need to manually migrate their customisations (which are not mentioned in the overview above) to your upgraded JIRA installation. One relatively common customisation that the upgrade wizard cannot automatically migrate is an SSL configuration defined in the `conf/server.xml` file of the JIRA Installation Directory.
    b. At the 'Upgrade Check List' step, back up your external database and check that any non-bundled plugins will be compatible with your upgraded JIRA version. You may have already conducted the latter (in step 5 of the Before You Start section above).
    c. Upon clicking '**Next**', your existing JIRA installation will be shut down if it is still running. The upgrade wizard will then:
        i. Back up your existing JIRA installation.
        ii. Delete the contents of the existing JIRA Installation Directory.
        iii. Install the new version of JIRA to the existing JIRA Installation Directory.
        iv. Starts your new (upgraded) JIRA installation.
            If you noted any files that contain customisations which must be migrated manually to your upgraded JIRA

installation (above), then:
1. Stop the upgraded JIRA installation.
2. Migrate the customisations from these files into the upgraded JIRA Installation Directory.
3. Restart the upgraded JIRA installation.

6. At the last step of the upgrade wizard, select the option to launch the upgraded JIRA installation in a browser so you can check the upgrade.

Congratulations, you have completed upgrading your JIRA installation on Windows!

Upgrading JIRA on Linux

1. Download the appropriate **JIRA 'Linux 64-bit / 32-bit Installer'** (.bin) file that suits your operating system (for the new version of JIRA) from the JIRA Download page.
2. Open a Linux console and change directory (cd) to the '.bin' file's directory.
   If the '.bin' file is not executable after downloading it, make it executable, for example:
   `chmod a+x atlassian-jira-X.Y.bin`
   (where X.Y represents your version of JIRA)
3. Execute the '.bin' file to start the upgrade wizard.
4. When prompted to choose between creating a new JIRA installation or upgrading an existing installation, choose the '**Upgrade an existing JIRA installation**' option.
5. Specify the JIRA Installation Directory of your JIRA installation to be upgraded.
   The upgrade wizard will attempt to find an existing JIRA installation and will provide its location as a choice. However, always verify this location, particularly if you have multiple JIRA installations running on the same machine.
6. During subsequent steps of the upgrade wizard, you will be prompted to specify or do the following options:
   a. Choose the option to back up JIRA's directories. This creates 'zip' archive file backups of your existing JIRA Installation and JIRA Home Directories in their respective parent directory locations.
      Please Note:
      - **Choosing this option is strongly recommended!**
      - At this point, the upgrade wizard notes any customisations in your existing JIRA Installation Directory which it cannot automatically migrate to your upgraded JIRA installation. If you are informed of any files containing such customisations, please make a note of these files as you will need to manually migrate their customisations (which are not mentioned in the overview above) to your upgraded JIRA installation. One relatively common customisation that the upgrade wizard cannot automatically migrate is an SSL configuration defined in the `conf/server.xml` file of the JIRA Installation Directory.

   b. At the 'Upgrade Check List' step, back up your external database and check that any non-bundled plugins will be compatible with your upgraded JIRA version. You may have already conducted the latter (in step 5 of the Before You Start section above).
   c. Upon proceeding, your existing JIRA installation will be shut down if it is still running. The upgrade wizard will then:
      i. Back up your existing JIRA installation.
      ii. Delete the contents of the existing JIRA installation directory.
      iii. Install the new version of JIRA to the existing JIRA installation directory.
      iv. Starts your new (upgraded) JIRA installation.
      If you noted any files that contain customisations which must be migrated manually to your upgraded JIRA installation (above), then:
      1. Stop the upgraded JIRA installation.
      2. Migrate the customisations from these files into the upgraded JIRA Installation Directory.
      3. Restart the upgraded JIRA installation.

7. The last step of the upgrade wizard provides you with a link to launch the upgraded JIRA installation in a browser, so you can check the upgrade.

Congratulations, you have completed upgrading your JIRA installation on Linux!

Upgrade Check List

The upgrade wizard requests that you perform the following tasks before it actually commences the upgrade of your existing JIRA installation.

Back Up Your External Database

Perform a backup of your external database (using your database's native backup tools) and verify that the backup was created correctly.

- If your database's native backup tools support 'online backups' (i.e. which would typically create a 'snapshot' of your JIRA database while the database is still in use), you can leave the upgrade wizard running while you perform the database backup and then continue on with the wizard after verifying that the database backup was created correctly.
- If your database's native backup tools do not allow you to perform an 'online backup' of your JIRA database, you should:
  1. Quit the upgrade wizard now.
  2. Prevent users from updating your existing JIRA data (to ensure structural consistency of your database backup) by temporarily restricting access to JIRA. Refer to the Preventing users from accessing JIRA during backups page for more information. (To access this page in the documentation for another version of JIRA, click the documentation link for your version of JIRA at the top of the left Table of Contents column and use the search box below to find this page.)
  3. Use your database's native backup tools to perform an 'offline backup' of your JIRA database and verify that this backup was created correctly.
  4. Re-run the Linux / Windows Installer to start the upgrade wizard again and continue from where you left off.
- JIRA's 'internal' database is HSQLDB, which should be used for evaluating JIRA only. If you happen to accidentally use the HSQLDB database for a production system, quit the upgrade wizard now and use the Migrating JIRA to Another Server procedure to

upgrade JIRA.

Inconsistent database backups may not restore correctly! If you are unfamiliar with your database's native backup/restore facilities, then before proceeding, test your database backup's integrity by:

- restoring the database backup to a different (test) system and
- connecting a test instance of your current JIRA version to this restored database.

Alternatively, use the Migrating JIRA to Another Server procedure to upgrade JIRA instead.

Check Plugin Compatibility

As you may have already done in point 5 of the Before You Start section above, if you have installed any additional JIRA plugins (i.e. not included with JIRA), please verify that they will be compatible with the version of JIRA you are upgrading to. You can find a plugin's compatibility information from the the plugin's home page on the Atlassian Plugin Exchange. Once you have confirmed the availability of compatible versions, you should upgrade your plugins after successfully upgrading JIRA. This can be done via the '**Plugin Repository**' in JIRA's **Administration Area**.

Upgrading JIRA Manually

This document describes how to upgrade to **JIRA 5.0.x** *manually*.

Use this procedure if:

- You are **upgrading from**:
  JIRA Standalone versions 4.0.0 – 4.2.x on Linux or Windows.
  or
  JIRA WAR version 4.0.0 and later.
  or
  JIRA version 4.3.0 or later on Solaris (excluding JIRA WAR installations).
  AND
- You are **not changing**any of the following:
  - The operating system that will run JIRA.
  - The database that will store JIRA's data.
  - The location of JIRA's index and/or attachments paths.

Otherwise, if any of the following points applies to your situation, use the *migration procedure* (see Migrating JIRA to Another Server) t upgrade JIRA instead:

- You are **upgrading from a version of JIRA prior to 4.0.0**.
- You are **changing the location of your index and/or attachments path** for **JIRA 5.0.x**.
- You are **changing the operating system** that will run **JIRA 5.0.x**.
- You are **changing the database or database system** that will store JIRA's data.

If you *are already using* JIRA version 4.3.0 or later on Linux or Windows (excluding JIRA WAR installations), please also note the *recommended procedure* (see Upgrading JIRA) for upgrading to **JIRA 5.0.x**.

Upgrading to JIRA 5.0?

If so, please review the JIRA 5.0 Release Notes and JIRA 5.0 Upgrade Notes for important information about this version of JIRA.

On this page:

- 1. Before You Start
- 2. Backing Up
  - 2.1 Back up your database
  - 2.2 Back up your JIRA Home directory
  - 2.3 Back up your attachments directory if located outside your JIRA Home directory
  - 2.4 Back up your JIRA Installation directory
- 3. Performing the Upgrade
  - 3.1 Install the new version of JIRA
  - 3.2 Migrate your existing JIRA configurations over to your new JIRA installation
  - 3.3 Point your new JIRA to your existing JIRA Home directory
  - 3.4 Connect your new JIRA to your existing database
  - 3.5 Start your new version of JIRA
- 4. Post Upgrade Checks and Tasks

Please read/perform *all steps and sub-steps* in consecutive order.

1. Before You Start

1. You will need current software maintenance to perform the upgrade. If you are unsure, confirm the following:
    - Your license support period is still valid.
    - If your current license has expired but you have a new license with you, please update your license in JIRA.

      ⚠ If you forget to do this and your license has expired, you will receive errors during the upgrade process. Refer to the instructions on upgrading beyond current license period.
2. Read the release notes and upgrade guide for the version of JIRA you are upgrading to. The upgrade guide (in particular) contains important information that may be relevant to your JIRA installation.
    - If you plan to skip a few JIRA versions for your next JIRA upgrade, **we strongly recommend** that you read the upgrade guides for all major versions between your current version and the version to which you are upgrading. Refer to Important Version-Specific Upgrade Notes for quick links to these guides.
3. Confirm that your operating system, database, other applicable platforms and hardware still comply with the requirements for **JIRA 5.0.x**. Newer versions of JIRA may have different requirements and supported platforms to previous JIRA versions.

   ✅ The End of Support Announcements for JIRA page also has important information regarding platform support for future versions of JIRA.
4. Some anti-virus or other Internet security tools may interfere with the JIRA upgrade process and prevent the process from completing successfully. If you experience or anticipate experiencing such an issue with your anti-virus/Internet security tool, disable this tool first before proceeding with the JIRA upgrade.
5. Check for any known issues in the JIRA Knowledge Base.
6. If you have installed any additional JIRA plugins (i.e. not included with JIRA), please verify that they will be compatible with the version of JIRA you are upgrading to. You can find a plugin's compatibility information from the the plugin's home page on the Atlassian Plugin Exchange. Once you have confirmed the availability of compatible versions, you should upgrade your plugins after successfully upgrading JIRA. This can be done via the '**Plugin Repository**' in your **Administration Console**.
7. **Test first!**— We strongly recommend performing your upgrade in a test environment first. Do not upgrade your production JIRA server until you are satisfied that your test environment upgrade has been successful.
    - If you have any problems with your test environment upgrade which you cannot resolve, create an issue at our support site so that we can assist you.
    - If you have any problems during the upgrade of your production JIRA server, do not allow your users to start using this server. Instead:
        - Continue to use your old JIRA server — this will help ensure that you do not lose production data.
        - Also create an issue at our support site so that we can help you resolve the problems with production JIRA upgrade.

2. Backing Up

⚠ Before you begin the JIRA upgrade, we ***strongly recommend*** that you back up your existing JIRA installation ***because you cannot roll back a JIRA upgrade (after step 3.3)***.

2.1 Back up your database

Perform a backup of your external database (using your database's native backup tools) and verify that the backup was created correctly.

- If your database's native backup tools support 'online backups' (i.e. which would typically create a 'snapshot' of your JIRA database while the database is still in use), you can leave JIRA running while you perform the database backup.
- If your database's native backup tools do not allow you to perform an 'online backup' of your JIRA database, you should:
    - Prevent users from updating your existing JIRA data (to ensure structural consistency of your database backup) by temporarily restricting access to JIRA. Refer to the Preventing users from accessing JIRA during backups page for more information. (To access this page in the documentation for another version of JIRA, click the documentation link for your version of JIRA at the top of the left Table of Contents column and use the search box below to find this page.)
    - Use your database's native backup tools to perform an 'offline backup' of your JIRA database and verify that this backup was created correctly.
- JIRA's 'internal' database is HSQLDB, which should be used for evaluating JIRA only. If you happen to accidentally use the HSQLDB database for a production system, use the Migrating JIRA to Another Server procedure to upgrade JIRA instead of this one.

⚠ **Inconsistent database backups may not restore correctly!** If you are unfamiliar with your database's native backup/restore facilities, then before proceeding, test your database backup's integrity by:

- restoring the database backup to a different (test) system and
- connecting a test instance of your current JIRA version to this restored database.

Alternatively, use the Migrating JIRA to Another Server procedure to upgrade JIRA instead.

2.2 Back up your JIRA Home directory

Ensure JIRA is shut down before continuing.

The location of this directory is defined within the `jira-application.properties` configuration file, which is located inside the `<jira-application-dir>/WEB-INF/classes` directory within your JIRA Installation Directory.

2.3 Back up your attachments directory if located outside your JIRA Home directory

Your attachments directory may be located outside your JIRA Home Directory. If so, it will also need to be backed up. To confirm the location of your attachments directory, refer to Configuring File Attachments page in the documentation for your version of JIRA. (To do this, click the

documentation link for your version of JIRA at the top of the left Table of Contents column and use the search box below to find the 'Configuring File Attachments' page.)

2.4 Back up your JIRA Installation directory

The 'JIRA Installation Directory' is the directory into which the JIRA application files and libraries were extracted when JIRA was installed.

3. Performing the Upgrade

If you are running a 'mission-critical' JIRA server, we highly recommend performing the remaining steps of this guide in a test environment (e.g. using a copy of your JIRA database and JIRA Home Directory) before performing the upgrade for production use.

3.1 Install the new version of JIRA

First, you must start with a fresh installation of your new JIRA version.

Download and extract the JIRA distribution you require, to a new directory. **Do not overwrite your existing JIRA installation. Simply shut this down and install the new JIRA version to a new location.**

Follow the installation instructions for either:

- Installing JIRA (recommended), or
- Installing JIRA WAR

If you are using JIRA WAR, remember to build your new JIRA web application and deploy it to your server. For specific instructions, read the relevant guide (specific to your application server) within the Installing JIRA WAR section.

3.2 Migrate your existing JIRA configurations over to your new JIRA installation

You may have modified a number of properties within configuration files of your existing JIRA installation.

If so, you need to **make the same modifications in your new JIRA installation**. However, do not simply copy configuration files from your existing JIRA installation and replace the equivalent files in your new JIRA installation, since the properties in these files may have changed from the old version.

For each file you have modified in your existing JIRA installation, you need to **manually edit each equivalent file in your new JIRA installation and re-apply your modifications**. If a file is not present in your new JIRA installation (for example, `osuser.xml` in recent JIRA versions), then simply copy that file over to your new JIRA installation.

The table below lists the most commonly modified files and their locations within your JIRA Installation Directory:

File	Location in 'recommended' (formerly 'Standalone') JIRA distributions	Location in JIRA WAR	Description

jira-application.properties	`atlassian-jira/WEB-INF/classes`	`webapp/WEB-INF/classes`	Location of the JIRA Home Directory and Advanced JIRA Configuration in JIRA 4.3.x and earlier. Any custom property values defi in the `jira-application.propert` file of your existing JIRA 4.3.x (c earlier) installation must be migr across to the `jira-application.propert` file of your new **JIRA 5.0.x** installation *before you start your new JIRA installation*. Upon starting your new JIRA installation, any custom property values in the `jira-application.propert` file will automatically be migrate across to either the JIRA databa or jira-config.properties file. `jira.home` is the only property the `jira-application.propert` file subsequently used by JIRA.
setenv.bat (Windows) or setenv.sh (Linux)	`bin`	Application Server's `bin` directory	Increasing JIRA Memory
osuser.xml (not required if upgrading from JIRA 4.3.0 or later)	`atlassian-jira/WEB-INF/classes`	`webapp/WEB-INF/classes`	Modified if you have integrated LDAP with JIRA, integrated Cro with JIRA, or if you are using a custom form of external user management or user authentica
seraph-config.xml	`atlassian-jira/WEB-INF/classes`	`webapp/WEB-INF/classes`	Modified if you have integrated Crowd with JIRA.
server.xml	`conf`	Application Server's `conf` directory	Modified in the following situatio • If you had previously configured JIRA's TCP ports differently from th defaults. • If you had implemented SSL. • When connecting JIRA database in JIRA 4.3.x earlier.

✅ The version-specific upgrade notes contain details on properties which may have changed in these commonly modified files.

In addition to the files above, you should also consider and/or perform the following configurations as part of the upgrade process:

- **Using JIRA with Atlassian's Crowd?** — If you are using Crowd with JIRA, configure your new JIRA to talk to Crowd as described in Integrating Crowd with JIRA.
- **Allocating additional memory to JIRA** — If you had previously allocated additional memory to JIRA, do the same for your new JIRA instance. For more information refer to Increasing JIRA memory.
- **Plugins**— For any plugins that you had installed in your old JIRA:
  1. Download the plugin version for your new version of JIRA from the http://plugins.atlassian.com site.
  2. Install the JAR file(s) in your new JIRA, and carry out any other required installation for the plugin.
  3. If the plugin has a properties file, apply the same changes to it as you had in the old properties file (don't just copy over the old properties file).
- **Character encoding** — Please ensure that character encoding (ie. locale) is the same on the new and old locations. You may have problems with encoding of the file names, if attachments are moved between two system with incompatible encoding.
  ⚠️ Your new version of JIRA may not function correctly or could encounter problems or errors if these are not implemented.
- **Customisations** — If you had made any customisations (code, templates or configuration files), copy over compatible versions of these changes to the new JIRA. (The developers within your organisation who made the customisations to your old version will need to build and test equivalent changes for the new version, and provide you with the files to copy to your upgraded JIRA installation).
- **(Optional) Disabling Email Access** — If you need to perform some initial tests on your new JIRA installation, you can disable its email access to prevent unintended emails being sent during testing. Be sure to re-enable email access once any testing is complete.

- **(Optional) Running JIRA on a different port** — If your new JIRA is installed on the same machine as your old JIRA, you may wish to make sure it runs on a different port (in case you ever need to restart your old JIRA). See Changing JIRA's TCP Ports for details.

3.3 Point your new JIRA to your existing JIRA Home directory

If your new JIRA 5.0.x installation is a 'recommended' (not WAR) distribution, you can:

1. Open the JIRA Configuration Tool.
2. Click the '**JIRA Home**' tab.
3. Update the 'JIRA Home Directory' field to the path of your **existing** JIRA Home Directory, which you backed up (above).
   For more information about this directory, see JIRA Home Directory.

Otherwise:

1. Edit the `jira-application.properties` file located within the `<jira-application-dir>/WEB-INF/classes` subdirectory of your new **JIRA 5.0.x** Installation Directory JIRA Installation Directory.
2. Update the `jira.home` property in this file to the path of your **existing** JIRA Home Directory, which you backed up (above).
   For more information about this directory, see JIRA Home Directory.
3. Remove the '#' at the beginning of this line (so that JIRA no longer regards this line as a comment).
4. Save your updated `jira-application.properties` file.

You can also set your JIRA Home Directory's location by defining an operating system environment variable `JIRA_HOME`. This value of this variable takes precendence over the value of the `jira.home` property in the `jira-application.properties` file in your JIRA Installation Directory. See Setting your JIRA Home Directory for details.

3.4 Connect your new JIRA to your existing database

Now you need to configure your new JIRA installation to connect to and use your existing database, which was backed up (above).

Once you have performed this step and start your new JIRA server (in step 3.5 below), your JIRA database will be upgraded and it will no longer be compatible with earlier versions of JIRA.

Configure your new JIRA installation to use your existing database. For details refer to the appropriate instructions for your database:

- Configuring JIRA with PostgreSQL
- Configuring JIRA with MySQL
- Configuring JIRA with Oracle
- Configuring JIRA with SQL Server 2005
- Configuring JIRA with SQL Server 2008
- Configuring JIRA with HSQLDB
  The HSQL database is supported for evaluation purposes only.

If your new **JIRA 5.0.x** installation is a JIRA WAR distribution and its using a new application server installation, then ensure your database's JDBC driver has been copied across to your new application server. For details, refer to the appropriate instructions:

- Copying the PostgreSQL JDBC Driver
- Copying the MySQL JDBC Driver
- Copying the Oracle JDBC Driver
- Copying the SQL Server 2005 JDBC Driver
- Copying the SQL Server 2008 JDBC Driver
- Copying the HSQLDB JDBC Driver
  The HSQL database is supported for evaluation purposes only.

3.5 Start your new version of JIRA

1. Verify that your old JIRA installation is shut down — if this JIRA server is still operating, shut it down.
2. If you installed the JIRA WAR distribution within **Tomcat**, **delete the Tomcat work directory before restarting JIRA**. If you do not do this, users may encounter errors when they try to display JIRA pages.
3. Start up your new version of JIRA. For:
   - **'Recommended' distributions** — follow the Starting JIRA instructions.
   - **WAR distributions** — follow the instructions for starting JIRA for your application server within the Installing JIRA WAR section.
     During the startup process, your new JIRA installation will create any required database indexes.
4. Visit JIRA in your web browser and log in using a username from your previous JIRA installation. You should be able to log in immediately, without seeing the Setup Wizard.
5. Take a quick look around your JIRA site to confirm that your projects and issues are present and everything looks normal. You should see the new JIRA version number in the page footer.

Do not restart your old JIRA installation!

Your old JIRA installation may likely still be configured to use the same JIRA Home directory and database as your new JIRA installation. Running two separate JIRA installations which share a common JIRA Home directory and database can result in serious data corruption!

Nevertheless, we recommend that you do not delete any aspect (or backed up component) of your old JIRA installation, until you are satisfied that your upgraded JIRA installation is functioning as expected.

4. Post Upgrade Checks and Tasks

It is strongly recommended that you perform the following checks and tasks after you have started your new instance of JIRA:

1. Check your server logs for error messages, even if JIRA appears to be running correctly. If there are any errors there that you cannot resolve, create a support case in https://support.atlassian.com, attach your log file and we will advise you on the errors.
2. If you were previously using External User Management, enable it in the new JIRA instance.
3. If you changed machines when upgrading, change the paths to the indexes, attachments and backup directories, from within the **Administration** section of JIRA.
4. Enable email, if you disabled it during testing.
5. If you migrated any customisations from your old JIRA to the new JIRA, ensure that they are tested thoroughly.
    a. If you had downloaded plugins for the new version of JIRA, install the downloaded JAR file(s) in your new JIRA version and carry out any other required installation for the plugin.
    b. If the plugin has a properties file, apply the same changes to it as you had in the old properties file (don't just copy over the old properties file).

Congratulations! You have completed your JIRA upgrade.

See Also

Disabling Auto-Export

Migrating JIRA to Another Server

This document describes how to perform any of the following:

- migrate/upgrade to **JIRA 5.0.x**on different server hardware or in a different server environment — for example:
    - a new operating system that will run JIRA,
    - new locations for storing your index and/or attachments, or
    - a new database or database system that will store JIRA's data.
- upgrade to **JIRA 5.0.x** from a version of JIRA prior to 4.0.0.

  Please Note: This is a two-step process. You will need to use this procedure to upgrade to JIRA 4.4.x first before you can upgrade to **JIRA 5.0.x**. See the JIRA 5.0 Upgrade Notes for details.

If you *are already using*:

- JIRA version 4.3.0 or later on Linux or Windows (excluding JIRA WAR installations), please note the *recommended procedure* (see Upgrading JIRA) for upgrading to **JIRA 5.0.x**.
- JIRA version 4.0.0 – 4.2.x (or 4.3.0 or later on Solaris - excluding WAR), or JIRA WAR version 4.0.0 or later, please note the *manua procedure* (see Upgrading JIRA Manually) for upgrading to **JIRA 5.0.x**.

Upgrading to JIRA 5.0?

If so, please review the JIRA 5.0 Release Notes and JIRA 5.0 Upgrade Notes for important information about this version of JIRA.

On this page:

- 1. Before You Start
- 2. Backing Up
  - 2.1 Ensure that users cannot update your existing JIRA data
  - 2.2 Back up your database
  - 2.3 Back up your JIRA Home directory
  - 2.4 Back up your attachments and index directories if located outside your JIRA Home directory
  - 2.5 Back up your JIRA Installation directory
- 3. Setting up your New JIRA Installation
  - 3.1 Install the new version of JIRA
  - 3.2 Point your new JIRA to (a copy of) your existing JIRA Home directory
  - 3.3 Connect the new version of JIRA to a new, empty database
  - 3.4 Migrate your existing JIRA configurations over to your new JIRA installation
  - 3.5 Start your new version of JIRA
  - 3.6 Import your old JIRA data into your new JIRA
- 4. Post Upgrade Checks and Tasks

Please read/perform *all steps and sub-steps* in consecutive order.

1. Before You Start

1. You will need current software maintenance to perform the upgrade. If you are unsure, confirm the following:
   - Your license support period is still valid.
   - If your current license has expired but you have a new license with you, please update your license in JIRA.
     If you forget to do this and your license has expired, you will receive errors during the upgrade process. Refer to the instructions on upgrading beyond current license period.
2. Read the release notes and upgrade guide for the version of JIRA you are upgrading to. The upgrade guide (in particular) contains important information that may be relevant to your JIRA installation.
   - If you plan to skip a few JIRA versions for your next JIRA upgrade, **we strongly recommend** that you read the upgrade guides for all major versions between your current version and the version to which you are upgrading. Refer to Important Version-Specific Upgrade Notes for quick links to these guides.
3. Confirm that your operating system, database, other applicable platforms and hardware still comply with the requirements for **JIRA 5.0.x**. Newer versions of JIRA may have different requirements and supported platforms to previous JIRA versions.
   The End of Support Announcements for JIRA page also has important information regarding platform support for future versions of JIRA.
4. Some anti-virus or other Internet security tools may interfere with the JIRA upgrade process and prevent the process from completing successfully. If you experience or anticipate experiencing such an issue with your anti-virus/Internet security tool, disable this tool first before proceeding with the JIRA upgrade.
5. Check for any known issues in the JIRA Knowledge Base.
6. If you have installed any additional JIRA plugins (i.e. not included with JIRA), please verify that they will be compatible with the version of JIRA you are upgrading to. You can find a plugin's compatibility information from the the plugin's home page on the Atlassian Plugin Exchange. Once you have confirmed the availability of compatible versions, you should upgrade your plugins after successfully upgrading JIRA. This can be done via the '**Plugin Repository**' in your **Administration Console**.
7. **Test first!**— We strongly recommend performing your upgrade in a test environment first. Do not upgrade your production JIRA server until you are satisfied that your test environment upgrade has been successful.
   - If you have any problems with your test environment upgrade which you cannot resolve, create an issue at our support site so that we can assist you.
   - If you have any problems during the upgrade of your production JIRA server, do not allow your users to start using this server. Instead:
     - Continue to use your old JIRA server — this will help ensure that you do not lose production data.
     - Also create an issue at our support site so that we can help you resolve the problems with production JIRA upgrade.

2. Backing Up

Before you begin the JIRA upgrade, we strongly recommend that you back up your existing JIRA installation.

2.1 Ensure that users cannot update your existing JIRA data

In subsequent steps, you will be required to export JIRA's database from your existing JIRA installation (via an XML backup) and later restore this backup into a new JIRA installation. To ensure that data consistency in your XML backup is maintained, you must prevent users from updating your existing JIRA data by temporarily restricting access to JIRA. Refer to the Preventing users from accessing JIRA during backups page for more information. (To access this page in the documentation for another version of JIRA, click the documentation link for your version of JIRA at the top of the left Table of Contents column and use the search box below to find this page.)

Be aware! Inconsistent XML backups cannot be restored!

2.2 Back up your database

Perform an XML backup of your existing JIRA installation's external database.

- For large (corporate) JIRA installations, this process may require several hours to complete.
- The 'embedded database' is the HSQLDB database supplied with JIRA for evaluation purposes only. If you happen to accidentally

use the HSQLDB database in a production system, perform an XML backup of this database and continue on with this procedure.

2.3 Back up your JIRA Home directory

Ensure JIRA is shut down before continuing.

The location of this directory is defined within the `jira-application.properties` configuration file, which is located inside the `<jira-application-dir>/WEB-INF/classes` directory within your JIRA Installation Directory.

2.4 Back up your attachments and index directories if located outside your JIRA Home directory

Your attachments and index directories may be located outside your JIRA Home Directory. If so, they will also need to be backed up. To confirm the location of:

- Your attachments directory — refer to Configuring File Attachments page in the documentation for your version of JIRA.
- Your index directory — refer to Search Indexing page in the documentation for your version of JIRA.

To access these pages, click the documentation link for your version of JIRA at the top of the left Table of Contents column and use the search box below to find either the 'Configuring File Attachments' or 'Search Indexing' page.

Also refer to Backing Up Data for more information about backing up attachments in JIRA.

2.5 Back up your JIRA Installation directory

The 'JIRA Installation Directory' is the directory into which the JIRA application files and libraries were extracted when JIRA was installed.

3. Setting up your New JIRA Installation

> If you are running a 'mission-critical' JIRA server, we highly recommend performing the remaining steps of this guide in a test environment (e.g. using a separate test JIRA database and a copy of your JIRA Home directory) before performing the upgrade for production use.

3.1 Install the new version of JIRA

First, you must start with a fresh installation of your new JIRA version.

Download and extract the JIRA distribution you require, to a new directory. **Do not overwrite your existing JIRA installation. Ensure this has been shut down and install the new JIRA version to a new location.**

Follow the installation instructions for either:

- Installing JIRA (recommended), or
- Installing JIRA WAR

If you are using JIRA WAR, remember to build your new JIRA web application and deploy it to your server. For specific instructions, refer to the JIRA WAR installation page for your application server within the Installing JIRA WAR section.

3.2 Point your new JIRA to (a copy of) your existing JIRA Home directory

If your new **JIRA 5.0.x** installation is on a new server, **copy your existing JIRA Home Directory** from the old server to the new server before proceeding.

If your new JIRA 5.0.x installation is a 'recommended' (not WAR) distribution, you can:

1. Open the JIRA Configuration Tool.
2. Click the '**JIRA Home**' tab.
3. Update the 'JIRA Home Directory' field as follows:
   - If your **JIRA 5.0.x** installation is on a new server, update the 'JIRA Home Directory' field to the path of your copied JIRA Home directory.
   - If your **JIRA 5.0.x** installation is on the same server, update the 'JIRA Home Directory' field to the path of your **existing** JIRA Home directory.
     For more information about this directory, see JIRA Home Directory.

Otherwise:

1. Edit the `jira-application.properties` file located within the `<jira-application-dir>/WEB-INF/classes` subdirectory of your new **JIRA 5.0.x** Installation Directory JIRA Installation Directory.
2. Update the `jira.home`property in this file to the path of the new JIRA Home Directory:
   - If your **JIRA 5.0.x** installation is on a new server, update the `jira.home` property to the path of your copied JIRA Home directory.
   - If your **JIRA 5.0.x** installation is on the same server, update the `jira.home` property to the path of your **existing** JIRA Home directory.
     For more information about this directory, see JIRA Home Directory.

3. Remove the '#' at the beginning of this line (so that JIRA no longer regards this line as a comment).
4. Save your updated `jira-application.properties` file.

You can also set your JIRA Home Directory's location by defining an operating system environment variable `JIRA_HOME`. This value of this variable takes precendence over the value of the `jira.home` property in the `jira-application.properties` file in your JIRA Installation Directory. See Setting your JIRA Home Directory for details.

3.3 Connect the new version of JIRA to a new, empty database

You need to create a new, empty database that your new JIRA installation will use to store its data.

Follow the appropriate '**Connecting JIRA to...**' instructions for your database from stage 2, although from stage 4 of that procedure, be aware of the yellow note below:

- Connecting JIRA to PostgreSQL
- Connecting JIRA to MySQL
- Connecting JIRA to Oracle
- Connecting JIRA to SQL Server 2005
- Connecting JIRA to SQL Server 2008

> If you are using a database (called **jiradb** for example) with your existing JIRA installation and the database for your new JIRA installation is running on the same machine or database server, create your new database with a different name (e.g. something intuitive like **jiradb_440** for JIRA 4.4.0). However, ensure the new database has identical access permissions to the old JIRA database. Consult your database administrator if you need assistance with this.

You do not need to create a new database if you are using the embedded HSQL database.

3.4 Migrate your existing JIRA configurations over to your new JIRA installation

You may have modified a number of properties within configuration files of your existing JIRA installation.

If so, you need to **make the same modifications in your new JIRA installation**. However, do not simply copy configuration files from your existing JIRA installation and replace the equivalent files in your new JIRA installation, since the properties in these files may have changed from the old version.

For each file you have modified in your existing JIRA installation, you need to **manually edit each equivalent file in your new JIRA installation and re-apply your modifications**. If a file is not present in your new JIRA installation (for example, `osuser.xml` in recent JIRA versions), then simply copy that file over to your new JIRA installation.

The table below lists the most commonly modified files and their locations within your JIRA Installation Directory:

File	Location in 'recommended' (formerly 'Standalone') JIRA distributions	Location in JIRA WAR	Description
jira-application.properties	`atlassian-jira/WEB-INF/classes`	`webapp/WEB-INF/classes`	Location of the JIRA Home Directory and Advanced JIRA Configuration in JIRA 4.3.x and earlier. Any custom property values defi in the `jira-application.propert` file of your existing JIRA 4.3.x (c earlier) installation must be migr across to the `jira-application.propert` file of your new **JIRA 5.0.x** installation *before you start your new JIRA installation*. Upon starting your new JIRA installation, any custom property values in the `jira-application.propert` file will automatically be migrate across to either the JIRA databa or jira-config.properties file. `jira.home` is the only property the `jira-application.propert` file subsequently used by JIRA.

setenv.bat (Windows) or setenv.sh (Linux)	`bin`	Application Server's `bin` directory	Increasing JIRA Memory
osuser.xml (not required if upgrading from JIRA 4.3.0 or later)	`atlassian-jira/WEB-INF/classes`	`webapp/WEB-INF/classes`	Modified if you have integrated LDAP with JIRA, integrated Cro with JIRA, or if you are using a custom form of external user management or user authentica
seraph-config.xml	`atlassian-jira/WEB-INF/classes`	`webapp/WEB-INF/classes`	Modified if you have integrated Crowd with JIRA.
server.xml	`conf`	Application Server's `conf` directory	Modified in the following situatio • If you had previously configured JIRA's TCP ports differently from th defaults. • If you had implemented SSL. • When connecting JIRA database in JIRA 4.3.x earlier.

The version-specific upgrade notes contain details on properties which may have changed in these commonly modified files.

In addition to the files above, you should also consider and/or perform the following configurations as part of the upgrade process:

- **Using JIRA with Atlassian's Crowd?** — If you are using Crowd with JIRA, configure your new JIRA to talk to Crowd as described in Integrating Crowd with JIRA.
- **Allocating additional memory to JIRA** — If you had previously allocated additional memory to JIRA, do the same for your new JIRA instance. For more information refer to Increasing JIRA memory.
- **Plugins** — For any plugins that you had installed in your old JIRA, download the plugin version for your new version of JIRA from the http://plugins.atlassian.com site.
- **Character encoding** — Please ensure that character encoding (ie. locale) is the same on the new and old locations. You may have problems with encoding of the file names, if attachments are moved between two system with incompatible encoding.
  Your new version of JIRA may not function correctly or could encounter problems or errors if these are not implemented.
- **Customisations** — If you had made any customisations (code, templates or configuration files), copy over compatible versions of these changes to the new JIRA. (The developers within your organisation who made the customisations to your old version will need to build and test equivalent changes for the new version, and provide you with the files to copy to your upgraded JIRA installation).
- **(Optional) Disabling Email Access** — If you need to perform some initial tests on your new JIRA installation, you can disable its email access to prevent unintended emails being sent during testing. Be sure to re-enable email access once any testing is complete.
- **(Optional) Running JIRA on a different port** — If your new JIRA is installed on the same machine as your old JIRA, you may wish to make sure it runs on a different port (in case you ever need to restart your old JIRA). See Changing JIRA's TCP Ports for details.

3.5 Start your new version of JIRA

1. Verify that your old JIRA installation is shut down — if this JIRA server is still operating, shut it down.
2. If you installed the JIRA WAR distribution within **Tomcat**, **delete the Tomcat work directory before restarting JIRA**. If you do not do this, users may encounter errors when they try to display JIRA pages.
3. Start up your new version of JIRA. For:
   - **'Recommended' distributions** — follow the Starting JIRA instructions.
   - **WAR distributions** — follow the instructions for starting JIRA for your application server within the Installing JIRA WAR section.

     During the startup process, your new JIRA installation will create any required database indexes. If you created any custom database indexes, please check them afterwards and remove any that duplicate the indexes added by JIRA.

Do not restart your old JIRA installation...

If your new **JIRA 5.0.x** installation is on the same server as your old one, it may still be configured to use the same JIRA Home directory as your new JIRA installation. Running two separate JIRA installations which share a common JIRA Home directory can lead to serious data corruption.

Nevertheless, we recommend that you do not delete any aspect (or backed up component) of your old JIRA installation, until you are satisfied that your upgraded JIRA installation is functioning as expected.

3.6 Import your old JIRA data into your new JIRA

After you have successfully started your new JIRA installation, you will need to import the data from your old instance into the new instance. You will need the backup file of data from your old JIRA that you created earlier in these instructions (above).

To import your old JIRA data into your new JIRA,

1. Access JIRA via your web browser. You will see the Setup Wizard.
2. Click the **'import your existing data'** link.
3. The 'Import Existing Data' page will be displayed.
    - In the **'File name'** field, specify the XML backup file you created previously.
4. Restore the attachments directory that you backed up previously, into the attachments directory of your new JIRA. (See Restoring Data.)
   It is recommended that you avoid passing through a proxy when performing an XML restore, especially if your JIRA instance is very large. Using a proxy may cause timeout errors.
5. Access JIRA via your web browser again and log in using a username from your previous JIRA installation.
6. Take a quick look around your JIRA site to confirm that your projects and issues are present and everything looks normal. You should see the new JIRA version number in the page footer.

4. Post Upgrade Checks and Tasks

It is strongly recommended that you perform the following checks and tasks after you have started your new instance of JIRA:

1. Check your server logs for error messages, even if JIRA appears to be running correctly. If there are any errors there that you cannot resolve, create a support case in https://support.atlassian.com, attach your log file and we will advise you on the errors.
2. If you were previously using External User Management, enable it in the new JIRA instance.
3. If you changed machines when upgrading, change the paths to the indexes, attachments and backup directories, from within the **Administration** section of JIRA.
4. Enable email, if you disabled it during testing.
5. If you migrated any customisations from your old JIRA to the new JIRA, ensure that they are tested thoroughly.
    a. If you had downloaded plugins for the new version of JIRA, install the downloaded JAR file(s) in your new JIRA version and carry out any other required installation for the plugin.
    b. If the plugin has a properties file, apply the same changes to it as you had in the old properties file (don't just copy over the old properties file).

Congratulations! You have completed your JIRA migration/upgrade.

See Also

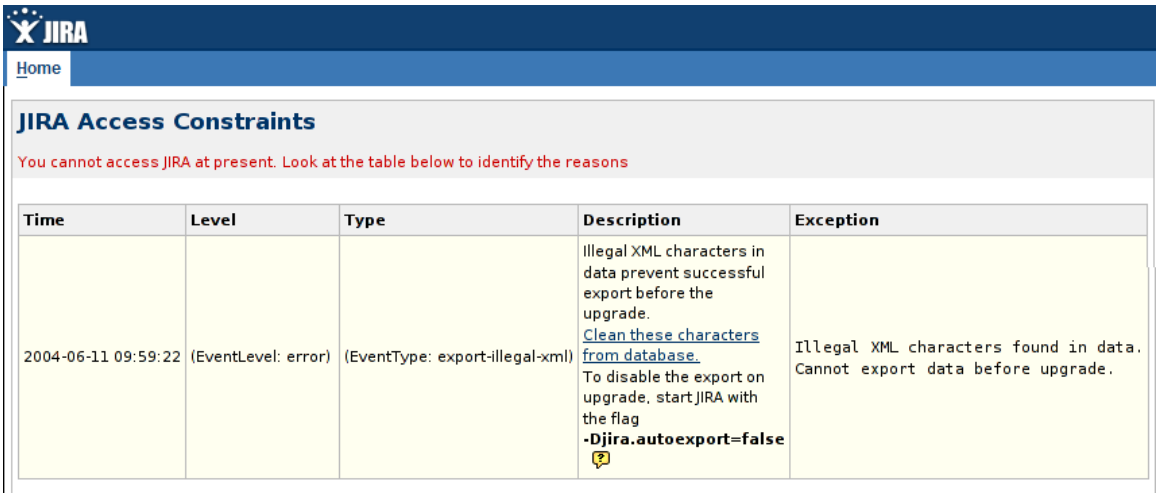
Disabling Auto-Export
Restoring Data
Upgrading JIRA
Switching Application Servers to Apache Tomcat
Switching Databases

Disabling Auto-Export

When upgrading JIRA, one points the new JIRA installation at the old JIRA database. JIRA will automatically make any structural database modifications required to support new JIRA features.

To be safe, JIRA first tries to create an XML backup of your data at the point just before the upgrade. This would allow you to 'roll back' to the old JIRA version, should anything go wrong.

Sometimes the automatic XML backup procedure fails, often resulting from characters in the database which cannot be represented in XML — such as non-displayable control characters that have been 'cut-and-pasted' into a JIRA field.



In these circumstances, you can force the upgrade to proceed by editing your `jira-config.properties` file (in the JIRA Home Directory) and setting the property `jira.autoexport=false`

See Making changes to the `jira-config.properties` file for more information.

After having successfully upgraded JIRA, it is best to remove this property (or disable it with a '#') as it should no longer be required.

If you have any upgrade problems not covered here or in the upgrade documentation, please contact us — we're happy to help.

Rolling Back a JIRA Upgrade

The 'roll back' procedures on this page describe how to restore your previous version of JIRA in the unlikely event that you encounter an issue with your JIRA upgrade. Please follow the procedure below that relates to the upgrade procedure you used.

If you upgraded JIRA using the Migrating JIRA to Another Server procedure, your previous JIRA installation should still be 'intact' (assuming you haven't deleted it) and there should not be a need to perform any 'roll back'.

Rolling Back a JIRA Upgrade Conducted Using the Upgrade Wizard

Use this procedure to roll back a JIRA upgrade conducted using the upgrade wizard.

Prior to rolling back your JIRA upgrade, ensure that you have the following backups from your previous JIRA version:

- The JIRA database (generated by your database's own backup tools).
- The JIRA Home Directory.
- The JIRA Installation Directory.

To roll back your JIRA upgrade conducted using the upgrade wizard:

1. Stop the JIRA upgrade or the upgraded JIRA server if it is running.
2. Use your database server's tools to restore the JIRA database backup you had created.
3. Delete the contents of the JIRA Installation Directory.
4. Restore the backed-up JIRA Installation Directory to the same location in the previous step.
5. Delete the contents of the JIRA Home Directory.
6. Restore the backed-up JIRA Home Directory to the same location in the previous step.
7. Start JIRA (by running the `start-jira.sh` or `start-jira.bat` file in the `bin` subdirectory of your restored JIRA installation directory).

Rolling Back a JIRA Upgrade Conducted Manually

Use this procedure to roll back a JIRA upgrade conducted using the manual JIRA upgrade procedure (involving an 'in-place' database upgrade). The intended result of this procedure is to restore your previous JIRA installation to its original state (consisting of the restored database as well as the JIRA Installation and Home directories in their original locations).

Prior to rolling back your JIRA upgrade, ensure that you have the following backups from your previous JIRA version:

- The JIRA database (generated by your database's own backup tools).
- The JIRA Home Directory.
- The JIRA Installation Directory.

To roll back your JIRA upgrade conducted manually with an 'in-place' database upgrade:

1. Stop the JIRA upgrade or the upgraded JIRA server if it is running.
2. Use your database server's tools to restore the JIRA database backup you had created.
3. If you had deleted the JIRA Installation Directory of your previous JIRA version, restore the backed-up JIRA Installation Directory to its original location.
4. Delete the contents of the JIRA Home Directory.
5. Restore the backed-up JIRA Home Directory to the same location in the previous step.
6. Start JIRA (by running the `start-jira.sh` or `start-jira.bat` file in the `bin` subdirectory of your restored JIRA installation directory).

Important Directories and Files

On this page:

- JIRA Installation Directory
  - Important Files and Directories
    - `<jira-application-dir>/WEB-INF/classes/jira-application.properties`
    - `<jira-application-dir>/WEB-INF/classes/jpm.xml`
    - `<jira-application-dir>/WEB-INF/lib/`
    - `<jira-application-dir>/WEB-INF/classes/log4j.properties`
    - `<jira-application-dir>/WEB-INF/classes/entityengine.xml`
    - `conf/server.xml`
    - Memory Settings
- JIRA Home Directory
- Important Files
  - `dbconfig.xml`
  - `jira-config.properties`
- Important Subdirectories
  - `data`
  - `export`

- `log`
- `plugins`
- `caches`
- `tmp`

JIRA Installation Directory

The 'JIRA Installation Directory' is the directory into which the JIRA application files and libraries have been extracted, either:

- by the Windows or Linux installers, or
- by extracting the contents of a JIRA installation archive or WAR distribution archive file (i.e. a `.tar.gz` or `.zip` file).

JIRA does not modify or store any data in this directory.

Important Files and Directories

The directories/files described below are found under different sub-directories of the 'JIRA Installation Directory', depending on whether you have installed a 'recommended' or WAR distribution of JIRA. Please substitute the following directories for the **`<jira-application-dir>`** placeholder below, as follows:

- **'Recommended' distributions** — the `atlassian-jira` subdirectory of the 'JIRA Installation Directory' installed using the 'Windows Installer', 'Linux Installer' or from an 'Archive File'.
- **WAR distribution** — the `webapp` subdirectory of the 'JIRA Installation Directory'.
  Please Note: To edit files in this `webapp` directory, first copy them to the `edit-webapp` subdirectory (if they are not already present in `edit-webapp`) and edit the copies in `edit-webapp`. Refer to the 'Webapp Layout' section of the JIRA WAR Configuration Overview for details. Copies of the `jira-application.properties` and `entityengine.xml` file are already available in the `edit-webapp` sub-directory.

`<jira-application-dir>/WEB-INF/classes/jira-application.properties`

This file tells JIRA where to find the JIRA Home Directory.
Be aware that your JIRA Home Directory defined in this file can be overridden. See Setting your JIRA Home Directory for more information.

`<jira-application-dir>/WEB-INF/classes/jpm.xml`

This file stores the default values for JIRA's advanced configuration settings and should not be modified. The default values of properties in this file are customised (i.e. overridden) by redefining them in either the `jira-config.properties` file (in your JIRA Home Directory) or the JIRA database (via the JIRA administration area). See Advanced JIRA Configuration for more information.

`<jira-application-dir>/WEB-INF/lib/`

This is the directory where plugins built on Atlassian's Plugin Framework 1 (i.e. 'Plugins 1' plugins) are stored. If you are installing a new 'Plugins 1' plugin, you will need to deploy it into this directory.
'Plugins 2' plugins should be stored in the JIRA Home Directory.

`<jira-application-dir>/WEB-INF/classes/log4j.properties`

JIRA's logging configuration file. See Logging and Profiling.

The actual log files generated by JIRA can be found in the following locations:

- **JIRA application log** — `bin/atlassian-jira.log`
- **Application server log** — generally the application server log file can be found under the `logs` directory. However, this can vary depending on the application server you are running. Please see Where are the application server logs? for further details.

`<jira-application-dir>/WEB-INF/classes/entityengine.xml`

This file configures the OFBiz Entity Engine which JIRA uses to store persist data in a datasource. See Configuring the Entity Engine for JIRA.

The sub-directories/files described below are found under the root of the JIRA Installation directory.

conf/server.xml

This file is used for JIRA SSL configuration. See Running JIRA over SSL or HTTPS.

Memory Settings

The file used to edit JAVA_OPTS memory settings will depend on the method used to install JIRA, as well as the operating system used for your installation.

For example, if you are running JIRA on Tomcat in Windows (manual startup), you would update the following file:
`bin\setenv.bat`
whereas for JIRA on Tomcat in Linux/Unix, you would update this file:
`bin/setenv.sh`

See Increasing JIRA Memory for further details.

JIRA Home Directory

The 'JIRA Home Directory' contains key data that help define how JIRA works. This document outlines the purpose of the various files and subdirectories within the JIRA Home Directory.

If JIRA was installed using the automated Windows or Linux installers, the default location of the JIRA Home Directory is:

- `C:\Program Files\Atlassian\Application Data\JIRA` (on Windows) or
- `/var/atlassian/application-data/jira` (on Linux)

If you install JIRA from an archive file, the JIRA Home Directory can be any suitable location that is accessible by your JIRA installation. Typical example locations might be:

- `C:\jira\home` (on Windows) or
- `/var/jira-home` (on Linux or Solaris)

However, avoid locating the JIRA Home Directory inside the JIRA Installation Directory.

For information on specifying the location of the JIRA Home Directory, please see Setting your JIRA Home Directory.

Important Files

dbconfig.xml

This file (located at the root of your JIRA Home Directory) defines all details for JIRA's database connection. This file is typically created by running the JIRA Setup Wizard on new installations of JIRA or by configuring a database connection using the JIRA Configuration Tool.

You can also create your own `dbconfig.xml` file. This is useful if you need to specify additional parameters for your specific database configuration, which are not generated by the Setup Wizard or JIRA Configuration Tool. For more information, refer to the 'manual' connection instructions of the appropriate database configuration guide in the Connecting JIRA to a Database section.

jira-config.properties

This file (also located at the root of your JIRA Home Directory) stores custom values for most of JIRA's advanced configuration settings. Properties defined in this file override the default values defined in the `jpm.xml` file (located in your JIRA Installation Directory). See Advanced JIRA Configuration for more information.

In new JIRA installations, this file may not initially exist and if so, will need to be created manually. See Making changes to the `jira-config.properties` file for more information. This file is typically present in JIRA installations upgraded from version 4.3 or earlier, whose advanced configuration options had been customised (from their default values).

Important Subdirectories

data

This directory contains application data for your JIRA instance, including attachments (for every version of each attachment stored in JIRA).

export

JIRA will place its automated backup archives into this directory.

log

JIRA will place its logs into this directory. (Note: if the JIRA home directory is not configured, then the logs will be placed into the current working directory instead).

The logs will only start showing up once the first log message is written to them. For example, the internal access log will not be created util JIRA starts writing to it.

You can change the location of the log file using `log4j.properties` as described in the documentation on Logging and Profiling.

`plugins`

This is the directory where plugins built on Atlassian's Plugin Framework 2 (i.e. 'Plugins 2' plugins) are stored. If you are installing a new 'Plugins 2' plugin, you will need to deploy it into this directory under the `installed-plugins` sub-directory.

'Plugins 1' plugins should be stored in the JIRA Installation Directory.

This directory is created on JIRA startup, if it does not exist already.

`caches`

This is where JIRA stores caches including:

- Lucene indexes - see Searching, Indexing, and Filters Troubleshooting
- OSGi framework caches

These files are vital for JIRA performance and should not be modified or removed externally while JIRA is running.

Note:

See Search Indexing for further details.

`tmp`

Any temporary content created for various runtime functions such as exporting, importing, file upload and indexing is stored under this directory.

You can remove files from this directory while JIRA is running, but we recommend that you shut down JIRA first before altering the contents of this directory.

JIRA Installation Directory

The 'JIRA Installation Directory' is the directory into which the JIRA application files and libraries have been extracted, either:

- by the Windows or Linux installers, or
- by extracting the contents of a JIRA installation archive or WAR distribution archive file (i.e. a `.tar.gz` or `.zip` file).

JIRA does not modify or store any data in this directory.

Important Files and Directories

The directories/files described below are found under different sub-directories of the 'JIRA Installation Directory', depending on whether you have installed a 'recommended' or WAR distribution of JIRA. Please substitute the following directories for the **`<jira-application-dir>`** placeholder below, as follows:

- **'Recommended' distributions** — the `atlassian-jira` subdirectory of the 'JIRA Installation Directory' installed using the 'Windows Installer', 'Linux Installer' or from an 'Archive File'.
- **WAR distribution** — the `webapp` subdirectory of the 'JIRA Installation Directory'.
  Please Note: To edit files in this `webapp` directory, first copy them to the `edit-webapp` subdirectory (if they are not already present in `edit-webapp`) and edit the copies in `edit-webapp`. Refer to the 'Webapp Layout' section of the JIRA WAR Configuration Overview for details. Copies of the `jira-application.properties` and `entityengine.xml` file are already available in the `edit-webapp` sub-directory.

`<jira-application-dir>/WEB-INF/classes/jira-application.properties`

This file tells JIRA where to find the JIRA Home Directory.
Be aware that your JIRA Home Directory defined in this file can be overridden. See Setting your JIRA Home Directory for more information.

`<jira-application-dir>/WEB-INF/classes/jpm.xml`

This file stores the default values for JIRA's advanced configuration settings and should not be modified. The default values of properties in this file are customised (i.e. overridden) by redefining them in either the `jira-config.properties` file (in your JIRA Home Directory) or the JIRA database (via the JIRA administration area). See Advanced JIRA Configuration for more information.

<jira-application-dir>/WEB-INF/lib/

This is the directory where plugins built on Atlassian's Plugin Framework 1 (i.e. 'Plugins 1' plugins) are stored. If you are installing a new 'Plugins 1' plugin, you will need to deploy it into this directory.
'Plugins 2' plugins should be stored in the JIRA Home Directory.

<jira-application-dir>/WEB-INF/classes/log4j.properties

JIRA's logging configuration file. See Logging and Profiling.

The actual log files generated by JIRA can be found in the following locations:

- **JIRA application log** — `bin/atlassian-jira.log`
- **Application server log** — generally the application server log file can be found under the `logs` directory. However, this can vary depending on the application server you are running. Please see Where are the application server logs? for further details.

<jira-application-dir>/WEB-INF/classes/entityengine.xml

This file configures the OFBiz Entity Engine which JIRA uses to store persist data in a datasource. See Configuring the Entity Engine for JIRA.

The sub-directories/files described below are found under the root of the JIRA Installation directory.

conf/server.xml

This file is used for JIRA SSL configuration. See Running JIRA over SSL or HTTPS.

Memory Settings

The file used to edit JAVA_OPTS memory settings will depend on the method used to install JIRA, as well as the operating system used for your installation.

For example, if you are running JIRA on Tomcat in Windows (manual startup), you would update the following file:
`bin\setenv.bat`
whereas for JIRA on Tomcat in Linux/Unix, you would update this file:
`bin/setenv.sh`

See Increasing JIRA Memory for further details.

JIRA Home Directory

The 'JIRA Home Directory' contains key data that help define how JIRA works. This document outlines the purpose of the various files and subdirectories within the JIRA Home Directory.

If JIRA was installed using the automated Windows or Linux installers, the default location of the JIRA Home Directory is:

- `C:\Program Files\Atlassian\Application Data\JIRA` (on Windows) or
- `/var/atlassian/application-data/jira` (on Linux)

If you install JIRA from an archive file, the JIRA Home Directory can be any suitable location that is accessible by your JIRA installation. Typical example locations might be:

- `C:\jira\home` (on Windows) or
- `/var/jira-home` (on Linux or Solaris)

However, avoid locating the JIRA Home Directory inside the JIRA Installation Directory.

For information on specifying the location of the JIRA Home Directory, please see Setting your JIRA Home Directory.

Important Files

dbconfig.xml

This file (located at the root of your JIRA Home Directory) defines all details for JIRA's database connection. This file is typically created by running the JIRA Setup Wizard on new installations of JIRA or by configuring a database connection using the JIRA Configuration Tool.

You can also create your own `dbconfig.xml` file. This is useful if you need to specify additional parameters for your specific database

configuration, which are not generated by the Setup Wizard or JIRA Configuration Tool. For more information, refer to the 'manual' connection instructions of the appropriate database configuration guide in the Connecting JIRA to a Database section.

`jira-config.properties`

This file (also located at the root of your JIRA Home Directory) stores custom values for most of JIRA's advanced configuration settings. Properties defined in this file override the default values defined in the `jpm.xml` file (located in your JIRA Installation Directory). See Advanced JIRA Configuration for more information.

In new JIRA installations, this file may not initially exist and if so, will need to be created manually. See Making changes to the `jira-config.properties` file for more information. This file is typically present in JIRA installations upgraded from version 4.3 or earlier, whose advanced configuration options had been customised (from their default values).

Important Subdirectories

`data`

This directory contains application data for your JIRA instance, including attachments (for every version of each attachment stored in JIRA).

`export`

JIRA will place its automated backup archives into this directory.

`log`

JIRA will place its logs into this directory. (Note: if the JIRA home directory is not configured, then the logs will be placed into the current working directory instead).

The logs will only start showing up once the first log message is written to them. For example, the internal access log will not be created util JIRA starts writing to it.

You can change the location of the log file using `log4j.properties` as described in the documentation on Logging and Profiling.

`plugins`

This is the directory where plugins built on Atlassian's Plugin Framework 2 (i.e. 'Plugins 2' plugins) are stored. If you are installing a new 'Plugins 2' plugin, you will need to deploy it into this directory under the `installed-plugins` sub-directory.

'Plugins 1' plugins should be stored in the JIRA Installation Directory.

This directory is created on JIRA startup, if it does not exist already.

`caches`

This is where JIRA stores caches including:

- Lucene indexes - see Searching, Indexing, and Filters Troubleshooting
- OSGi framework caches

These files are vital for JIRA performance and should not be modified or removed externally while JIRA is running.

Note:

See Search Indexing for further details.

`tmp`

Any temporary content created for various runtime functions such as exporting, importing, file upload and indexing is stored under this directory.

You can remove files from this directory while JIRA is running, but we recommend that you shut down JIRA first before altering the contents of this directory.

Setting your JIRA Home Directory

The JIRA Home Directory contains key data that help define how JIRA works. You must have a JIRA home directory specified for your JIRA instance before you can start it. This document describes how to specify the location of the JIRA home directory for your JIRA instance.

> **One JIRA home per JIRA instance**
>
> You can only have one JIRA Home Directory per JIRA installation. If you have multiple JIRA installations, you will need to set up a JIRA Home Directory for each installation. A lock is placed at the root level of a JIRA Home Directory when it is created to ensure that it can only used by one JIRA installation.

You only need to specify the location of the root directory for your JIRA home. The sub-directories will be created automatically when JIRA is started or when you use a function in JIRA that requires a particular sub-directory.

How do I set my JIRA home?

There are a few methods available for specifying the location of your JIRA Home Directory in JIRA. However, please be aware of the notes below before your specify this location.

Recommended Methods

The recommended methods for specifying the location of your JIRA Home Directory in JIRA are to:

- Use the JIRA Configuration Tool to change the location of your JIRA Home Directory.
  The JIRA Configuration Tool is not available in JIRA WAR distributions.
- Edit the `jira-application.properties` file and set the value of the '`jira.home`' property to the desired location for your JIRA Home Directory. If you are specifying this location's path on Windows, use double back-slashes ("\") between subdirectories. For example, `X:\\path\\to\\JIRA\\Home`
  See the JIRA Installation Directory page to find where this file is located.
- Set an environment variable named `JIRA_HOME` in your operating system whose value is the location of your JIRA Home Directory. To do this:
  - On Windows, do one of the following:
    - Configure this environment variable through the Windows user interface (typically through 'My Computer' or 'Computer')
    - At the command prompt, enter the following command (with your own JIRA Home path) before running JIRA from the command prompt:
      - `set JIRA_HOME=X:\path\to\JIRA\Home`
        Please set your `JIRA_HOME` environment variable value using this format, where:
        - `X` is the drive letter where your JIRA Home Directory is located and
        - no spacing has been added around the equal sign ('=')
    - Specify the command above in a batch file used to start JIRA.
  - On Linux/Solaris, do one of the following:
    - Enter the following command at a shell/console prompt (with your own JIRA Home path) before running JIRA:
      - `export JIRA_HOME=/path/to/jira/home`
    - Specify the command above in a script used to start JIRA.

Please Note: If you have specified different values for a '`jira.home`' property in the `jira-application.properties` file and a `JIRA_HOME` environment variable, the value of the `JIRA_HOME` environment variable takes precedence.

Alternative Method

Alternatively, you can specify the location of your JIRA Home Directory as property within your application server:

- Configure a new web context property called '`jira.home`' for your application server. To do this, you need to define this web context property inside a `<parameter/>` element (as a child of the `<context/>` element) in your `server.xml` file (or `jira.xml` file for JIRA WAR).
  The `server.xml` file is located within the `conf` subdirectory of your JIRA Installation Directory and for JIRA WAR, the `jira.xml` file is typically located within the `conf/Catalina/localhost` subdirectory of the Apache Tomcat installation running JIRA:

```
<Context ...>
    ...
        <Parameter name="jira.home" value="c:/jira/home"/>
    ...
</Context>
```

Please Note: A '`jira.home`' web context property defined in your application server overrides the value of the '`jira.home`' property defined in your `jira-application.properties` file. However, a `JIRA_HOME` environment variable defining your JIRA Home Directory will override either of these '`jira.home`' values.

What location should I specify for my JIRA home?

You can specify any location on a disk for your JIRA home directory. Please be sure to specify an absolute path.

Please note that you cannot use the same JIRA home directory for multiple instances of JIRA. We recommend locating your JIRA Home Directory completely independently of the JIRA Installation Directory (i.e. not nesting one within the other) as this will minimise information being lost during major operations (e.g. backing up and restoring instances).

How do I change my JIRA home?

To change the location of your JIRA home directory,

1. Set your JIRA home to the new location, using your preferred method as described in *"How do I set my JIRA home?"* (above).
2. Restart JIRA.

Notes:

- If you are using the Windows installer, you do not need to configure the JIRA Home Directory separately as you will be prompted to specify this location during the installation process.
- If you installed a 'Recommended' distribution of JIRA, you do not need to do this — please see the JIRA Configuration Tool instead.
- If you are using JIRA WAR, you need to set your JIRA home *before* you build JIRA.

Tomcat security best practices

The following outlines some basic techniques to secure an Apache Tomcat instance. This is a *basic must-do* list and should not be considered comprehensive. For more advanced security topics see the "Further Information" section below.

- User Permissions
  - Unix/Linux cheat-sheet
  - Windows cheat-sheet
- Tomcat Installation Permissions
  - Unix/Linux cheat-sheet
  - Windows cheat-sheet
- Web-Application Installation Permissions
  - Unix/Linux cheat-sheet
  - Windows cheat-sheet
- Further Information

User Permissions

Tomcat should never be run as a privileged user (root on UNIX or Admistrator or Local System on Windows).

Tomcat should be run as a low-privilege user. Ideally it should be run as a user created only for the purpose of running one application.

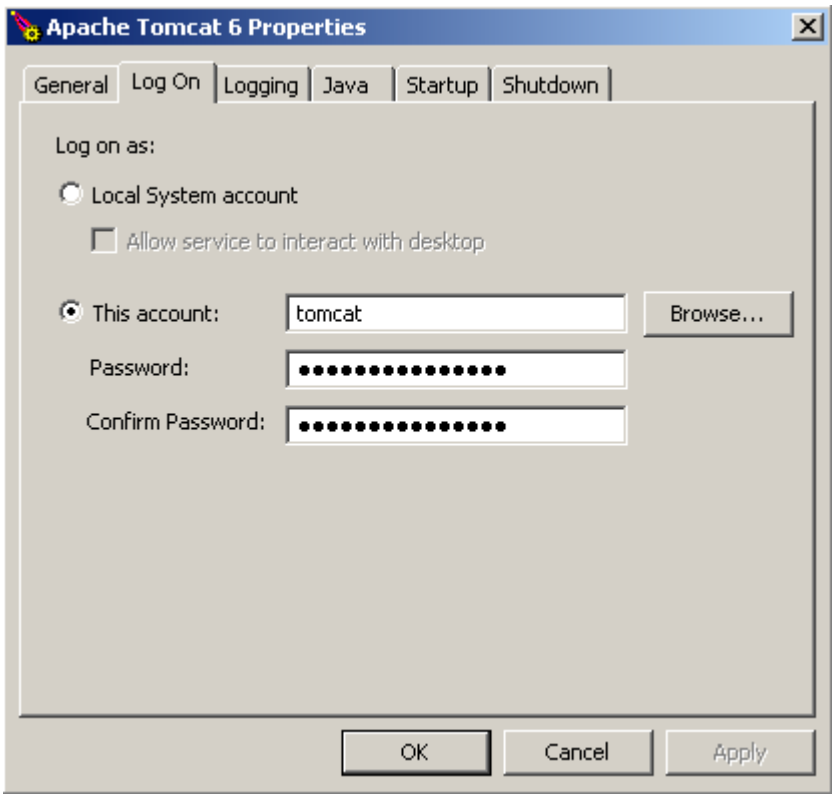
In practice this means you can't run it on port 80. If you need to run Tomcat on port 80, you should put it behind a webserver such as Apache; see Integrating JIRA with Apache for an example configuration.

Unix/Linux cheat-sheet

- Create a JIRA user:
  ```
  sudo adduser jira-tomcat
  ```
- Run Tomcat as a specific user:
  ```
  sudo -u jira-tomcat ${CATALINA_HOME}/bin/catalina.sh run
  ```

Windows cheat-sheet

- Create an unprivileged account (if your host is part of an Active Directory there may be already a template for service users). The user should have "Log on as a service" rights assigned to it.
- Make sure the Apache Tomcat service is set to run as that user. If Tomcat was installed using the JIRA Windows Installer, the system tray utility lets you set this via Configuration -> Log On:



Tomcat Installation Permissions

The Tomcat installation directory (sometimes referred to as CATALINA_HOME) should be installed as a user that is different to the one it will be run as. Under Linux, unpacking the Tomcat distribution as root is the simplest method of doing this.

Unfortunately, Tomcat does require write access to some directories in the distribution directory, but they should be enabled only as needed.

Tomcat ships with some default admin applications in its webapps directory. Unless you need these they should be disabled.

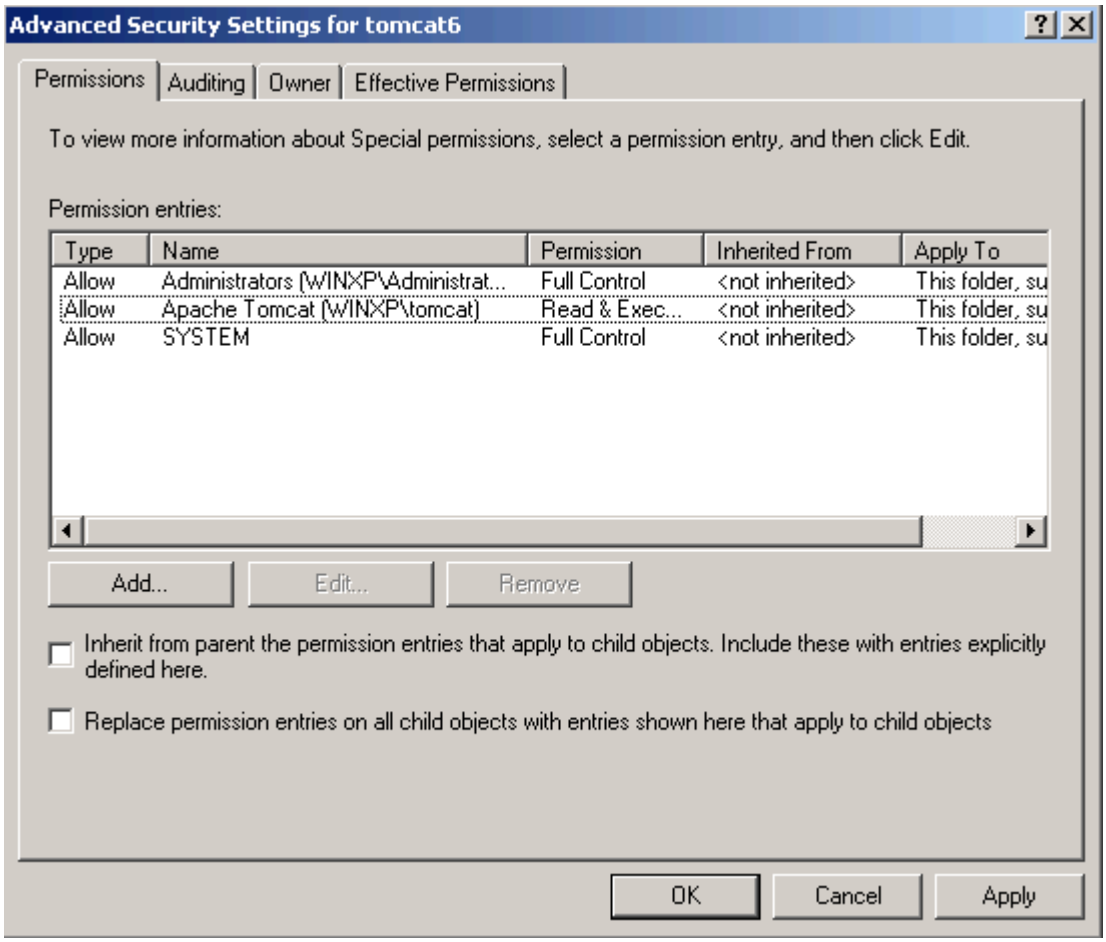
Unix/Linux cheat-sheet

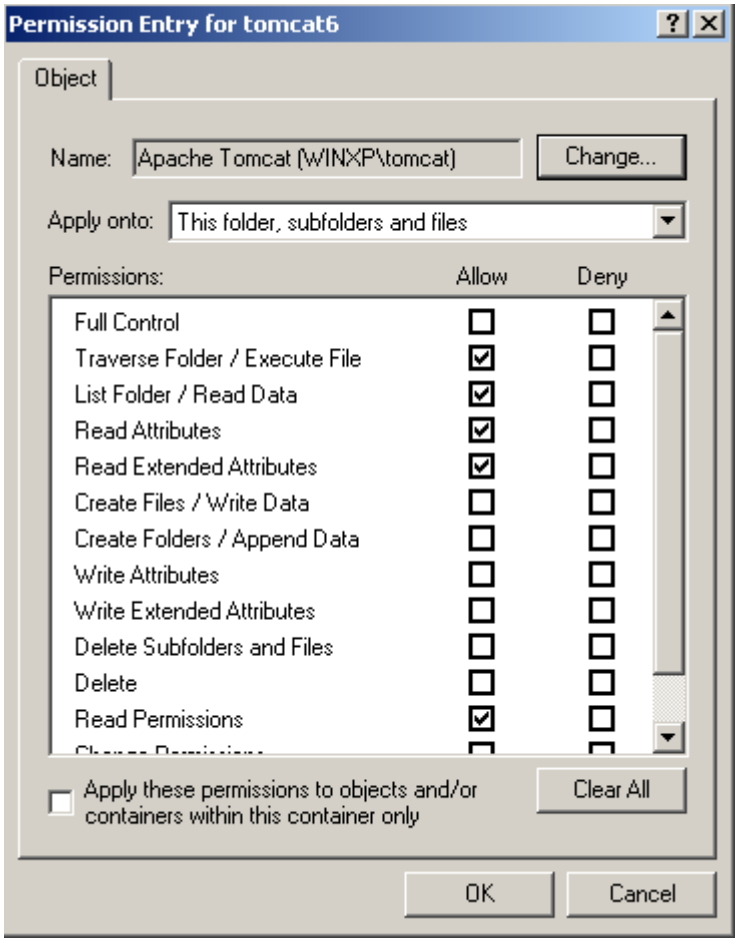
- Unpack Tomcat as root:
  `sudo tar xzvf apache-tomcat-6.0.20.tar.gz`
- Remove the default webapps:
  `sudo rm -rf apache-tomcat-6.0.20/webapps/*`
- Remove write permissions:
  `sudo chmod -R go-w apache-tomcat-6.0.20`
- Allow write on needed directories only:
  `cd apache-tomcat-6.0.20/; sudo chown -R jira-tomcat work/ temp/ logs/`

Windows cheat-sheet

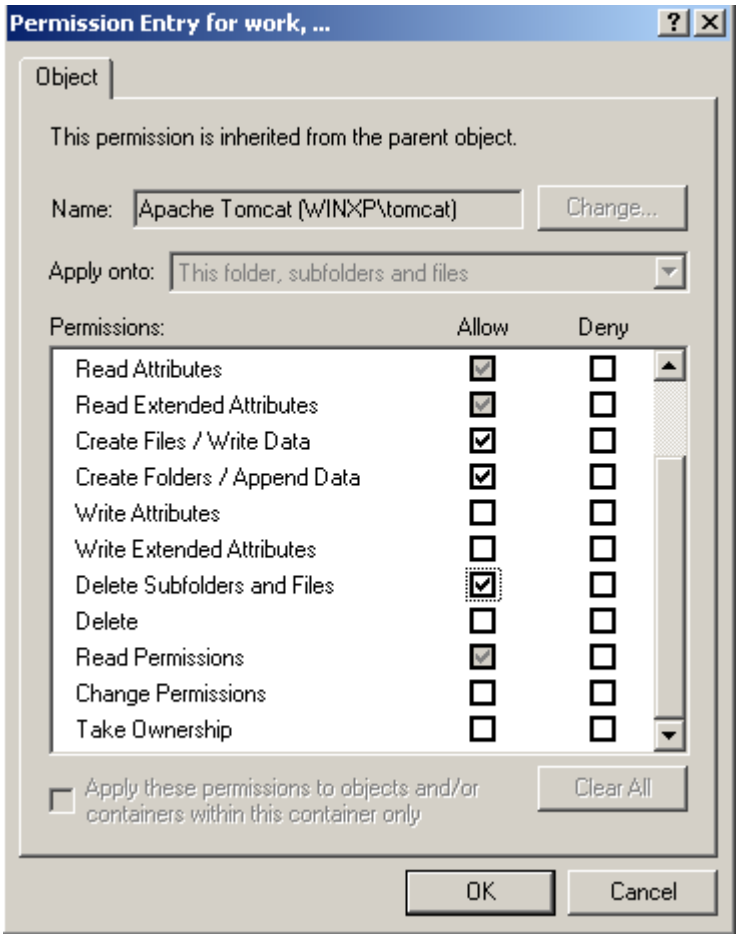
Note: If your host is part of a Domain/Active Directory, consult your Windows system administrator sysadmins to get the right permissions.

- Unpack and install Tomcat, and update the permissions on CATALINA_HOME to be writeable by Administrators and System only. The Tomcat service user should have read, execute and directory traverse privileges





- Under CATALINA_HOME, the `work`, `temp` and `logs` directories need write and delete access for the Tomcat user. Make sure it does not have permissions to change permission or take ownership.



Web-Application Installation Permissions

The directory you unpack the application WAR into should not be writable by the Tomcat user (i.e. `jira-tomcat` in the examples above). Again, the simplest method to do this is to unpack the WAR as root.

Unix/Linux cheat-sheet

- Unpack the war as root:
  `sudo unzip confluence-webapp-3.2.war`

Windows cheat-sheet

- Unpack the war as your user and confirm that the Tomcat user does not have write access to the webapp.

Further Information

- Securing Tomcat at OWASP.
- Critical Steps to Secure Tomcat on Windows NT/2K/XP
- Tomcat Security FAQ

Customising Your JIRA Installation

- Changing JIRA's TCP Ports
- Running JIRA in a Virtualised Environment
- Running JIRA over SSL or HTTPS
- Installing Confluence and JIRA Together
- Integrating JIRA with a Web Server
- Securing JIRA with Apache HTTP Server

Changing JIRA's TCP Ports

If you installed JIRA using the 'Windows Installer', 'Linux Installer' or from an 'Archive File', you can use the JIRA Configuration Tool to reconfigure the TCP ports that JIRA runs through. The JIRA WAR distribution does not include this tool.

Why Change JIRA's TCP Ports?

By default, JIRA uses TCP listening port **8080** (including default Apache Tomcat installations running JIRA WAR) and hence, JIRA is typically available at `http://<yourserver>:8080`.

If, however, an existing service running on your machine is claiming port **8080**, there will be a conflict and JIRA will fail to start. You may see errors like this:

```
LifecycleException:  Protocol handler initialization failed: java.net.BindException: Address
already in use:8080
```

This can be fixed by changing JIRA to use another TCP listening port (eg. **8100**) and shutdown port (eg. **8015**).

To do this, edit the `server.xml` file in the `conf` subdirectory of the JIRA Installation Directory (or of the Apache Tomcat installation that runs your JIRA WAR installation).

If use JIRA WAR, please be aware that changing JIRA's TCP ports of the Tomcat installation running JIRA will affect any other web applications deployed to the same Tomcat installation.

The start of the file looks like:

```
<Server port="8005" shutdown="SHUTDOWN">

  <Service name="Catalina">

    <Connector port="8080"
      maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
      enableLookups="false" redirectPort="8443" acceptCount="100" connectionTimeout="20000"
disableUploadTimeout="true" />

    ...
```

For example, change the shutdown port from "**8005**" to "**8015**" and the listening port (i.e. in the `<connector/>` element) from "**8080**" to "**8100**". (See below to decide which TCP port numbers should be used for JIRA.)

Then, restart JIRA and point a browser to `http://<yourserver>:8100`

If you are running on a Unix server and bind the ports below 1024 (such as port 80 for example), you will **need to start JIRA as root** in order to successfully bind to the port.

Which port number should I choose?

If you are not sure which port number to choose, use a tool such as *netstat* to determine which port numbers are free to use by JIRA. The highest port number that can be used is 65535 because it is the highest number which can be represented by an unsigned 16 bit binary number. The Internet Assigned Numbers Authority (IANA) lists the registration of commonly used port numbers for well-known Internet services, it's advisable to avoid any of those ports.

A note about firewalls

When you choose a port number for JIRA, bear in mind that your firewall may prevent people from connecting to JIRA based on the port number. Organisations with a local network protected by a firewall typically need to consider modifying their firewall configuration whenever they install a web-based application (such as JIRA) that is running on a new port or host. Even personal laptop and desktop machines often come with firewall software installed that necessitates the same sort of change as described above.

If JIRA does not need to be accessed from outside the firewall, then no firewall configuration changes will be necessary.

RELATED TOPICS

Changing Confluence's listening ports

Running JIRA in a Virtualised Environment

This page provides some performance data and observations on running JIRA with VMware. The information on this page is intended to help you decide whether or not to run JIRA using a VMware product. It does not contain detailed instructions on how to set this up (please see the VMware product documentation instead). We currently only provide information for VMware as it is the most requested platform from our customers. At this time, we do not have plans to officially support other virtualised environments.

On this page:

- Summary
- Recommendations
  - General

    - VMware ESX 3.5
    - VMware ESX 4i
- Performance Testing Setup
    - Server Configuration
    - Installed Software
    - Testing Tool
- Test Results
    - Low-load JIRA
    - Medium-load JIRA

Summary

Unsurprisingly, JIRA is generally slower in a virtualised environment. As can be seen in the test results below, the amount by which JIRA slows down varies based on the workload.

Under low load there are several operations which are in fact faster under VMware. This is probably due to the 4CPU VM instance running on 8 real CPUs as opposed to there being only 4 real CPUs on the baseline machine.

Please note, no performance tuning was applied to VMware for these tests. It may be possible to improve JIRA performance by tuning VMWare, however this may cause other applications to run more slowly on the virtual environment. We recommend that you consult the VMware documentation before deciding whether to do this.

Recommendations

General

- If you are a running a **high-load** instance, your biggest performance gain will be to run the application and database on a real machine and not on virtual infrastructure.
- Under **high-load**, moving the database onto another machine will help.
- Always ensure that there are enough virtual CPUs and memory allocated to the virtual instance. This may not be possible under VMware ESX 3.5 due to limitations of 4 vCPUs per VM.
- Always ensure that there is enough CPU time and memory available on the physical host to service all VMs. Applications should not go into swap.
- Use modern CPUs with VT extensions — there is still a noticeable performance penalty for using a VM with these CPUs, but it will likely be much higher when using old CPUs.
- Carefully monitor your VMware hosts to ensure that there is no resource starvation.
- Review Timekeeping best practices for Linux guests as not having the correct kernel parameters can cause high CPU usage inside the guest

VMware ESX 3.5

- If possible, upgrade to VMware ESX 4i.
- Under **low-load**, using a non-virtualised database will generally result in better response times.

VMware ESX 4i

- Under **low-load**, keep the database inside the virtual machine if there is enough CPU time for both the database and application.
- Using VMware EX 4i and virtual machine version 7, you will be able to allocate up to 8 vCPUs to an instance.

Performance Testing Setup

Server Configuration

All testing was performed on the following hardware. In the case of virtual machines, one VM per machine was configured.

Platform	CPU	Real Ram	Disk	Virtualisation Software	Virtual machine version	Virtual CPU's	Virtual Ram
Dell R610	2 x Intel 'Nehalem' Xeon E5520 (Quad Core) [1]	32Gb (8x 4Gb DDR3)	2 x 15K 146Gb SAS, Raid 1 [4,5]	VMware ESX 3.5	4	4	32Gb
Dell R610	2 x Intel 'Nehalem' Xeon E5520 (Quad Core) [1]	32Gb (8x 4Gb DDR3)	2 x 15K 146Gb SAS, Raid 1 [4,5]	VMware ESXi 4	7	4	32Gb

Dell R610	2 x Intel 'Nehalem' Xeon E5520 (Quad Core) [2,3]	32Gb (8x 4Gb DDR3)	2 x 15K 146Gb SAS, Raid 1 [5]	N/A	N/A	N/A	N/A

Notes:

1. *VT extensions were enabled in the BIOS on the machines running VMWare.*
2. *VT extensions were disabled in the BIOS on the machines not running VMWare, as per Dell best practices.*
3. *In order to limit the CPUs in the baseline test to match the number in VMWare, the kernel boot parameter `maxcpus=4` was added to the startup.*
4. *The full disk was allocated to VMware.*
5. *The filesystem used in all machines was EXT3.*

Installed Software

Each server was set up with identical software, as follows:

Atlassian Product	JIRA 4.0.0-Beta2
Database	MySQL 5.0.45-7
Application Server	Tomcat 5.5.27
Java	Java(TM) SE (build 1.6.0_07-b06), Java HotSpot(TM) 64-Bit Server VM (build 10.0-b23, mixed mode)

Operating System	Redhat Enterprise Linux 5.3 (Tikanga) 64bit (Kernel 2.6.18-128.2.1.el5). The file system used for all tests was EXT3 with the default options.
	The following tuning was applied to the operating system, in order to allow for more memory usage by the database server and better network throughput:
	`net.ipv4.ip_forward = 0`
	`net.ipv4.conf.default.rp_filter = 1`
	`net.ipv4.conf.default.accept_source_route = 0`
	`kernel.sysrq = 0`
	`kernel.core_uses_pid = 1`
	`net.ipv4.tcp_syncookies = 1`
	`kernel.msgmnb = 65536`
	`kernel.msgmax = 65536`
	`kernel.shmmax = 1310720000`
	`kernel.shmall = 4294967296`
	`net.core.rmem_max = 16777216`
	`net.core.wmem_max = 16777216`
	`net.ipv4.tcp_rmem = 4096 87380 16777216`
	`net.ipv4.tcp_wmem = 4098 65536 16777216`
	`net.ipv4.tcp_no_metrics_save = 1`
	`net.ipv4.tcp_moderate_rcvbuf = 1`
	`net.core.netdev_max_backlog = 2500`

Testing Tool

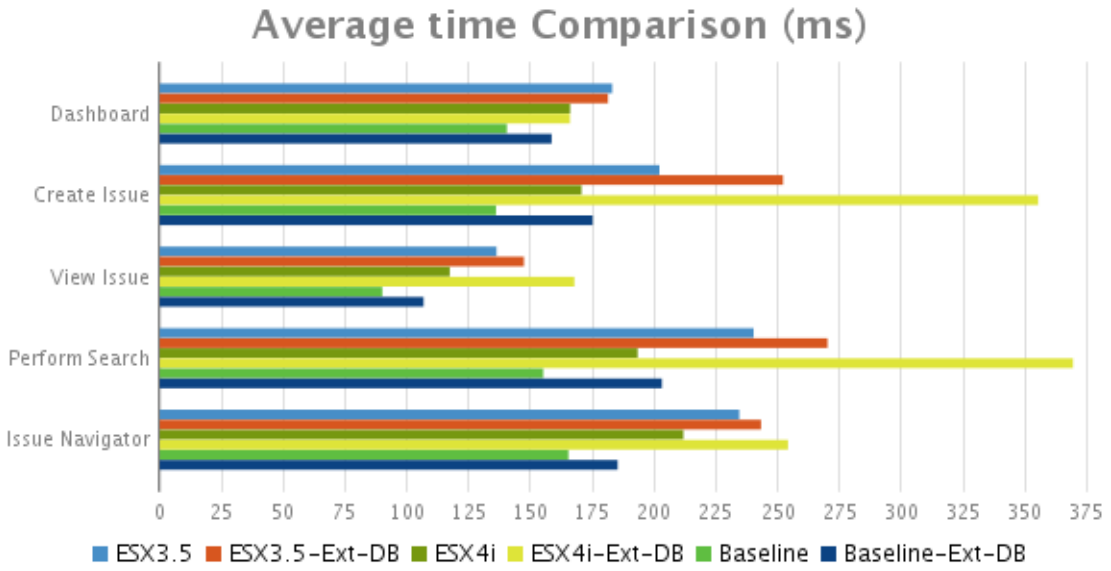
Performance tests were conducted with **Apache Jakarta JMeter 2.3.4** using the standard JIRA performance tests.

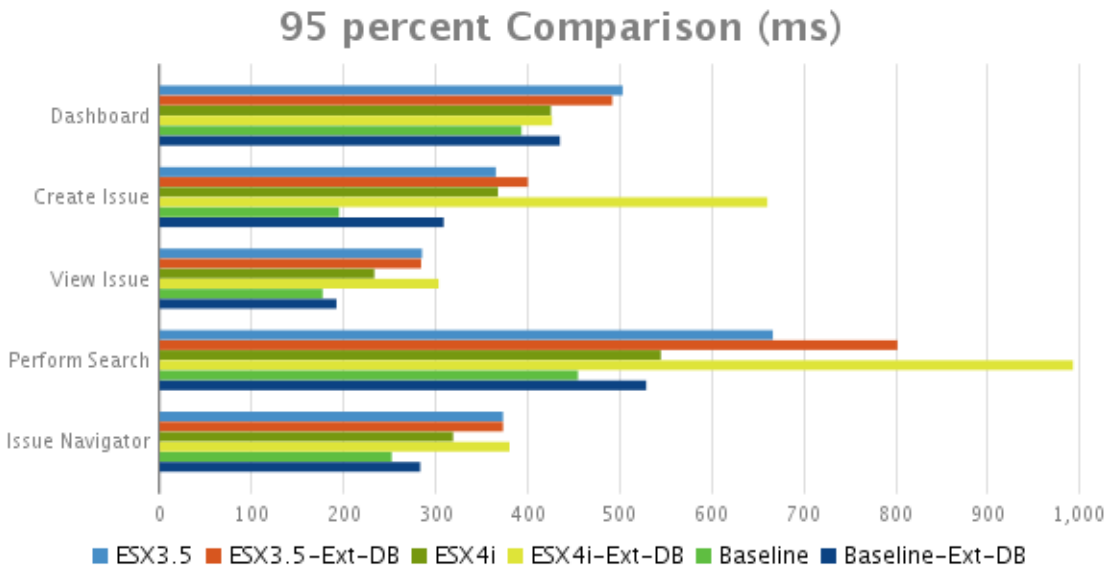
Test Results

The following tests were performed for each application. In each case, the test was performed with a database local to the host instance (i.e. in the same operating system image) and also with the database residing on a separate, non-virtualised physical server of the same specifications as above.

Low-load JIRA

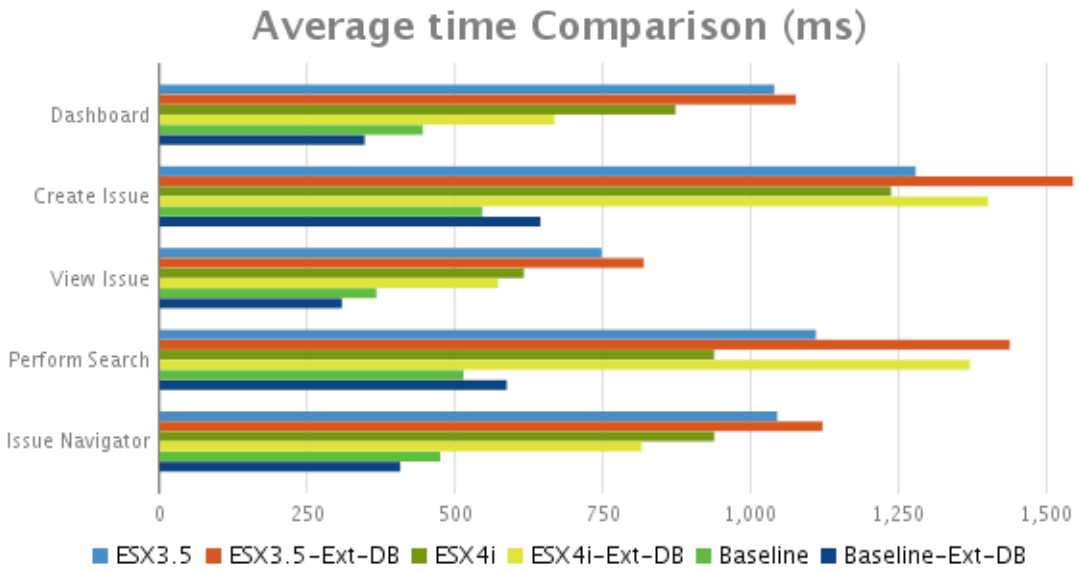
This test performs around **16 requests/second** on the JIRA instance. This is not enough to saturate the host CPU time and during the test there is around **60-80% idle time**.

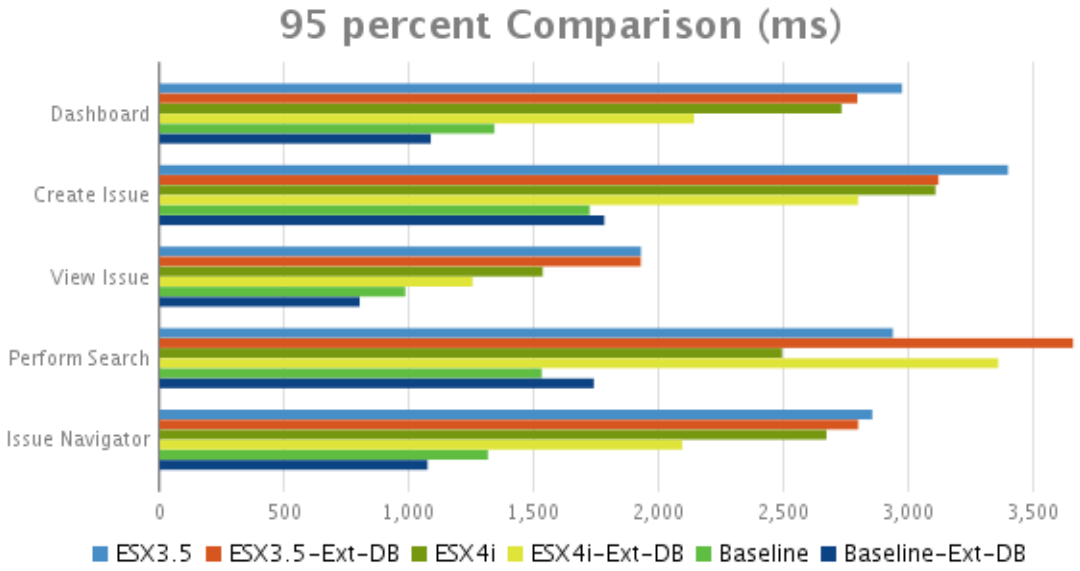




Medium-load JIRA

This test tries to perform double the requests/second of the low load test (i.e. approximately **32 requests/second**) on the JIRA instance. This is enough load to saturate the available CPU time on a 4 CPU machine.





Running JIRA over SSL or HTTPS

When web applications are being accessed across the internet, there is always the possibility of usernames and passwords being intercepted by intermediaries between your computer and the ISP/company. It is often a good idea to enable access via HTTPS (HTTP over SSL) and make this a requirement for pages where passwords are sent. Note, however, that using HTTPS may result in slower performance.In some cases where issue data is sensitive, all pages should be accessed via HTTPS.

Please note that Atlassian Support will refer SSL support to the institution that issues the Certificate. We provide this documentation for reference.

The process of enabling SSL access is specific to each application server, but the process for specifying which pages require protection is generic.

This procedure is a general guide for the way to configure **Tomcat** with HTTPS and only covers the common installation types of JIRA. It is by no means a definitive or comprehensive guide to configuring HTTPS and may not be applicable to your specific integration.

For JIRA installations installed using Windows Installer:

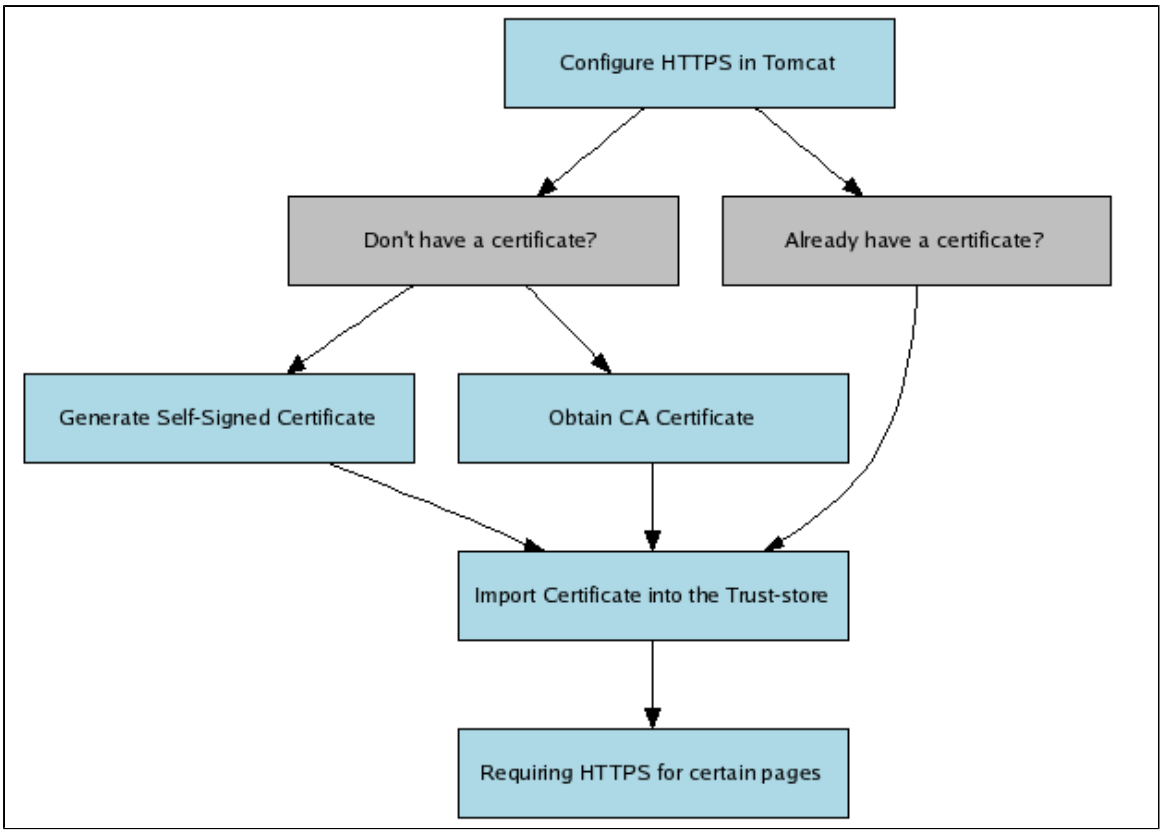
- The 'Windows Installer' installs its own Java Runtime Environment (JRE) Java platform, which is used to run Tomcat. When updating SSL certificates, please do so in this JRE installation.
- In this document, the term `<jira-install-dir>` refers to the JIRA Installation Directory itself.

On this page:

- Running JIRA over HTTPS
  - Generate Self-Signed Certificate
    - 1. Create the Keystore File
    - 2. Export the Certificate
    - 3. Import the Certificate
  - Obtain CA Certificate
  - Import Certificate into the Trust-store
  - Configure HTTPS in Tomcat
  - Redirecting certain pages to HTTPS
- Troubleshooting
  - SSL + Apache + IE problems
  - Can't find the keystore
  - Certificate reply and certificate in keystore are identical
  - Incorrect password
  - Passwords don't match
  - Wrong certificate
  - Using Apache Portable Runtime
    - Use the Http11Protocol to handle SSL connections
    - Configure the Connector to use the APR protocol
  - Enabling Client Authentication

Running JIRA over HTTPS

The following flowchart shows the process involved in configuring HTTPS on Tomcat. Click the links below this chart to go to the instructions for that step.



- Configure HTTPS in Tomcat
- Generate Self-Signed Certificate
- Obtain CA Certificate
- Import Certificate into the Trust-store
- Requiring HTTPS for certain pages (Redirecting certain pages to HTTPS)

Generate Self-Signed Certificate

Please Note:

- Self-signed certificates are useful in cases where you require encryption but do not need to verify the website identity.
- They are commonly used for testing and on internal corporate networks (intranets).
- Due to the certificate not being signed by a Certification Authority (CA), users may get prompted that the site is untrusted and may have to perform several steps to "accept" the certificate before they can access the site. This usually will only occur the first time they access the site.

1. Create the Keystore File

The following approach to create the certificate uses Java's keytool, and has been formatted for use with Java 1.6.

There are other tools for generating certificates such as openSSL which are not discussed in this procedure.

When running the following keytool command you will be prompted with:

```
What is your first and last name?
```

Instead of entering your first and last name as specified, you **must** enter the fully qualified hostname of the server running JIRA. This is the same as the name you would type in your web browser after the http:// (no port number) section to access your JIRA installation. When the client web browser examines the certificate, it checks this field, and makes sure that it matches the hostname. If it doesn't, it may prevent access to the site, and at the very least will generate pop-up messages saying that there is a mismatch. An example of a qualified hostname is: support.atlassian.com. Also, make sure the qualified host name matches the base URL you have set in JIRA (without the port).

The keytool utility will also prompt you for two passwords: the keystore password and the key password for Tomcat. You must use the same value for both passwords, and the value must be either:

a. "changeit" (this is the default value Tomcat expects), or
b. if you use a value other than "changeit", you must also specify it in `conf/server.xml`. You must add the following attribute to the Connector tag described above:

```
keystorePass="<password value>"
```

JIRA Installations on Windows:

- Installed using the 'Windows Installer':

```
"<jira-install-dir>\jre\bin\keytool" -genkey -alias tomcat -keyalg RSA
```

- Installed using the 'archive' or 'WAR distribution':

```
"%JAVA_HOME%\bin\keytool" -genkey -alias tomcat -keyalg RSA
```

JIRA Installations on Linux/Solaris:

(Installed using the 'archive' or 'WAR distribution'.)

```
$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA
```

This will create (if it doesn't already exist) a new .keystore file located in the home directory of the user you used to run the keytool command.

2. Export the Certificate

You will now need to export the certificate to make it ready for importing into the Trust-store with the following command:

JIRA Installations on Windows:

- Installed using the 'Windows Installer':

```
"<jira-install-dir>\jre\bin\keytool" -export -alias tomcat -file file.cer
```

- Installed using the 'archive' or 'WAR distribution':

```
"%JAVA_HOME%\bin\keytool" -export -alias tomcat -file file.cer
```

JIRA Installations on Linux/Solaris:

(Installed using the 'archive' or 'WAR distribution'.)

```
$JAVA_HOME/bin/keytool -export -alias tomcat -file file.cer
```

3. Import the Certificate

See Import Certificate into the Trust-store (below) for details.

^Back to the flowchart

Obtain CA Certificate

Digital Certificate that are issued by trusted 3rd party CAs (Certification Authority) provide verification that your Website does indeed represent your company, thereby verifying your company's identity. Many CAs simply verify the domain name and issue the certificate, whereas other such as VeriSign verifies the existence of your business, the ownership of your domain name, and your authority to apply for the certificate, providing a higher standard of authentication.

A list of CA's can be found here.
Some of the most well known CAs are:

- Verisign
- Thawte
- CAcert (relatively new CA, providing free CA certificates)

Next, import the certificate into the Trust-store.

^Back to the flowchart

Import Certificate into the Trust-store

> Your SSL Vendor may have different instructions, please refer to them for proper certificate installation. Examples include GoDaddy and VeriSign

Assuming your certificate is called "file.cer" whether obtained by a CA or self-generated, the following command will add this certificate to the Trust-store:

JIRA Installations on Windows:

- Installed using the 'Windows Installer':

```
"<jira-install-dir>\jre\bin\keytool" -import -alias tomcat -file file.cer
```

- Installed using the 'archive' or 'WAR distribution':

```
"%JAVA_HOME%\bin\keytool" -import -alias tomcat -file file.cer
```

JIRA Installations on Linux/Solaris:

(Installed using the 'archive' or 'WAR distribution'.)

This step must be performed as the root user, or with the use of sudo

```
$JAVA_HOME/bin/keytool -import -alias tomcat -file file.cer
```

Next, proceed to the step on redirecting certain pages to HTTPS.

^Back to the flowchart

Configure HTTPS in Tomcat

Edit `conf/server.xml`, and at the bottom before the `</Service>` tag, add this section (or uncomment it where you find it) in Tomcat 6:

```
<Connector port="8443" maxHttpHeaderSize="8192" SSLEnabled="true"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true" useBodyEncodingForURI="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS" />
```

This enables SSL access on port 8443 (the default for HTTPS is 443, but just as Tomcat uses 8080 instead of 80 to avoid conflicts, 8443 is used instead of 443 here).

^Back to the flowchart

Redirecting certain pages to HTTPS

Although HTTPS is now activated and available, the old HTTP URLs (http://localhost:8080) are still available. In most situations one wants these URLs to continue working, but for some to redirect to their https equivalent. This is done by editing `WEB-INF/web.xml`, and adding the following section at the end of the file, before the closing `</web-app>`:

```
<security-constraint>
<web-resource-collection>
<web-resource-name>all-except-attachments</web-resource-name>
<url-pattern>*.jsp</url-pattern>
<url-pattern>*.jspa</url-pattern>
<url-pattern>/browse/*</url-pattern>
</web-resource-collection>
<user-data-constraint>
<transport-guarantee>CONFIDENTIAL</transport-guarantee>
</user-data-constraint>
</security-constraint>
```

This means that all URLs *except attachments* are redirected from HTTP to HTTPS. IE has a bug which prevents attachments like .doc files being viewed via HTTPS if SSL protection is forced in `web.xml`.

Once this change is made, restart JIRA and access http://localhost:8080. You should be redirected to https://localhost:8443/secure/Dashboard.jspa. The port it redirects to is determined by the `redirectPort` value you specify in the `server.xml` file in the HTTP Connector stanza.

There does not seem to be an easy way to make subsequent pages revert to HTTP after logging in via HTTPS - see JRA-7250

Users with IE8 might have problems with force HTTPS, please refer to JRA-27077 for further info.

Troubleshooting

Here are some troubleshooting tips if you are using a self-signed key created by keytool, as described above.

When you enter "https://localhost:8443" in your browser, if you get a message such as "Cannot establish a connection to the server at localhost:8443", look for error messages in your logs/catalina.out log file. Here are some possible errors with explanations:

SSL + Apache + IE problems

Some people have reported errors when uploading attachments over SSL using IE. This is due to an IE bug, and can be fixed in Apache by setting:

```
BrowserMatch ".MSIE." \
nokeepalive ssl-unclean-shutdown \
downgrade-1.0 force-response-1.0
```

Google has plenty more on this.

Can't find the keystore

```
java.io.FileNotFoundException: /home/user/.keystore (No such file or directory)
```

This indicates that Tomcat cannot find the keystore. The keytool utility creates the keystore as a file called `.keystore` in the current user's home directory. For Unix/Linux the home directory is likely to be `/home/<username>`. For Windows it is likely to be `C:\Documents And Settings\<UserName>`.

Make sure you are running JIRA as the same user who created the keystore. If this is not the case, or if you are running JIRA on Windows as a service, you will need to specify where the keystore file is in `conf/server.xml`. Add the following attribute to the connector tag you uncommented:

```
keystoreFile="<location of keystore file>"
```

Certificate reply and certificate in keystore are identical

```
keytool error: java.lang.Exception: Certificate reply and certificate in keystore are identical
```

This error will happen if you have identical names or fingerprints, which is the result of attempting to recreate the cert in your existing keystore. If you need to recreate or update the Cert, you may remove the existing keystore and creating a fresh, new keystore. In this case, creating a new keystore and adding the related certs will fix the issue. The default path for it in this documentation is $JAVA_HOME/jre/lib/security/cacerts

Incorrect password

```
java.io.IOException: Keystore was tampered with, or password was incorrect
```

You used a different password than "changeit". You must either use "changeit" for both the keystore password and for the key password for Tomcat, or if you want to use a different password, you must specify it using the `keystorePass` attribute of the Connector tag, as described above.

Passwords don't match

```
java.io.IOException: Cannot recover key
```

You specified a different value for the keystore password and the key password for Tomcat. Both passwords must be the same.

Wrong certificate

```
javax.net.ssl.SSLException: No available certificate corresponds to the SSL cipher suites which
are enabled.
```

If the Keystore has more than one certificate, Tomcat will use the first returned unless otherwise specified in the SSL Connector in `conf/server.xml`.

Add the `keyAlias` attribute to the Connector tag you uncommented, with the relevant alias, for example:

```
<Connector port="8443" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true" useBodyEncodingForURI="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="/opt/local/.keystore"
keystorePass="removed"
keyAlias="tomcat"/>
```

Using Apache Portable Runtime

APR uses a different SSL engine, and you will see an exception like this in your logs

```
SEVERE: Failed to initialize connector [Connector[HTTP/1.1-8443]]
LifecycleException:  Protocol handler initialization failed: java.lang.Exception: No
Certificate file specified or invalid file format
```

The reason for this is that the APR Connector uses OpenSSL and cannot use the keystore in the same way. You can rectify this in one of two ways:

Use the Http11Protocol to handle SSL connections

Edit the server.xml so that the SSL Connector tag you just uncommented specifies the Http11Protocol instead of the APR protocol

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"
  maxHttpHeaderSize="8192" SSLEnabled="true" keystoreFile="${user.home}/.keystore"
  maxThreads="150" enableLookups="false" disableUploadTimeout="true"
  acceptCount="100" scheme="https" secure="true"
  clientAuth="false" sslProtocol="TLS" useBodyEncodingForURI="true"/>
```

Configure the Connector to use the APR protocol

This is only possible if you have PEM encoded certificates and private keys. If you have used OpenSSL to generate your key, then you will have these PEM encoded files - in all other cases contact your certificate provider for assistance.

```
<Connector
  port="8443" maxThreads="200"
  scheme="https" secure="true" SSLEnabled="true"
  SSLCertificateFile="${user.home}/certificate.pem"
  SSLCertificateKeyFile="${user.home}/key.pem"
  clientAuth="optional" SSLProtocol="TLSv1"/>
```

Enabling Client Authentication

To enable client authentication in Tomcat, ensure that the value of the `clientAuth` attribute in your `Connector` element of your Tomcat's `server.xml` file is `true`.

```
<Connector
 ...
 clientAuth="true"
 ... />
```

For more information about `Connector` element parameters, please refer to the 'SSL Support' section of the Tomcat 6.0 or Tomcat 5.5 documentation.

Installing Confluence and JIRA Together

For information on Atlassian's recommendation on JIRA and Confluence installation, see Installing Confluence and JIRA Together.

You may also wish to read Integrating JIRA and Confluence for helpful information on integrating JIRA and Confluence.

Do not deploy multiple Atlassian applications in a single Tomcat container — Deploying multiple Atlassian applications in a single Tomcat container is **not supported.** We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration (see this FAQ for more information).

There are also a number of practical reasons why we do not support deploying multiple Atlassian applications in a single Tomcat container. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

Finally, we recommend not deploying *any other applications* in the same Tomcat container that runs JIRA, especially if these other applications have large memory requirements or require additional libraries in Tomcat's `lib` subdirectory.

Integrating JIRA with a Web Server

The following pages contain information on integrating JIRA with a web server.

- Integrating JIRA with IIS
- Integrating JIRA with Apache

Integrating JIRA with IIS

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian **can not guarantee providing any support for it**. Please be aware that this material is provided for your information only and using it is done so at your own risk.

This page describes how to configure Microsoft's IIS web server and JIRA such that IIS forwards requests on to JIRA, and responses back to the user. This is useful if you already have IIS running serving web pages (e.g. http://mycompany.com), and wish to integrate JIRA as just another URL (e.g. http://mycompany.com/jira).

JIRA is written in Java, and needs a Java Application Server (servlet container) to run. As IIS does not provide services of a Java Application Server, it is not possible to deploy JIRA directly into IIS. It is possible, however, to configure IIS to proxy requests for JIRA to an application server where JIRA is deployed. Therefore, if your main website is running in IIS, it is possible to integrate JIRA into this website.

If you need to integrate JIRA with IIS, JIRA needs to be deployed into a Java application server (such as Apache Tomcat), which provides IIS integration capability.

If you are running JIRA against an application server other than Apache Tomcat, please consult that application server's documentation to

determine whether it is possible (and how) to integrate the application server with IIS.

To integrate JIRA with IIS you will need to:

1. Configure JIRA and test that it works on its own
2. Configure Tomcat to accept proxied requests from IIS
3. Configure IIS to forward JIRA requests to Tomcat
4. *(Optional)* Configure IIS to forward Confluence requests to Tomcat (if you are using both Confluence and JIRA).

1. Configure JIRA

1. Follow the JIRA installation guide to install and configure JIRA; or deploy the WAR distribution into Apache Tomcat. Note that JIRA can be installed on the same machine as IIS, but this is not necessary.
2. Change the context path of the JIRA web application:
   To allow IIS to proxy requests to JIRA, JIRA web application must be deployed with a context path (e.g. the **/jira** in http://localhost:8080/jira (http://localhost:8080*/jira*)) in Tomcat. The context path **must** be set to the path in the URL that IIS will use to proxy requests. For example, if your website is running with address `www.example.com` in IIS, and you would like to make JIRA available under `www.example.com/jira`, you will need to set JIRA's context path to "`/jira`" in Tomcat.
   To do this, edit the `conf/server.xml` file (or the `jira.xml` file if you are using the WAR distribution of JIRA). Change the `path` attribute of the `Context` element to "`/jira`".

3. Restart JIRA after changing the context path.
4. Set the '**Base URL**' to include the context path (see Configuring JIRA Options).
5. Turn JIRA's GZip compression **OFF** (since there will be no benefit from GZip compression once proxying is implemented).
6. Test that JIRA works correctly by pointing your web browser directly at Tomcat (e.g. http://localhost:8080/jira) and going through JIRA's Setup Wizard. If you have completed the Setup Wizard previously, try creating an issue or editing one. Please ensure that no errors occur.

2. Configure Tomcat to accept proxied requests

HTTP/1.1 Connector

If you are using the HTTP/1.1 Connector, you will need to add the following attributes to the Connector port in Tomcat's **`server.xml`**:

```
proxyName="mycompany.com" proxyPort="80"
```

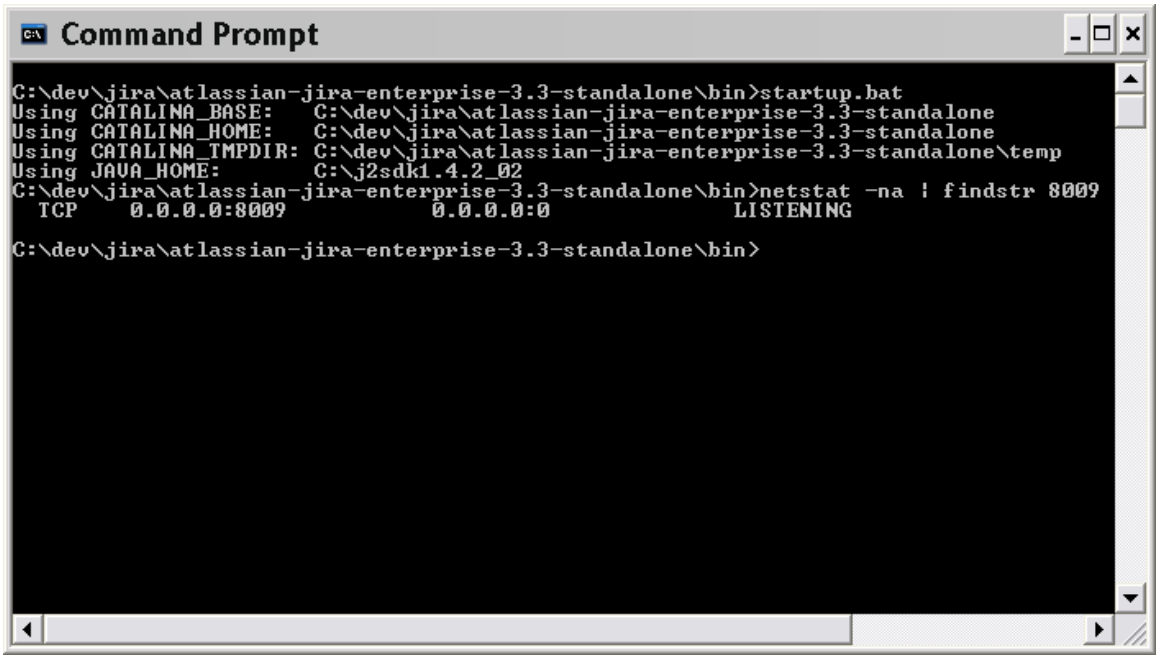
Please refer to the Integrating JIRA with Apache for reference.

1. Enable **AJP/1.3 Connector** in Tomcat: To allow Tomcat to accept requests for JIRA from IIS, edit the `conf/server.xml` file and ensure that the **AJP/1.3 Connector** is enabled (i.e. *not* commented out). To enable the AJP/1.3 Connector in a JIRA remove the comment symbols around the following section in the `conf/server.xml` file:

   ```
   <Connector port="8009" enableLookups="false" redirectPort="8443" protocol="AJP/1.3" />
   ```

   The above example configures Tomcat to listen for proxied IIS requests on port `8009`. If this port is already in use on the machine where JIRA is running, please change to another port.

2. Restart Tomcat and ensure that no errors regarding used ports appear in the logs or in the Tomcat Console.
3. Ensure that the AJP Connector is listening on the specified port (8009 by default). One way to do this is to use the "`netstat -na`" command in the command window and see if port 8009 is listed in the output:



3. Configure IIS to forward requests to JIRA

On the machine where IIS is deployed:

1. Download the ISAPI Redirect DLL from the Apache site. When downloading, choose the version of Windows that IIS is running on (either win32 or win64), and then **choose the latest available jk version**.

   > The file to download is named **isapi_redirect_X.X.X.dll,** where 'X.X.X' is the version number. You will need to remove the version number from the DLL file (i.e. it needs to be named isapi_redirect.dll).

2. Place the DLL and the associated properties files in an installation directory. For the purpose of this document, we will assume the directory is `C:\tomcat_iis_connector`. Place the **isapi_redirect.dll** in this directory. Then download the isapi_redirect.properties file and place this in the same directory as the **isapi_redirect.dll** file.
3. Create a directory called 'conf' in your installation directory (`C:\tomcat_iis_connector\conf`). Download the files uriworkermap.properties and workers.properties.minimal and place them in the `C:\tomcat_iis_connector\conf` directory.
4. Create a directory called 'logs' (`C:\tomcat_iis_connector\logs`). This is where the logs associated with the **isapi_redirect.dll** execution will be placed.
5. In the "`C:\tomcat_iis_connector`" directory you may need to modify the `isapi_redirect.properties` file. The `isapi_redirect.properties`file tells the connector where to find its configuration files and where the DLL can be found in relation to the IIS server. There are 5 properties in this file:
   a. extension_uri — the path to the virtual directory that contains the **isapi_redirect.dll**
   b. log_file — the path to write the log file to
   c. log_level — the level at which the logs should be generated
   d. worker_file — the path to your `workers.properties.minimal` file in your installation
   e. worker_mount_file — the path to your `uriworkermap.propertiesl` file in your installation.
   If you are installing the connector in `C:\tomcat_iis_connector` and you follow the instructions below about setting up the virtual directory for the **isapi_redirect.dll,** then you should not have to change any properties in the provided file.
6. In the "`C:\tomcat_iis_connector\conf`" directory you may need to modify the `uriworkermap.properties` and the `workers.properties.minimal`files.

   > The provided files contain the changes mentioned here and should work if you completely follow this document. **If you have deviated from this document, then you will need to modify these files as described below.**

   The `workers.properties.minimal` file tells IIS where (IP address and port) Tomcat is running. The `uriworkermap.properties` tells IIS what requests to proxy to Tomcat.
   To edit these files:

   a. Edit the `uriworkermap.propertiesand` ensure that it contains the following mapping for JIRA. You do not need any other mappings.

   ```
   /jira/*=worker1
   ```

The mapping (e.g. `/jira/`) ***must*** be the same as the context path that JIRA has been deployed with in Tomcat as described in the Configure JIRA section of this document.

b. Edit the `workers.properties.minimal` file and modify the `worker.ajp13w.host` property if necessary. This property should be set to the host name or the IP address of the machine where Tomcat (with JIRA) is running. If Tomcat is running on the same machine as IIS then you can leave the property set to `localhost`. If you have specified a host name as the value of this property, please ensure that the IIS machine can correctly resolve it to the appropriate IP address.
c. If you have modified the port for the AJP Connector you will need to modify the `worker.ajp13w.port`property. Here is an example of the file with Tomcat running on the same machine as IIS and using the default port (8009) for AJP:

```
worker.list=worker1

#
# Defining a worker named worker1 and of type ajp13.
# Note that the name and the type do not have to match.
#
worker.worker1.type=ajp13
worker.worker1.host=localhost
worker.worker1.port=8009
```

7. Open **Control Panel,** then **Administrative Tools** and open **Internet Information Services.**
8. **IIS 7.0 only:** If you are using **IIS 7.0,**you will need to install two required service roles, ISAPI Extensions and ISAPI Filters:
   a. Navigate to Start Menu > All Programs > Administration Tools > Service Manager.
   b. Select 'Web Server (IIS)' in Server Manager > Roles.
   c. Click 'Add Role Services' and follow the Wizard.
9. Add an **ISAPI Filter**to IIS, as described below:
   - **IIS 6.0 or earlier:**
     a. Right-click on **Default Web Site** (or the Web Site that should be responsible for proxying requests to JIRA), and click on **Properties.**
     b. Click the **ISAPI Filters** tab.
     c. Check if there is a Filter that points to the `isapi_redirect.dll` file and that it is in the right location. If not, click **Add** and create one. Enter `tomcat` as the Filter Name and enter the location of the `isapi_redirect.dll` file for the executable.
     d. Click **Apply** and then **OK.**
   - **IIS 7.0:**
     a. Click the **Default Web Site** (or the Web Site that should be responsible for proxying requests to JIRA), and click on **ISAPI Filters.**
     b. Click the **ISAPI Filters** icon.
     c. Check if there is a Filter that points to the `isapi_redirect.dll` file and that it is in the right location. If not, click **Add** and create one. Enter `tomcat` as the Filter Name and enter the location of the `isapi_redirect.dll` file.
     d. Click **OK.**
10. Create a **virtual directory**for JIRA in IIS.
    a. Right-click on **Default Web Site** (or the Web Site that should be responsible for proxying requests to JIRA), choose **New** and then **Virtual Directory.**
    b. Go through the creation wizard. Set the `alias` as the value of the Context Path (without slashes) that was set in the Configure JIRA section of this document (see above). In our example this is `jira` .
    c. This can point to any directory.
    d. Complete the wizard.

The reason for creating a virtual directory is so that requests without the trailing slash still work. For example, if you are deploying JIRA under `http://www.example.com/jira/` without the virtual directory, then requests to `http://www.example.com/jira` will fail.

11. Create a **virtual directory** for access to the **isapi_redirect.dll**in IIS, as described below:
    - **IIS 6.0 or earlier:**
      a. Right-click on **Default Web Site** (or the Web Site that should be responsible for proxying requests to JIRA), choose **New** and then **Virtual Directory.**
      b. Go through the creation wizard. Set the `alias` to be `jakarta` .
      c. This must point to the directory in which the **isapi_redirect.dll** is installed. In our example this is `C:\tomcat_iis_connector`.
      d. Complete the wizard, making sure that you grant the 'Execute' permission for the **Virtual Directory** by checking the 'Execute' checkbox.
    - **IIS 7.0:**
      a. Right-click on **Default Web Site** (or the Web Site that should be responsible for proxying requests to JIRA), and choose **Add Virtual Directory.**
      b. Set the `alias` to be `jakarta` .
      c. **Physical Path** must point to the directory in which the **isapi_redirect.dll** is installed. In our example this is `C:\tomcat_iis_connector`.

d. Click the 'jakarta' Virtual Directory and double-click 'Handler Mappings'.
e. Click 'Edit Feature Permissions' in the Action panel on the right-hand side.
f. Check the 'Execute' permission checkbox.

This Virtual Directory is needed for the connector to work. The alias that you give the directory needs to be the same as the path set in the `isapi_redirect.properties` file, `extension_uri` property. In our example this value is: `/jakarta/isapi_redirect.dll`.

12. If using IIS 6.0 or 7.0, you will need to add the dll as a **Web Service Extension,**as described below.
    - **IIS 6.0:**
        a. Right-click on **Web Service Extensions** and choose **Add a new Web Service Extension...**
        b. Enter `tomcat` for the **Extension Name** and then add the `isapi_redirect.dll` file to the required files.
        c. Select the **Set extension status to Allowed** check-box, then click **OK.**
    - **IIS 7.0:**
        a. Navigate to the servers and highlight your server.
        b. Navigate to 'ISAPI and CGI Restrictions'.
        c. Add and allow the isapi_redirect.dll extension.
13. You will need to restart the IIS Service. To do this, browse to **Control Panel,** click **Administrative Tools,** click on **Services,** find the IIS Admin Service and click **restart.**
14. You are done! To test the configuration, point your web browser at IIS and append JIRA's context path to the URL. For example, if your website is running under the address of `http://www.example.com` and you have deployed JIRA with the context path of `jira`, point your browser at `http://www.example.com/jira`.

4. Configure IIS to forward requests to Confluence as well as JIRA

You can configure IIS so that it forwards requests to both JIRA and Confluence.

The following instructions describe how to forward from IIS to separate instances of JIRA and Confluence, running in separate Tomcat servers. The instructions assume that you have already set up IIS to forward to JIRA as described in section 3 above. The instructions also assume that you have already installed Confluence as per the Confluence Installation Guide.

The instructions describe how to make JIRA available under `www.example.com/jira` as described above, and Confluence available under `www.example.com/confluence`.

1. If JIRA and Confluence are running on the same machine, ensure that Confluence is listening on a different port to JIRA:
    a. Edit the `conf/server.xml` file (or the `jira.xml` file if you are using the WAR/EAR distribution of Confluence).
    b. At the top of the file, change the `port` attribute of the `Server` element to a different port to the value for JIRA. For example, change it from `8005` to `8006`.
    c. Still in the `Server` element, Change the `port` attribute of the `Connector` sub-element to a different port to the value for JIRA. For example, change it from `8080` to `8090` .
2. Change the Confluence context path:
    a. Edit the `conf/server.xml` file `jira.xml` file (or the `jira.xml` file if you are using the WAR/EAR distribution of Confluence).
    b. Change the `path` attribute of the `Context` element to "`/confluence`".
3. Restart Confluence after changing the ports and the context path, and test that Confluence works correctly by pointing your web browser at http://localhost:8090/confluence.
4. Configure Confluence to accept proxied requests: Remove the comments around the AJP/1.3 Connector section in the Confluence `conf/server.xml` or `jira.xml` file and change the `port` attribute to a value different to the value for JIRA. For example, change it from `8009` to `8010`.
5. Restart Confluence and ensure that no errors regarding used ports appear in the logs or in the Tomcat console.
6. Edit the `uriworkermap.properties` file and add the following mapping:

```
/confluence/*=worker2
```

   The file should now contain the following mappings:

```
/jira/*=worker1
/confluence/*=worker2
```

7. Edit the `workers.properties.minimal` file:
   Change the line starting with `worker.list`to the following:

```
worker.list=worker1,worker2
```

   Add the following lines to the end of the file (assuming the host is on the same machine as IIS and you changed the AJP/1.3 Connector port for Confluence to `8010`):

```
worker.worker2.type=ajp13
worker.worker2.host=localhost
worker.worker2.port=8010
```

The `workers.properties.minimal`file should now look like the following:

```
worker.list=worker1,worker2

#
# Defining a worker named worker1 and of type ajp13.
# Note that the name and the type do not have to match.
#
worker.worker1.type=ajp13
worker.worker1.host=localhost
worker.worker1.port=8009

worker.worker2.type=ajp13
worker.worker2.host=localhost
worker.worker2.port=8010
```

8. Create a **virtual directory** for Confluence in IIS. Set the `alias` to `confluence`. It can point to any directory.
9. Restart the IIS Service.
10. You are done! Confluence should now be available under `www.example.com/confluence`, and JIRA should still be available under `www.example.com/jira`.

Troubleshooting

- ***Whenever I go to JIRA in my browser, a login panel pops up. I enter a valid username and password for JIRA, but the panel pops up again.***Make sure that you have Anonymous Access set on the `jira`virtual directory in IIS. It will be set to that if you have followed the above instructions.To check this:
  1. In **'Internet Information Services',** right click the `jira` virtual directory and choose **'Properties'.**
  2. Click the **'Directory Security'** tab.
  3. Click the **'Edit...'** button in the **'Anonymous access and authentication control'** section.
  4. Make sure that the **'Anonymous access'** tick box is selected, and make sure that nothing is selected in the **'Authenticated access'** section. Do not select **'Basic authentication'.** Do not select **'Integrated Windows authentication'.**

- ***Whenever I go to JIRA in Internet Explorer, a login panel pops up. I enter a valid username and password for JIRA, but the panel pops up again. This doesn't happen, however, in another browser such as Firefox or Safari. I can successfully log in to JIRA in those browsers.***Make sure that you have Internet Explorer's **User Authentication** set to **Anonymous login.**To check this:
  1. In Internet Explorer, click the **'Tools'** menu and select **'Internet Options'.**
  2. Click the **'Security'** tab.
  3. Select the security zone that the JIRA server is in.
  4. Click the **'Custom level...'** button.
  5. Scroll right down to the bottom to the **'User Authentication'** section.
  6. Select **'Anonymous logon'** (if it is not already selected).
  7. Click the **'OK'** button on this screen, and again on the next screen.
  8. Restart Internet Explorer.

- ***When I try to navigate to my JIRA instance at http://localhost/jira in my browser, it prompts me to download a file with nonsensical information, rather than showing me my JIRA instance.***Make sure that you have granted the 'Execute' permission to your Virtual Directory for JIRA in IIS. See step 11 of the '3. Configure IIS to forward requests to JIRA' section in this document for detailed instructions.

Known Issues

- **64 bit IIS:** If you are running a 64 bit OS, please use a 64 bit version of the Tomcat IIS connector.
- **Customer submitted solution:** If you must use a 32 bit IIS connector, you can do so by clicking `Application Pools > Advanced Settings > Allow 32bit applications`.
- **Customer submitted solution:** You need to set the ISAPI extension on the website.

Integrating JIRA with Apache

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian **can not guarantee providing any support for it**. Please be aware that this material is provided for your information only and using it is done so at your own risk.

This page describes how to integrate an Apache HTTP Server (httpd) with JIRA (via mod_proxy), such that 'httpd' forwards requests on to JIRA and responses back to the user. This is useful if you already have 'httpd' serving web pages on port 80 (e.g. http://mycompany.com), and wish to integrate JIRA as just another URL (eg. http://mycompany.com/jira).

You can only configure JIRA to respond to a single URL, and your Base URL setting must match the URL end users are requesting. You cannot (for example) have a different hostname or URL for internal and external users. This is especially important for JIRA 4.0 or higher, as any mismatch between the Base URL and the URL requested by the end user will cause problems with dashboard gadgets.

This documentation describes a straightforward implementation of mod_proxy. If you require a more complex solution, refer to Apache HTTP Server Version Documentation and, if necessary, consult with someone in your organisation who is knowledgeable in the configuration of 'httpd'.

- Step 1: Configure JIRA's application server
- Step 2: Configure 'httpd'
  - Enable mod_proxy and mod_proxy_http
  - Configure mod_proxy
  - Terminating an SSL connection at 'httpd'
- Troubleshooting
- See Also

Step 1: Configure JIRA's application server

Here we assume you are using the 'recommended' JIRA distribution (i.e. not JIRA WAR), which includes Tomcat. First, we need to edit Tomcat's **conf/server.xml** file, and set the context path:

```
<Server port="8005" shutdown="SHUTDOWN">

        <Context path="/jira" docBase="${catalina.home}/atlassian-jira" reloadable="false">
          <Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"
            ....
```

Here we have set the context path to **/jira,** assuming JIRA will be running on http://mycompany.com/jira/.

Restart Tomcat, and ensure you can still access JIRA normally (eg. at `http://localhost:8080/jira/`).

Note: if you want Tomcat responsible for all URLs, specify a blank context path with `path=""` — *not* `path="/"` .

Turn JIRA's GZip compression **OFF** (since there will be no benefit from GZip compression once proxying is implemented, and in fact GZIP has been reported to cause performance problems in this situation).

Step 2: Configure 'httpd'

Assuming an Apache HTTP Server version 2 installation, the following needs to be done:

Enable mod_proxy and mod_proxy_http

The exact steps will be specific to your operating system. Refer to the Apache documentation for your operating system. On Debian/Ubuntu it is done as follows:

```
teacup:/etc/apache2# a2enmod proxy_http
Enabling proxy as a dependency
Module proxy installed; run /etc/init.d/apache2 force-reload to enable.
Module proxy_http installed; run /etc/init.d/apache2 force-reload to enable.
teacup:/etc/apache2#
```

Configure mod_proxy

Here we create a config snippet for JIRA, in **sites-available/jira-mod_proxy:**

```
teacup:/etc/apache2# cd sites-available

teacup:/etc/apache2/sites-available# cat > jira-mod_proxy

<Proxy *>
Order deny,allow
Allow from all
</Proxy>

ProxyRequests       Off
ProxyPreserveHost On
ProxyPass           /jira       http://localhost:8080/jira
ProxyPassReverse    /jira       http://localhost:8080/jira

teacup:/etc/apache2/sites-available#
teacup:/etc/apache2/sites-available# a2ensite jira-mod_proxy
Site jira-mod_proxy installed; run /etc/init.d/apache2 reload to enable.

teacup:/etc/apache2/sites-available# /etc/init.d/apache2 reload
Reloading apache 2.0 configuration....
teacup:/etc/apache2/sites-available#
```

JIRA should now be integrated with 'httpd'. You should be able to view JIRA at `http://localhost/jira` (i.e. on port 80).

Terminating an SSL connection at 'httpd'

To add an SSL connection that terminates at 'httpd', using HTTP to connect to JIRA behind it, most of the relevant configuration is:

```
Listen 443

NameVirtualHost *:443
<VirtualHost *:443>
    SSLEngine On
    SSLCertificateFile /etc/apache2/ssl/apache.pem
    ProxyPass / http://localhost:8080/
    ProxyPassReverse / http://localhost:8080/
</VirtualHost>
```

Notes:

- The path '/jira' **must** be the same as the context path in Tomcat's conf/server.xml
- The **ProxyPreserveHost** directive allows Tomcat to know its public hostname and port. Without this, JIRA would redirect the public URL (e.g. `http://mycompany.com/jira/`) to `http://localhost:8080/jira/secure/Dashboard.jspa`.

If the links for Printable Version, RSS feeds, Word export and Excel export have incorrect URLs, starting with `localhost:8080/jira` instead of `http://mycompany.com/jira` , ensure that **ProxyPreserveHost** is set to **On**.

ProxyPreserveHost is only available on Apache HTTP Server version 2. For versions 1.1-1.3.x, you should instead specify **proxyName** and **proxyPort** attributes in Tomcat as follows:

```
<Server port="8005" shutdown="SHUTDOWN">

  <Service name="Catalina">

    <Connector port="8080"
      maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
      enableLookups="false" redirectPort="8443" acceptCount="100" connectionTimeout="20000"
disableUploadTimeout="true"
      proxyName="mycompany.com" proxyPort="80" />

    <Engine name="Catalina" defaultHost="localhost">
      <Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true">

        <Context path="/jira" docBase="${catalina.home}/atlassian-jira" reloadable="false">
          <Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"
            ....
```

If you are using Apache HTTP Server version 1.x, make sure you **don't** use caching (CacheRoot directive).

Some users have reported problems with user sessions being hijacked when the mod_cache module is enabled. If you have such problems, disable the **mod_cache** module. Note that this module is enabled by default in some Apache HTTP Server version 2 distributions.

Troubleshooting

- On **Fedora Core 4,** people have reported 'permission denied' errors when trying to get mod_proxy (and mod_jk) working. Disabling SELinux (`/etc/selinux/config`) apparently fixes this.
- If you are on Mac OS X:
  - If your gadgets are returning 404 errors, you may need to add the following to Tomcat's <Connector> tag (use port 443 instead of 80 if you're using SSL and terminating it at 'httpd'):

    ```
    proxyName="your-front-facing-domain.com" proxyPort="80"
    ```

  - Please disable **webperfcache,**which proxies port 80 by default. A user reported this as the likely cause of JIRA session problems, in the form of users' identities becoming mixed up:

    ```
    The OSX Servers enable webperfcache by default for Virtual Hosts, which for static
    content
            would be great, but for dynamic sites (which ALL of ours are) it is Evil
    and causes many
            issues.  Of note recently was the jira session issue. Also see :-

    http://developer.apple.com/documentation/Darwin/Reference/ManPages/man8/webperfcache.8
    http://www.wodeveloper.com/omniLists/webobjects-admin/2005/January/msg00009.html
            Unfortunately even if you disable webperfcache for a site, if there is a
    single site enabled
            then all sites will still proxy through webperfcache with resulting
    session problems.
    ```

- In general, if you are having problems:
  1. Ensure that JIRA works as expected when running directly from Tomcat on http://localhost:8080/jira
  2. Watch the log files (usually in /var/log/httpd/ or /var/log/apache2/). Check that you have a **LogLevel** directive in your httpd.conf, and turn up logging **('LogLevel debug')** to get more info.
  3. Check out the Knowledge Base .

See Also

- Integrating JIRA with Apache using SSL
- Configuring Apache Reverse Proxy Using the AJP Protocol
- For more advanced mod_webapp configurations (eg. SSL), see this mod_proxy guide.

Configuring Apache Reverse Proxy Using the AJP Protocol

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian **can not guarantee providing any support for it**. Please be aware that this material is provided for your information only and using it is done so at your own risk.

- Step 1: Configure JIRA's application server
- Step 2: Configure the Apache server
  - Using Apache 2.0.x
  - Using Apache 2.2.x
- See Also

Step 1: Configure JIRA's application server

Enable the AJP Connector on the Tomcat container hosting JIRA by uncommenting the following element in $JIRA_HOME/conf/server.xml:

```
<Connector port="8009" URIEncoding="UTF-8" enableLookups="false" redirectPort="8443"
protocol="AJP/1.3" />
```

Step 2: Configure the Apache server

Using Apache 2.0.x

1. Apache 2.0.x users must install `mod_jk` and configure it by adding the following to `httpd.conf` or included files:

```
LoadModule jk_module modules/mod_jk.so
JkWorkersFile conf/workers.properties
JkLogFile logs/mod_jk.log
JkLogLevel info
```

2. Add the following lines inside the `VirtualHost` tag, presenting JIRA to the userbase:

```
JkMount /jira jira_worker
JkMount /jira/* jira_worker
```

3. Define the connections to the JIRA application in the `conf/workers.properties` file:

```
worker.jira_worker.host=JIRA_HOST_NAME
worker.jira_worker.port=AJP_CONNECTOR_PORT (usually 8009)
worker.jira_worker.type=ajp13
```

Using Apache 2.2.x

Apache 2.2.x users can opt for native protocol support using `mod_proxy_ajp`.

1. Enable `mod_proxy` and `mod_proxy_ajp` submodule for protocol support in `httpd.conf`:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
```

2. Add the following in the `VirtuaHost` presenting JIRA:

```
ProxyRequests Off

<Proxy *>
Order deny,allow
Allow from all
</Proxy>

ProxyPass / ajp://JIRA_HOST_NAME:AJP_CONNECTOR_PORT/
ProxyPassReverse / ajp://JIRA_HOST_NAME:AJP_CONNECTOR_PORT/
```

See Also

- Integrating JIRA with Apache
- Integrating JIRA with Apache using SSL

Integrating JIRA with Apache using SSL

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian **can not guarantee providing any support for it**. Please be aware that this material is provided for your information only and using it is done so at your own risk.

This page describes using an SSL connection between Apache HTTP Server (httpd) and Tomcat, which is not a common configuration. This connection is usually unnecessary as it is behind the firewall and the SSL connection can terminate on 'httpd', and use an HTTP to connect to Tomcat. For information on integrating JIRA with 'httpd' *without* SSL, use the Integrating JIRA with Apache documentation. For the specific configuration of terminating the SSL connection at 'httpd', find the "Terminating an SSL connection at Apache" section.

If you want to use https (e.g. `https://mycompany.com/jira/`), then:

- Step 1. In 'httpd', ensure SSLProxyEngine is on
- Step 2. Configure Tomcat to use SSL ('recommended' distributions of JIRA)
- Step 3. Import 'httpd's public SSL key into Tomcat's keystore
  - Obtain the server's public key:
  - Import the public key
- Step 4. Restart the app server
- Note: Alternative keystore locations
- Note: Alternative configuration if HTTPS is terminated on the proxy server

Step 1. In 'httpd', ensure SSLProxyEngine is on

- In the 'httpd' config (`/etc/apache2/sites-available/jira-mod_proxy`), ensure you have **SSLProxyEngine on** specified, and redirect `/jira` to `https://localhost:8443/jira`:

```
<Proxy *>
Order deny,allow
Allow from all
</Proxy>

SSLProxyEngine on
ProxyRequests       Off
ProxyPreserveHost On
ProxyPass           /jira       https://localhost:8443/jira
ProxyPassReverse    /jira       https://localhost:8443/jira
```

- Please ensure that the ProxyPass and ProxyPassReverse directives do not include a trailing '/'. There have been reports that this may cause problems in JIRA 3.7 and above when serving static resources (javascript and css).

Step 2. Configure Tomcat to use SSL ('recommended' distributions of JIRA)

Edit `conf/server.xml`, and at the bottom before the </Service> tag, add this section (or uncomment it where you find it):

```
<Connector port="8443" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true" useBodyEncodingForURI="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS" />
```

This enables SSL access on port 8443 (the default for https is 443, but just as Tomcat uses 8080 instead of 80 to avoid conflicts, 8443 is used instead of 443 here).

Step 3. Import 'httpd's public SSL key into Tomcat's keystoreObtain the server's public key:

To quote Microsoft; "consult your system administrator". The public/private key pair will live somewhere on the server. The public key should be located and copied to the server hosting JIRA/Confluence. For example:

```
scp root@mail.yourcompany.com:/etc/ssl/certs/httpd.pem .
```

If you have openssl installed locally, the key can be retrieved with a command like:

```
donna-mcgahans-macbook-pro:~ dmcgahan$ openssl s_client -connect support.atlassian.com:https
CONNECTED(00000003)
depth=1 /C=US/O=DigiCert Inc/OU=www.digicert.com/CN=DigiCert Global CA
verify error:num=20:unable to get local issuer certificate
verify return:0
---
Certificate chain
 0 s:/C=au/ST=NSW/L=Sydney/O=ATLASSIAN SOFTWARE SYSTEMS PROPRIETARY
LIMITED/OU=IT/CN=*.atlassian.com
   i:/C=US/O=DigiCert Inc/OU=www.digicert.com/CN=DigiCert Global CA
 1 s:/C=US/O=DigiCert Inc/OU=www.digicert.com/CN=DigiCert Global CA
   i:/C=US/O=Entrust.net/OU=www.entrust.net/CPS incorp. by ref. (limits liab.)/OU=(c) 1999
Entrust.net Limited/CN=Entrust.net Secure Server Certification Authority
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIGYDCCBUigAwIBAgIQCi1wR9xdR7qYjJaF4e+4YDANBgkqhkiG9w0BAQUFADBc
MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMRkwFwYDVQQLExB3
d3cuZGlnaWNlcnQuY29tMRswGQYDVQQDExJEaWdpQ2VydCBHbG9iYWwgQ0EwHhcN
MDgwMTEwMDAwMDAwWhcNMTEwMTEzMjM1OTU5WjCBjDELMAkGA1UEBhMCYXUxDDAK
BgNVBAgTA05TVzEPMA0GA1UEBxMGU3lkbmV5MTcwNQYDVQQKEy5BVExBU1NJQU4g
U09GVFdBUkUgU1lTVEVNUyBQUk9QUklFVEFSWSBMSU1JVEVEMQswCQYDVQQLEwJJ
VDEYMBYGA1UEAxQPKi5hdGxhc3NpYW4uY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GN
ADCBiQKBgQDKjT2WNJaRLC2q/QEndjdVtriS/qMQfeX+sXgz4tSN+jd1zupOzuDo
xUfTilVLIt8aR5/bSa+XY3ykj5RcNRxki7Q/rr30FANY3cKCxY2TYZjVoPYVipnW
VDubtpjvUywE6E5LwI33oFqqnhL+HzEOioXOBHdU2/tZHj8n0VR7hQIDAQABo4ID
bzCCA2swHwYDVR0jBBgwFoAUp8cToHoBPJ3vgkiCSNVzUbYSViowHQYDVR0OBBYE
FOibDc5A2xBHAf8MBqnaEFQJswQBMCkGA1UdEQQiMCCCDyouYXRsYXNzaWFuLmNv
bYINYXRsYXNzaWFuLmNvbTB2BggrBgEFBQcBAQRqMGgwJAYIKwYBBQUHMAGGGGh0
dHA6Ly9vY3NwLmRpZ2ljZXJ0LmNvbTBABggrBgEFBQcwAoY0aHR0cDovL3d3dy5k
aWdpY2VydC5jb20vQ0FDZXJ0cy9EaWdpQ2VydEdsb2JhbENBLmNydDAOBgNVHQ8B
Af8EBAMCBaAwDAYDVR0TAQH/BAIwADB/BgNVHR8EeDB2MDmgN6A1hjNodHRwOi8v
Y3JsMy5kaWdpY2VydC5jb20vRGlnaUNlcnRHbG9iYWxDQS0yMDA4YS5jcmwwOaA3
oDWGM2h0dHA6Ly9jcmw0LmRpZ2ljZXJ0LmNvbS9EaWdpQ2VydEdsb2JhbENBLTIw
MDhhLmNybDCCAcYGA1UdIASCAb0wggG5MIIBtQYLYIZIAYb9bAEDAAEwggGkMDoG
CCsGAQUFBwIBFi5odHRwOi8vd3d3LmRpZ2ljZXJ0LmNvbS9zc2wtY3BzLXJlcG9z
aXRvcnkuaHRtMIIBZAYIKwYBBQUHAgIwggFWHoIBUgBBAG4AeQAgAHUAcwBlACAA
bwBmACAAdABoAGkAcwAgAEMAZQByAHQAaQBmAGkAYwBhAHQAZQAgAGMAbwBuAHMA
dABpAHQAdQB0AGUAcwAgAGEAYwBjAGUAcAB0AGEAbgBjAGUAIABvAGYAIAB0AGgA
ZQAgAEQAaQBnAGkAQwBlAHIAdAAgAEMAUAAvAEMAUABTACAAYQBuAGQAIAB0AGgA
ZQAgAFIAZQBsAHkAaQBuAGcAIABQAGEAcgB0AHkAIABBAGcAcgBlAGUAbQBlAG4A
dAAgAHcAaABpAGMAaAAgAGwAaQBtAGkAdAAgAGwAaQBhAGIAaQBsAGkAdAB5ACAA
YQBuAGQAIABhAHIAZQAgAGkAbgBjAG8AcgBwAG8AcgBhAHQAZQBkACAAaABlAHIA
ZQBpAG4AIABiAHkAIAByAGUAZgBlAHIAZQBuAGMAZQAuMB0GA1UdJQQWMBQGCCsG
AQUFBwMBBggrBgEFBQcDAjANBgkqhkiG9w0BAQUFAAOCAQEAspPrcCoRqI94BaPB
vujILnWqhnAjGp9QAI08YKNtAXp6X65Ytl48f3VOLivqCwVesm7FM7lXpFf46Kbj
9kfii/003x8+0rJo34lJcTIPO0EEu1tbvHKDcueII16g8Sfnpm9xZNi8imVunB6K
r9ID9Bl+ROl3u9wf6JgYIVeMxMD8lGKqCckjOimErIuB3Ca/A+L6+8eAp0/Y0yyE
z7cCI7kllKdjTvu5Y/GoN/cyBYKv57LeUrrNr7uMuyk0TJq0bFUl4KRMY6u3Rihe
zYNouvdneLKqlOwk4tBPODGm6LN0ubQc9C3J4pkrHhzEGXsEnk21O9syQ7ym9/1B
5++R3Q==
-----END CERTIFICATE-----
```

Cut and paste the certificate (including BEGIN and END lines) into a local file (eg. httpd.pem).

Import the public key

To do this, you need to use the `keytool` program that comes with Java. If you haven't already, add `$JAVA_HOME/bin` to your PATH, and then run the following:

```
jturner@teacup:~$ sudo keytool -import -alias mail.yourcompany.com -keystore
$JAVA_HOME/jre/lib/security/cacerts -file imapd.pem
Enter keystore password:  changeit
Owner: EMAILADDRESS=info@atlassian.com, CN=atlassian.com, O=Atlassian, L=Sydney, ST=NSW, C=AU
Issuer: EMAILADDRESS=info@atlassian.com, CN=atlassian.com, O=Atlassian, L=Sydney, ST=NSW, C=AU
Serial number: 0
Valid from: Fri Feb 11 14:09:05 EST 2005 until: Sat Feb 11 14:09:05 EST 2006
Certificate fingerprints:
MD5:  CB:AE:7D:5D:1A:08:06:77:93:3B:0F:53:BB:40:C0:D4
SHA1: 7C:02:44:0D:A9:8F:F9:FB:BB:7B:C6:F1:52:DE:CA:00:17:D9:3A:A0
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

This will import the public key (imapd.pem) into Java's default keystore, and marks it as trusted.

On Windows the command is similar, eg.:

```
C:\Program Files\Java\jre1.6.0_05>bin\keytool -import -file c:\certs\imapd.pem -alias
mail.yourcompany.com -keystore lib\security\cacerts
Enter keystore password:
Owner: CN=*.atlassian.com, OU=IT, O=ATLASSIAN SOFTWARE SYSTEMS PROPRIETARY LIMITED, L=Sydney,
ST=NSW, C=au
Issuer: CN=DigiCert Global CA, OU=www.digicert.com, O=DigiCert Inc, C=US
Serial number: a2d7047dc5d47ba988c9685e1efb860
Valid from: Thu Jan 10 11:00:00 EST 2008 until: Fri Jan 14 10:59:59 EST 2011
Certificate fingerprints:
         MD5:  9D:B4:9F:3D:0A:DE:6A:BD:BC:3D:95:BE:60:BD:70:02
         SHA1: 67:C6:E9:C8:3F:F1:7A:3C:66:E2:CE:62:78:A1:66:84:35:5E:62:1E
         Signature algorithm name: SHA1withRSA
         Version: 3
.....

Trust this certificate? [no]:  yes
Certificate was added to keystore

C:\Program Files\Java\jre1.6.0_05>
```

Step 4. Restart the app server

Restart, and if everything is correct, your webapp should now connect to the SSL resource without problems.

Note: Alternative keystore locations

Java will normally use a system-wide keystore in `$JAVA_HOME/jre/lib/security/cacerts`, but it is possible to use a different keystore by specifying a parameter, **-Djavax.net.ssl.trustStore=/path/to/keystore**, where '/path/to/keystore' is the absolute file path of the alternative keystore.

Setting this **is not recommended**, however, because if Java is told to use a custom keystore (eg. containing a self-signed certificate), then Java will not have access to the root certificates of signing authorities found in `$JAVA_HOME/jre/lib/security/cacerts`, and accessing most CA-signed SSL sites will fail. It is better to add new certificates (eg. self-signed) to the system-wide keystore (as above).

There is also a per-user truststore (~/.keystore) but (at least on Linux), but its contents do not appear to be logically appended to those in the system-wide keystore; ie. it is entirely separate, and only used if one specifies `-Djavax.net.ssl.trustStore=/home/<user>/.keystore`. This has the same disadvantage described above with custom keystores, so the per-user truststore is best avoided.

Note: Alternative configuration if HTTPS is terminated on the proxy server

If HTTPS is terminated on the proxy server, i.e.:

```
Client Browser --> HTTPS --> Apache proxy --> HTTP --> Tomcat/JIRA
```

then you will need to configure steps 1 and 2 slightly differently.

Specifically a HTTP Connector needs to be defined (identical to the default 8080 Connector) with the addition of the following attributes:
`scheme="https", proxyName="<proxy_server>", proxyPort="<proxy_port>"`

Default connector:

```
<Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443"
               URIEncoding="UTF-8"
               useBodyEncodingForURI="true"
 />
```

Connector that supports HTTPS terminated on the proxy server:

```
<Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443"
               URIEncoding="UTF-8"
               useBodyEncodingForURI="true"

               <!-- The below are new lines to add - the above is untouched -->
               scheme="https"
               proxyName="<proxy_server>"
               proxyPort="443"
 />
```

In this scenario, the 'httpd' `httpd.conf` file needs to be modified from:

```
ProxyPass             /jira       https://localhost:8443/jira
ProxyPassReverse      /jira       https://localhost:8443/jira
```

to

```
ProxyPass             /jira       http://localhost:8080/jira
ProxyPassReverse      /jira       http://localhost:8080/jira
```

(Note the changes to the scheme and port).

Securing JIRA with Apache HTTP Server

The following outlines some basic techniques to secure a JIRA instance using Apache HTTP Server. These instructions are basic to-do lists and should not be considered comprehensive. For more advanced security topics see the "Further Information" section below.

- Using Apache to Limit Access to the JIRA Administration Interface
- Using Fail2Ban to limit login attempts (JIRA 4.1 has login-rate limiting, but Fail2Ban can be useful for older versions and more advanced security setups.)

Further information

- Integrating JIRA with Apache

Using Apache to Limit Access to the JIRA Administration Interface

Limiting Administration to Specific IP Addresses

The JIRA administration interface is a critical part of the application; anyone with access to it can potentially compromise not only the JIRA instance but the entire machine. As well as limiting access to users who really need it, and using strong passwords, you should consider limiting access to it to certain machines on the network or internet. If you are using an Apache HTTP Server, this can be done with Apache's **Location** functionality as follows.

1. Create a file that defines permission settings

This file can be in the Apache configuration directory or in a system-wide directory. For this example we'll call it "sysadmin_ips_only.conf". This file should contain the following:

```
Order Deny,Allow
 Deny from All

 # Mark the Sysadmin's workstation
 Allow from 192.168.12.42
```

2. Add the file to your Virtual Host

In your Apache Virtual Host, add the following lines to restrict the administration actions to the Systems Administrator:

```
<LocationMatch Administrators.jspa>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteAttachment>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AcknowledgeTask>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ActivateWorkflow>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ActivateWorkflowStep2>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddIssueSecurity>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddIssueSecurityScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddLevel>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddNotification>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddNotificationScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddPermission>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddPermissionScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddPopMailServer>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddProject>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddProjectCategory>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddRepository>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddSmtpMailServer>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddUser>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowSchemeEntity>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransition>
```

```
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransitionCondition>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransitionConditionParams>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransitionFunctionParams>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransitionPostFunction>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransitionValidator>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AddWorkflowTransitionValidatorParams>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AssociateFieldToScreens>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AssociateIssueTypeSchemes>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch AssociateIssueTypeSchemesWithDefault>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch BugzillaImport>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch BulkEditUserGroups>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CloneWorkflow>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureCache>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureCsvMapping>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureCustomField>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureFieldLayout>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureFieldLayoutScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureFieldScreen>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureFieldScreenScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureFogBugzMapping>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureIssueTypeScreenScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureLogging>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ConfigureOptionSchemes>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CopyFieldLayout>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CopyFieldLayoutScheme>
```

```
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CopyIssueSecurityScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CopyNotificationScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CopyPermissionScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CopyWorkflowScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CreateCustomField>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CreateDraftWorkflow>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CsvImporter>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch CurrentUsersList>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteCustomField>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteGroup>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteIssueSecurity>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteIssueSecurityLevel>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteIssueSecurityScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteIssueType>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteLinkType>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteMailServer>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteNotification>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteNotificationScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteOptionScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeletePermission>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeletePermissionScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeletePriority>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteProject>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteProjectCategory>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteProjectRole>
```

```
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteRepository>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteResolution>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteStatus>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteSubTaskIssueType>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteTrustedApplication>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteUser>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteUserProperty>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowSchemeEntity>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowStep>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowTransitionCondition>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowTransitionPostFunction>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowTransitions>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DeleteWorkflowTransitionValidator>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch DisableSubTasks>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditAnnouncementBanner>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditApplicationProperties>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditAttachmentSettings>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditBasicConfig>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditCustomField>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditCustomFieldDefaults>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditCustomFieldOptions>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditDefaultFieldLayoutItem>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldLayout>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldLayoutItem>
```

```
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldLayoutItemRenderer>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldLayoutItemRendererConfirmation>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldLayoutScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldScreen>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldScreenScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditFieldScreenSchemeItem>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditIssueSecurities>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditIssueSecurityScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditIssueType>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditIssueTypeScreenScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditLinkType>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditListener>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditLookAndFeel>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditNotifications>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditNotificationScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditPermissions>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditPermissionScheme>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditPriority>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditProjectCategory>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditProjectRole>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditResolution>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditService>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditStatus>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditSubTaskIssueTypes>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch EditTrustedApplication>
```

```
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditUser>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditUserDefaultSettings>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditUserGroups>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditUserProjectRoles>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditUserProperties>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditUserProperty>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflow>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowSchemeEntities>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowStep>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowTransition>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowTransitionConditionParams>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowTransitionPostFunctionParams>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EditWorkflowTransitionValidatorParams>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch EnterpriseSelectProjectRepository>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ExternalImport>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch FogBugzImport>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch GlobalPermissions>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch GroupBrowser>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ImportWorkflowFromXml>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch IndexAdmin>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch IndexOptimize>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch IntegrityChecker>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch JellyRunner>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch JiraSupportRequest>
```

```
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch LDAPConfigurer>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ListEventTypes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ListWorkflows>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch MailQueueAdmin>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch MakeDefaultLevel>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ManageConfiguration>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ManageConfigurationScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ManageIssueTypeSchemes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ManageSubTasks>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch MantisImport>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch MigrateIssueTypes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectEmail>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportBackupOverviewProgress>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportMappingProgress>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportMissingMandatoryUsersCannotCreate>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportMissingMandatoryUsersExtMgmt>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportMissingOptionalUsersCannotCreate>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportMissingOptionalUsersExtMgmt>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportMissingUsersAutoCreate>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportProgress>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportResults>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportSelectBackup>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportSelectProject>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ProjectImportSummary>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch PublishDraftWorkflow>
```

```
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch RepositoryTest>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ResetFailedLoginCount>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchedulerAdmin>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeComparisonPicker>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeComparisonTool>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeGroupToRoleMapper>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeGroupToRoleResult>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeGroupToRoleTransformer>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeMerge>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeMergePreview>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeMergeResult>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemePicker>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemePurgeToolPreview>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemePurgeToolResults>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemePurgeTypePicker>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeTools>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SchemeTypePicker>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectFieldLayoutScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectIssueTypeSchemeForProject>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectIssueTypeScreenScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectCategory>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectIssueSecurityScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectPermissionScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectRepository>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectScheme>
```

```
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectSecuritySchemeStep2>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectWorkflowScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectWorkflowSchemeStep2>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectProjectWorkflowSchemeStep3>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SelectScreenScheme>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SendBulkMail>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SendTestMail>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ServiceExecutor>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SetGlobalEmailPreference>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch SetPassword>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch TaskAdmin>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch TimeTrackingAdmin>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch TrackbackAdmin>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch UpdatePopMailServer>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch UpdateRepository>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch UpdateSmtpMailServer>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch UserBrowser>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewApplicationProperties>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewAttachmentSettings>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewCustomFields>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewDefaultProjectRoleActors>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewFieldLayouts>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewFieldLayoutSchemes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewFieldScreens>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewFieldScreenSchemes>
```

```
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewGroup>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewIssueColumns>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewIssueFields>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewIssueSecuritySchemes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewIssueTypes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewIssueTypeScreenSchemes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewLicense>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewLinkTypes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewListeners>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewLogging>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewLookAndFeel>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewMemoryInfo>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewNotificationSchemes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewPermissionSchemes>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewPlugins>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewPriorities>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewProjectCategories>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewProjectRoles>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewProjectRoleUsage>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewResolutions>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewServices>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewStatuses>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewSystemInfo>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewTranslations>
    Include sysadmin_ips_only.conf
  </LocationMatch>
  <LocationMatch ViewTrustedApplications>
```

```
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewUpgradeHistory>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewUser>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewUserDefaultSettings>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewUserProjectRoles>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowSchemes>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowStep>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowStepMetaAttributes>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowSteps>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowTransition>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowTransitionConditionalResult>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowTransitionMetaAttributes>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch ViewWorkflowXml>
  Include sysadmin_ips_only.conf
</LocationMatch>
<LocationMatch XmlBackup>
  Include sysadmin_ips_only.conf
</LocationMatch>
```

```
<LocationMatch XmlRestore>
  Include sysadmin_ips_only.conf
</LocationMatch>
```

Using Fail2Ban to limit login attempts

JIRA 4.1 includes a rate-limiting mechanism, but older versions and other applications such as Confluence need external help from a tool such as Fail2Ban.

What is Fail2Ban?

We need a means of defending sites against brute-force login attempts. Fail2Ban is a Python application which trails logfiles, looks for regular expressions and works with Shorewall (or directly with iptables) to apply temporary blacklists against addresses that match a pattern too often. This can be used to limit the rate at which a given machine hits login URLs for Confluence.

*The information on this page does not apply to **Confluence OnDemand**.*

Prerequisites

- Requires Python 2.4 or higher to be installed
- Needs a specific file to follow, which means your Apache instance needs to log your Confluence access to a known logfile. You **should adjust the configuration below** appropriately.

How to set it up

This list is a skeletal version of the instructions

- There's an RPM available for RHEL on the download page, but you can also download the source and set it up manually
- Its configuration files go into `/etc/fail2ban`
- The generic, default configuration goes into `.conf` files (`fail2ban.conf` and `jail.conf`). Don't change these, as it makes upgrading difficult.
- Overrides to the generic configuration go into `.local` files corresponding to the `.conf` files. These only need to contain the specific settings you want overridden, which helps maintainability.
- Filters go into `filter.d` — this is where you define regexps, each going into its own file
- Actions go into `action.d` — you probably won't need to add one, but it's handy to know what's available
- "jails" are a configuration unit that specify one regexp to check, and one or more actions to trigger when the threshold is reached, plus the threshold settings (e.g. more than 3 matches in 60 seconds causes that address to be blocked for 600 seconds)
- Jails are defined in `jail.conf` and `jail.local`. Don't forget the `enabled` setting for each one — it can be as bad to have the wrong ones enabled as to have the right ones disabled.

Running Fail2Ban

- Use `/etc/init.d/fail2ban {start|stop|status}` for the obvious operations
- Use `fail2ban-client -d` to get it to dump its current configuration to STDOUT. Very useful for troubleshooting.
- Mind the CPU usage; it can soak up resources pretty quickly on a busy site, even with simple regexp
- It can log either to syslog or a file, whichever suits your needs better

Common Configuration

jail.local

```
# The DEFAULT allows a global definition of the options. They can be override
# in each jail afterwards.

[DEFAULT]

# "ignoreip" can be an IP address, a CIDR mask or a DNS host. Fail2ban will not
# ban a host which matches an address in this list. Several addresses can be
# defined using space separator.
# ignoreip = <space-separated list of IPs>

# "bantime" is the number of seconds that a host is banned.
bantime  = 600

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime  = 60

# "maxretry" is the number of failures before a host get banned.
maxretry = 3


[ssh-iptables]

enabled  = false


[apache-shorewall]

enabled  = true
filter   = cac-login
action   = shorewall
logpath = /var/log/httpd/confluence-access.log
bantime = 600
maxretry = 3
findtime = 60
backend = polling
```

Configuring for Confluence

The following is an example only, and you should adjust it for your site.

filter.d/confluence-login.conf

```
[Definition]

failregex = <HOST>.*"GET /login.action

ignoreregex =
```

Configuring for JIRA

The following is an example only, and you should adjust it for your site.

filter.d/jira-login.conf

```
[Definition]

failregex = <HOST>.*"GET /login.jsp

ignoreregex =
```

Deployment Planning Activity

Planning for rolling out our products or capacity planning for large instances is better suited for service offerings than Atlassian Support. We will refer this kind of activity to our partner network. This includes establishing specific upgrade and deployment plans for existing installations.

We suggest customers run the Performance Testing Scripts available for products to see how well their software instance performs given the hardware, configuration in use and expected workload. Using this data, the instance can then be tuned for performance should there be any issues encountered.

It is also recommended that you closely monitor your production instance to ensure that performance does not degrade as your instance grows in size.

Atlassian does not have provide benchmarking data at this time.

Should you require any assistance, it is best to take advantage of our public forums or contact our partners.

JIRA Releases

Latest Production Releases

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Please view the release notes to get up-to-date information about the improvements made in each release. If upgrading from a previous version of JIRA please pay attention to the *Upgrade Notes/Guide* of the latest version and any version of JIRA that your are 'skipping' during the upgrade.

Release Summary

The features of each JIRA release, up to and including the latest version, can be found in the JIRA Release Summary.

For full details on each of the JIRA releases, please read the release notes for the previous releases listed below.

Previous Releases

See the complete list of Releases Notes and Upgrade Guides for information about older releases.

JIRA Release Summary

This page shows the highlights of the major JIRA releases.

Current Release

For information about the latest release, please go to the Release Notes.

JIRA 4.4 — 2 August 2011

- User Time Zones
- Visual Workflow Designer
- Workflow Viewer on the 'View Issue' Screen
- Search (JQL) Enhancements
- Improved Setup Wizard with Database Configuration
- Improved JIRA Standalone Installer/Uninstaller and Automated Upgrade
- New-Look Administration Area
- Simplified Project Administration
- New Email Style
- Issue Linking when Resolving an Issue
- Editable Options for Custom Fields
- Multiple File Selection and Upload from the 'File Upload' Dialog Box
- New-look Activity Stream
- Graph of Vote History
- More in release notes

JIRA 4.3 — 16 March 2011

- Full Integration with LDAP and Active Directory
- New Plugin Management System
- Improved Importer
- JIRA Now Supports 'In-place Database Upgrades'
- Search for Issue Changes, Relative Dates and Relative Versions with JQL

- Quick Search Enhancements
- Revamped User Avatars
- Improvements to Issue Links
- Remembered Assignees
- Security Enhancements
- Application Links: Connecting Applications Together
- REST API Improvements
- More in release notes

JIRA 4.2 — 21 October 2010

- Keyboard Shortcuts and 'Operations Dialog'
- Editable 'Original Estimate'
- 'Log Work' Fields Available When Resolving Issues
- Labels
- User Avatars
- Viewable ZIP Files
- REST API (Alpha)
- More in release notes

JIRA 4.1 — 8 April 2010

- New-look 'View Issue'
- Streamlined Keyboard Shortcuts
- Customisable Email Subject
- Image Gallery
- ZIP Download of Attachments
- List of Logged-in Users
- JIRA Standalone ships with JIRA Configuration Tool, Database Drivers and Tomcat 6.0
- More in release notes

JIRA 4.0 — 6 October 2009

- Advanced Searching
- Dashboard Gadgets
- Activity Streams
- New-look "Browse Project"
- Charting Now Comes Standard
- New-look Header
- Issue Actions in the Issue Navigator
- Project Icons
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- More in release notes

JIRA 3.13 — 9 September 2008

- Shareable dashboards
- Improved filter sharing
- Favourite filters and dashboards
- Restoring projects
- Editable active workflows
- Enhanced sub-task quick creation
- Personal licenses
- New plugins
- Progress bar for long-running operations
- Application improvements
- More in release notes

JIRA 3.12 — 7 December 2007

- 'Trusted' Confluence
- 'JIRA System Administrators' permission
- FishEye plugin now bundled with JIRA
- Improvements to the Subversion plugin
- Improvements to the 'Project Statistics' and 'Filter Statistic' portlets
- New post function for workflows: 'Assign to Current User'
- Enhanced language support for searching
- Visual SourceSafe plugin
- More in release notes

JIRA 3.11 — 25 September 07

- Sub-task progress shown within issues
- Issue Navigator offers sub-task aggregates
- Time Tracking reports now include sub-tasks

- Multi-project 'Road Map' portlet
- Performance improvements
- Indexing improvements
- JIRA Labels Plugin
- More in release notes

JIRA 3.10 — 9 July 2007

- Editable Worklogs
- Start Date for Worklogs
- New way to browse Components
- New way to browse Versions
- Auto-complete 'User-picker' and 'Issue-picker'
- Auto-complete 'Issue-picker'
- More in release notes

JIRA 3.9 — 8 May 2007

- Ability to convert sub-tasks to issues (and vice versa)
- Convenient new scheduler for filter subscriptions
- Separate permissions for 'Delete Comment', 'Delete Attachment' and 'Delete Issue'
- Performance Improvements for Project Roles
- More in release notes

JIRA 3.8 — 13 March 2007

- Editable comments
- Self-installer for JIRA
- CAPTCHA for new account signup
- Integration with Crowd
- Improvements to the Bugzilla importer
- DHTML-loading of Issue screens
- More in release notes

JIRA 3.7 — 18 December 2006

- Project Roles - assign users and groups to roles on a per project basis
- Chart View - view charts in Issue Navigator using the JIRA Charting plugin
- RSS Improvements
- User Properties - record arbitrary information to the user profile (admin only)
- SVN project panel plugin - provides a summary of all commits made against a particular project or a project version
- More in release notes

JIRA 3.6 — 18 April 2006

- Custom Events - extension point for notification and workflow schemes
- Group Picker Custom Field - searchable in the issue navigator
- Per-issue Group Notifications and Permissions - based on the group picker custom field
- "I'm Feeling Lucky" Quick Search
- Collapsible Fields - control the level of detail of environment, description, individual comment fields and any textarea custom field
- Nestable Conditions - construct complex workflow conditions using nested conditions with AND or OR statements
- More in release notes

JIRA 3.5 — 01 February 2006

- Bulk Workflow Transition
- FogBugz Importer
- Charting Plugin
- MS Word Export
- JIRA Page Linker Plugin - linking a JIRA issue with a Confluence URL
- Component Lead Notification Type
- Bulk Assignment of Users to Groups
- More in release notes

JIRA 3.4 — 15 November 2005

- Issue Types Per Project
- Renderers - Confluence markup in JIRA text-based fields such as description and comments
- Issue Operation Plugin
- Announcement Banner
- RSS Support Improvements - live bookmarking with supported browsers
- Change Parent of Sub-Task
- Multi-user Custom Field
- More in release notes

JIRA 3.3 — 05 August 2005

- Multiple Project Filters - execute a search across multiple projects
- Bulk Move
- User Custom Field as Notification Target
- Extended Search Capabilities - search by date range for 'Created' and 'Updated' system fields and the custom field 'Date Time'
- JIRA Standalone ships with Tomcat 5.5
- More in release notes

JIRA 3.2 — 27 May 2005

- Field screens - configuration of field position and visibility for each issue operation and in Professional and Enterprise editions for each workflow transition screen
- Contextual custom fields - shared between projects and issue types
- Extended Bulk Edit Capabilities - Due Date, Reporter, Issue Security Level, Issue Type
- Improved internationalisation - Issue Constant Translations (Priorities, Statuses, Issue Types and Resolutions)
- Improved performace - quicker searching in the issue navigator and reports generation
- Smart Query
- Excel View
- More in release notes

JIRA 3.1 — 14 February 2005

- CSV Importer Wizard
- Add Comment on 'View Issue' field
- Webwork Plugin Type
- Assign Issues by Mail (via the CC field) using the Create Issue Handler
- More in release notes

JIRA 3.0 — 12 October 2004

- Workflow editor and configurable workflows
- Sub-tasks
- Plugin System
- Pluggable Custom Fields
- Dashboard Overhaul
- Issue cloning
- More in release notes

Production Releases

This page lists release notes and upgrade guides from past versions of JIRA.

If upgrading from a previous version of JIRA please pay attention to the *Upgrade Guide* of the version you are upgrading to, and any version of JIRA that you are 'skipping' during the upgrade.

You can also view lists of the Release Notes or Upgrade Guides separately.

- JIRA 5.0 Release Notes — JIRA 5.0 connects people, teams and applications. Users can share issues and search results with each other. They can also mention a colleague with '@username' in an issue to call their attention to an issue. Remote issue links connect JIRA issues to objects and pages in other web applications. Activity streams show events from all linked Atlassian applications and third-party apps.
    - JIRA 5.0 Upgrade Notes
        - Plugin Developer Notes for JIRA 5.0
            - How to add activities to the Third Party feed (REST API)
            - How to add activities to the Third Party feed (Java API)
            - JIRA Remote Issue Links
                - Remote Issue Links Field Guide
                - Remote Issue Links - Issue Link Creation Dialog Guide
                - Remote Issue Links - Issue Link Renderer Plugin Guide
                - Remote Issue Links - REST API Guide
            - JIRA REST API in JIRA 5.0
                - Issue Fields in JIRA REST
                - JIRA REST 5.0 - How to update an Issue ("Issue Mutation")
                - The Shape of an Issue in JIRA REST 5.0
            - Java API Policy for JIRA 5.0 onwards
            - Activity Streams Demos
            - Changes to markup, CSS and Decorators
            - JIRA Java API Changes in JIRA 5.0
        - Upgrading JIRA 3.x Data to JIRA 5.0
- JIRA 4.4 Release Notes — JIRA 4.4 brings you a visual Workflow Designer, a Workflow Viewer from the 'View Issue' page, simplified Administration and user-specific Time Zones. We are also very pleased to announce that the JIRA installation and upgrade processes have been improved and largely automated.
    - JIRA 4.4 Upgrade Notes
        - Plugin Developer Notes for JIRA 4.4

- JIRA 4.4.5 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.4.5. This point release contains several updates and fixes.
  - JIRA 4.4.5 Upgrade Notes
- JIRA 4.4.4 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.4.4. This point release contains several updates and fixes, plus an important bug fix (JRA-26172). This bug lead to a database lock, which typically occurred when custom plugins create issues in a high-load environment (specifically inside the same thread where another issue is being updated).
  - JIRA 4.4.4 Upgrade Notes
- JIRA 4.4.3 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.4.3. This point release contains new features that give users more JQL enhancements to filter issues based on system fields that possessed a specified value at some point in the past, in addition to several updates and fixes.
  - JIRA 4.4.3 Upgrade Notes
- JIRA 4.4.2 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.4.2. This point release contains new features that give users more JQL enhancements to filter issues based on system fields that possessed a specified value at some point in the past, in addition to several updates and fixes.
  - JIRA 4.4.2 Upgrade Notes
- JIRA 4.4.1 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.4.1. This point release contains new features that give administrators the ability to manage other users' shared filters and shared dashboards, in addition to several updates and fixes.
  - JIRA 4.4.1 Upgrade Notes

- JIRA 4.3 Release Notes — Identity management comes of age in JIRA 4.3, with complete LDAP integration. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins, and a new data importer for easier migration from your old systems. Additionally, a raft of new JQL functions give you many powerful new searching options, such as the ability to search an issue's change history.
  - JIRA 4.3 Upgrade Guide
    - Plugin Developer Notes for JIRA 4.3
  - JIRA 4.3.4 Release Notes
    - JIRA 4.3.4 Upgrade Guide
  - JIRA 4.3.3 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.3.3. This point release contains several updates and fixes, plus version 2.3.1 of the JIRA Importers Plugin, providing support for Pivotal Tracker and improvements in the way attachments are imported. The FishEye plugin version 3.1.5 is also included.
    - JIRA 4.3.3 Upgrade Guide
  - JIRA 4.3.2 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.3.2. This point release fixes two issues (JRA-24251 & JRA-24291) associated with the Windows Installer.
    - JIRA 4.3.2 Upgrade Guide
  - JIRA 4.3.1 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.3.1. This point release contains several updates and fixes, plus version 2.1 of the JIRA Importers Plugin, providing improved support for Bugzilla. Please see the documentation: Importing Data from Bugzilla.
    - JIRA 4.3.1 Upgrade Guide
- JIRA 4.2 Release Notes — JIRA 4.2 gives you a few extra minutes of precious time every day, by providing the ability to triage issues directly from the Issue Navigator without you having to open each issue. For the mouse-averse, the new 'Operations Dialog' box provides access to all menu options via the keyboard.
  - JIRA 4.2 Upgrade Guide
    - Updating JIRA Plugins for JIRA 4.2
  - JIRA 4.2.4 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.2.4. This point release fixes an issue with the 'Filter Results' gadget. It also contains version 1.7.1 of the JIRA Importers Plugin.
    - JIRA 4.2.4 Upgrade Guide
  - JIRA 4.2.3 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.2.3. This point release contains several updates and fixes, plus a new release of the JIRA Importers Plugin, providing improved support for Mantis and CSV imports. Please see the documentation: Importing Data from Mantis and Importing Data from CSV.
    - JIRA 4.2.3 Upgrade Guide
  - JIRA 4.2.2 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.2.2. This point release contains several updates and fixes, plus a Beta release of the JIRA Importers Plugin, providing improved support for Bugzilla. Please see the documentation: Importing Data from Bugzilla.
    - JIRA 4.2.2 Upgrade Guide
  - JIRA 4.2.1 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.2.1. This point release contains several updates and fixes, including the issue of remembering which sections of the 'View Issue' screen are collapsed. This point release is a highly recommended upgrade as it contains important fixes to security vulnerabilities — please refer to the JIRA Security Advisory 2010-12-06 for details.
    - JIRA 4.2.1 Upgrade Guide
- JIRA 4.1 Release Notes — This release makes your JIRA experience easier and more convenient than ever. The issue UI has been redesigned for a simpler, friendlier experience, and keyboard shortcuts have been streamlined. Issues can now be actioned directly from your dashboard via a handy dropdown in the gadgets. Each issue's attachments are now displayed in an image gallery, and can all be downloaded to a single ZIP file with just one click.
  - JIRA 4.1 Upgrade Guide
    - Updating JIRA Plugins for JIRA 4.1
  - JIRA 4.1.2 Release Notes — The Atlassian JIRA team is proud to announce the release of JIRA 4.1.2. This point release is a highly recommended upgrade as it contains important fixes to security vulnerabilities — please refer to the JIRA Security Advisory 2010-06-18 for details.
    - JIRA 4.1.2 Upgrade Guide
  - JIRA 4.1.1 Release Notes — The Atlassian JIRA team announces the release of JIRA 4.1.1. This point release is a highly recommended upgrade as it contains important fixes to security vulnerabilities in JIRA (listed below). For more information about these security vulnerabilities and patches to fix these vulnerabilities in earlier versions of JIRA, please refer to the JIRA Security Advisory 2010-04-16.
    - JIRA 4.1.1 Upgrade Guide
- JIRA 4.0 Release Notes — We have improved the UI to provide contextual awareness, improving the navigation and usability with features like activity streams and issue history. We have also added the most powerful searching capabilities ever seen in a bug tracker, called JIRA Query Language (JQL). The simple auto-complete entry system makes it incredibly easy for any user to create sophisticated queries.
  - JIRA 4.0 Upgrade Guide

        - JIRA 4.0 Database Schema Changes for MySQL and Oracle
        - Updating JIRA Plugins for JIRA 4.0
        - Upgrading JIRA 2.x Data to JIRA 4.0
        - Writing a Plugin Upgrade Task for JIRA 4.0
    - JIRA 4.0.2 Release Notes — The Atlassian JIRA team is proud to announce the release of JIRA 4.0.2. This point release contains over 40 bug fixes and improvements, notably including:
        - JIRA 4.0.2 Upgrade Guide
    - JIRA 4.0.1 Release Notes — The Atlassian JIRA team is proud to announce the release of JIRA 4.0.1. This point release contains over 60 bug fixes and improvements, notably including the gadget loopback issue. We are also very pleased to announce support for WebSphere 6.1.0.27.
        - JIRA 4.0.1 Upgrade Guide
- JIRA 3.13 Release Notes — This release fulfils some of the most popular JIRA feature requests. Dashboards can now be shared, and filter sharing has been improved — so it's easy to set up multiple 'template' dashboards, each with specific portlets and filters. New JIRA users can then simply select the dashboards most suited to them.
    - JIRA 3.13 Upgrade Guide
    - JIRA 3.13.5 Release Notes — The Atlassian JIRA team is proud to announce the release of JIRA 3.13.5 in Standard, Professional and Enterprise editions. This point release includes over 30 bug fixes and improvements.
        - JIRA 3.13.5 Upgrade Guide
    - JIRA 3.13.4 Release Notes — The Atlassian JIRA team is proud to announce the release of JIRA 3.13.4 in Standard, Professional and Enterprise editions. This point release includes over 20 bug fixes and improvements.
        - JIRA 3.13.4 Upgrade Guide
    - JIRA 3.13.3 Release Notes — The Atlassian JIRA team is proud to announce the release of JIRA 3.13.3 in Standard, Professional and Enterprise editions. This point release includes over 85 bug fixes and improvements, including an important security fix — please see JIRA Security Advisory 2009-04-02 for details.
        - JIRA 3.13.3 Upgrade Guide
    - JIRA 3.13.2 Release Notes
        - JIRA 3.13.2 Upgrade Guide
    - JIRA 3.13.1 Release Notes
        - JIRA 3.13.1 Upgrade Guide
- JIRA 3.12 Release Notes — JIRA 3.12 provides a number of enhancements for the upcoming JIRA Studio. Because some of these enhancements may be of benefit to you, we have decided to provide them as a public release rather than making you wait until JIRA 4.0.
    - JIRA 3.12 Upgrade Guide
        - JIRA 3.12 DB Schema Changes
    - JIRA 3.12.3 Release Notes
        - JIRA 3.12.3 Upgrade Guide
    - JIRA 3.12.2 Release Notes
        - JIRA 3.12.2 Upgrade Guide
    - JIRA 3.12.1 Release Notes
        - JIRA 3.12.1 Upgrade Guide
- JIRA 3.11 Release Notes — This release focuses on time-tracking. Time-tracking data (that is, the estimated and actual time spent on an issue) now includes the issue's sub-tasks. The aggregated time-tracking data is displayed both within individual 'parent' issues and in the Issue Navigator, so it can be easily reported on, exported to Excel, etc.
    - JIRA 3.11 Upgrade Guide
- JIRA 3.10 Release Notes — Editable worklogs; new ways to browse Components and Versions.
    - JIRA 3.10 Upgrade Guide
    - JIRA 3.10.2 Release Notes
        - JIRA 3.10.2 Upgrade Guide
    - JIRA 3.10.1 Release Notes
        - JIRA 3.10.1 Upgrade Guide
- JIRA 3.9 Release Notes — Convert issues to sub-tasks, and sub-tasks to issues. Use the convenient new scheduler to subscribe to issue filters.
    - JIRA 3.9 Upgrade Guide
    - JIRA 3.9.3 Release Notes — Professional French and German translations
        - JIRA 3.9.3 Upgrade Guide
    - JIRA 3.9.2 Release Notes
        - JIRA 3.9.2 Upgrade Guide
    - JIRA 3.9.1 Release Notes
        - Important Security Patch for JIRA versions 3.7.x & 3.8.x
        - JIRA 3.9.1 Upgrade Guide
- JIRA 3.8 Release Notes — Editable Comments, Self-installer for Windows, CAPTCHA for new account signup, Integration with Crowd, Improvements to the Bugzilla importer
    - Feedback for DHTML-loading of Issue screens
    - JIRA 3.8 Upgrade Guide
        - JIRA 3.8 Database Schema Changes
    - JIRA 3.8.1 Release Notes
        - JIRA 3.8.1 Upgrade Guide
- JIRA 3.6 Release Notes — Custom Events. Group Picker Custom Field, Wiki-Style Linking, Expandable Fields
    - JIRA 3.6 Upgrade Guide
    - JIRA 3.6.5 Release Notes
        - JIRA 3.6.5 Upgrade guide
    - JIRA 3.6.4 Release Notes
        - JIRA 3.6.4 Upgrade guide
    - JIRA 3.6.3 Release Notes
        - JIRA 3.6.3 Upgrade guide
    - JIRA 3.6.2 Release Notes
        - JIRA 3.6.2 Upgrade guide
    - JIRA 3.6.1 Release Notes
        - JIRA 3.6.1 Upgrade Guide
- JIRA 3.5 Release Notes — Bulk Workflow Transition, a new Charting Plugin, MS Word Export, JIRA Page Linker Plugin, Component

Lead Notification Type
    - JIRA 3.5 Upgrade Guide
    - JIRA 3.5.3 Release Notes
        - JIRA 3.5.3 Upgrade Guide
    - JIRA 3.5.2 Release Notes
        - JIRA 3.5.2 Upgrade Guide
    - JIRA 3.5.1 Release Notes
        - JIRA 3.5.1 Upgrade Guide
- JIRA 3.4 and 3.4.1 Release Notes — Issue by project, renderers, clone portlets, issue operation plugin, improvements and bug fixes.
    - JIRA 3.4 and 3.4.1 Upgrade Guide
        - Upgrading Custom Field Types in JIRA 3.4
        - Upgrading Custom Field Types in JIRA 3.4.1
    - JIRA 3.4.3 Release Notes
        - JIRA 3.4.3 Upgrade Guide
    - JIRA 3.4.2 Release Notes
        - JIRA 3.4.2 Upgrade Guide
    - JIRA 3.4.1 Release Notes
- JIRA 3.3 Release Notes — Create filters with multiple projects, improved date search, bulk move and performance improvements.
    - JIRA 3.3 Upgrade Guide
        - Parameter changes in Issue Navigator
        - Upgrading custom CustomFieldTypes in JIRA 3.3
        - Upgrading to JIRA 3.3 Standalone
    - JIRA 3.3.3 Release Notes
        - JIRA 3.3.3 Upgrade Guide
    - JIRA 3.3.2 Release Notes
    - JIRA 3.3.1 Release Notes
        - JIRA 3.3.1 Upgrade Guide
            - JIRA 3.3.1 Issue Tab Panel extension
- JIRA 3.2 Release Notes — Fields can be configured on a per-screen basis, screens can contain tabs, more flexible custom fields, and performance improvements.
    - JIRA 3.2 Upgrade Guide
        - Notifications no longer sent to raw email addresses if anonymous browsing disabled
        - Restricting Edit based on Issue Status
        - Upgrading custom CustomFieldTypes in JIRA 3.2
        - Upgrading Workflow Plugins for JIRA 3.2
        - Using Oracle 10g drivers to solve the 4000 character limitation
        - Workflows using default "Closed" status
    - 3.2 performance benchmarks
    - JIRA 3.2.3 Release Notes
    - JIRA 3.2.2 Release Notes
    - JIRA 3.2.1 Release Notes
- JIRA 3.0 Release Notes — Configurable workflow with editor, sub-tasks, plugin system, custom fields improvements, clone issue, change reporter, multiple attachments, introduction of the standard edition.
    - JIRA 3.0 Upgrade Notes
    - JIRA 3.0.3 Release Notes
    - JIRA 3.0.2 Release Notes
    - JIRA 3.0.1 Release Notes
- All JIRA Release Notes (version 3.x and later)
- All JIRA Upgrade Guides (version 3.x and later)
    - Aggregated JIRA 3.x Upgrade Guides
- JIRA 2.6 Release Notes
- JIRA 3.1 Release Notes — CSV import wizard, workflow action keyboard shortcuts, assign issues by mail and performance improvements.
    - JIRA 3.1 Upgrade Notes
    - JIRA 3.1.1 Release Notes
- Important Version-Specific Upgrade Notes — Below is a list of upgrade notes/guides for all previous major and minor releases of JIRA, each of which has important information you should be aware of for your JIRA upgrade: Upgrade notes/guides for minor releases are indented.
- JIRA 3.7 Release Notes — Project Roles, 'Charting' View for Issue Navigator, RSS Improvements, User Properties, SVN Project Panel plugin, SVN Commit Acceptance plugin
    - Issue Operations plugin
    - JIRA 3.7 Upgrade Guide
        - ActionManager Removed
        - JIRA 3.7 Database Schema Changes
        - Possible upgrade problems + solutions
        - SQL Scripts for 3.6.x to 3.7 schema upgrade
    - JIRA 3.7.4 Release Notes
        - JIRA 3.7.4 Upgrade Guide
    - JIRA 3.7.3 Release Notes
        - JIRA 3.7.3 Upgrade Guide
    - JIRA 3.7.2 Release Notes
        - JIRA 3.7.2 Upgrade Guide
    - JIRA 3.7.1 Release Notes
        - JIRA 3.7.1 Upgrade guide

JIRA 5.0 Release Notes

22 February 2012

The Atlassian team is proud to bring you JIRA 5.0.

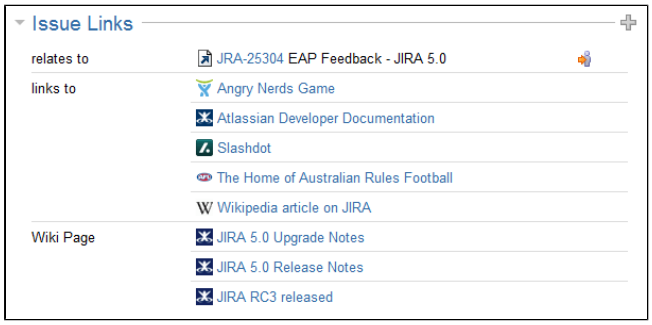
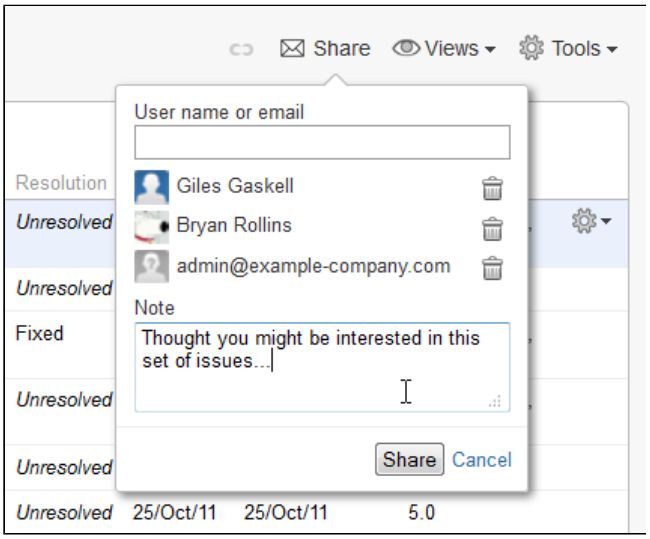
Overview

JIRA 5.0 connects people, teams and applications. Users can share issues and search results with each other. They can also mention a colleague with '@username' in an issue to call their attention to an issue. Remote issue links connect JIRA issues to objects and pages in other web applications. Activity streams show events from all linked Atlassian applications and third-party apps.

Even further, JIRA 5.0's new APIs are designed for developers who want to connect JIRA to other applications:

- New REST APIs to manage every facet of an issue.
- New APIs for remote issue links and activity streams.
- A stable Java API for JIRA.

Highlights:

Remote issue links	Share issues and mention users
	
• Create links to JIRA issues on another JIRA site • Create links to any URL • Create links via JIRA's Java or REST API	• Share issues quickly with other users • Mention a user in an issue comment • The username or email address autocomplete as you type

And More:

- Search for issues based on their history
- Activity streams now show activity from other applications
- Manage other users shared filters and dashboards
- Administration user interface improvements

- REST API (with tutorials) for working with issues in JIRA
- Stable Java API
- New troubleshooting and debugging tools
- New email handler wizard
- Enhancements to the 'view issue' page
- JIRA Add-Ons *(Plugins and Integrations)*
- Other enhancements and fixes

Thank you for your feedback:

More than **30** new feature requests implemented

Over **1500** votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

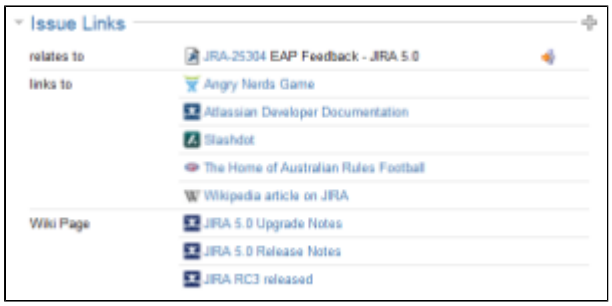
Upgrading to JIRA 5.0

JIRA 5.0 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 5.0 Upgrade Notes.

Note to developers: Please see Preparing for JIRA 5.0 on the Atlassian Developers site.

Highlights

Remote issue links



The *remote issue links* feature provides a powerful way to link JIRA issues to items external to your JIRA installation, residing on external applications.

Along with a Java and REST API to add these links, end users can also:

- Add an issue link from a JIRA issue to an issue on another JIRA site, including reciprocal links between these issues.
- Search for a Confluence page from a JIRA issue and add an issue link to that page.
- Add an issue link from a JIRA issue to any web page URL, such as a page of documentation, a technical note, or any other page o another web site.

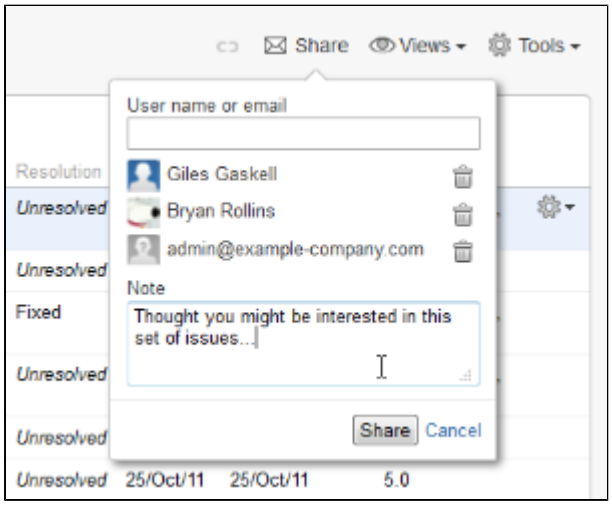
Check out a live example here.

If you are a:

- JIRA user — see Linking Issues for details on using remote issue links within JIRA.
- JIRA system administrator — see Configuring Issue Linking for details on how to make remote issue linking available to your users by setting up the required application links between JIRA and other applications.
- JIRA developer — see JIRA Remote Issue Links on our developer documentation site.

^Top

Share issues and mention users



Screenshot: Sharing a list of issues with other users

Screenshot: Mentioning a user in an issue comment

Need someone else to take a look at a JIRA issue or a list of issues?

- View any issue or a list of issues on the issue navigator, click the **Share** button at the top-right (or type **s**) and specify JIRA users (based on their names or user names) or *any* email address of people you want to share the issue with.

  Recipients will be emailed a link to the issue (or a list of issues 'shared' via the issue navigator).

- When creating, editing or commenting on an issue, simply mention other JIRA users in an issue's **Description** or **Comment** field.

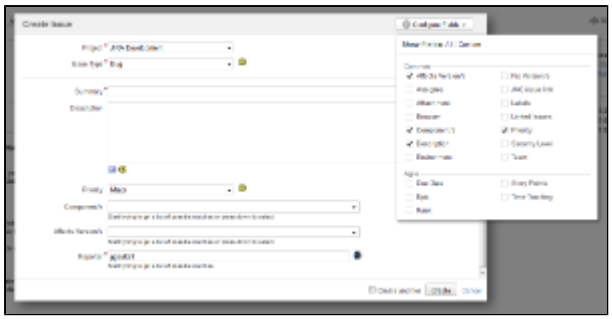
  Any JIRA users mentioned on an issue will receive details about the issue in an email message (sent to the addresses registered with their user accounts). The message's subject line will indicate that the person who used this feature 'mentioned' them on that issue.

Please Note:

- You require the **Browse Users** global permission to access the **Share** button or the 'suggested users' feature when 'mentioning' a user. However, if you know the username of a JIRA user, you can still mention them without this permission.
- JIRA system administrators will need to configure JIRA's outgoing SMTP mail server for the 'share' and 'mentions' features to work.

^Top

Rapidly create and edit issues



JIRA 5.0 lets you create and edit issues and sub-tasks much faster. Creating and editing is now performed in a dialog box (rather than a separate form), so that you do not need to leave the current page or have it reload just to start creating or editing an issue or sub-task.

You can customise fields on the **Create Issue**/**Edit Issue** dialog boxes by removing or adding fields fields through the **Configure Fields** button. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly used fields whenever you create or edit an issue.

The **Create Issue** dialog box allows you to rapidly create a series of related issues with similar options. When you select the **Create another** check box before clicking the **Create** button, JIRA creates your issue and automatically pre-populates a new **Create Issue** dialog box with your previous field values, whilst leaving the **Summary** field blank. Be aware that any attachments attached to your previously created issue are not carried across by this feature.

The **Assignee**, **Project** and **Issue Type** fields use 'autocompletion' too. Hence, you no longer a need to scroll through a whole raft of items to specify these fields when creating or editing an issue.

You can easily access these dialog boxes by typing '**c**' to create an issue, or '**e**' from a selected issue on the issue navigator or 'view issue' page to edit that issue.

^Top

Search for issues based on their history

'CHANGED' operator introduced

Introduced in JIRA 4.4.3, JQL's **CHANGED** operator can accept the optional predicates **FROM**, **TO**, **ON**, **DURING**, **BEFORE**, **AFTER** and **BY**, and can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

For example, this link shows all the issues logged against the JIRA project on our 'jira.atlassian.com' site, whose **Fix Version** field was changed to **"5.0"**.

You can also create more complex JQL queries with the **CHANGED** operator by fine-tuning them with predicates. For example, the following JQL query:

```
status changed FROM "In QA Review" TO "QA Rejected" BY freddo AFTER startOfWeek() BEFORE
endOfWeek()
```

will find any issues whose **Status** field value was at some point **"In QA Review"** but changed to **"QA Rejected"**, by user **freddo** between the start and end of the current week.

You can use complex queries such as these to generate the 'Single Level Group By Report' in the screenshot above, which shows grouping

by **Team**.

'WAS' operator enhanced

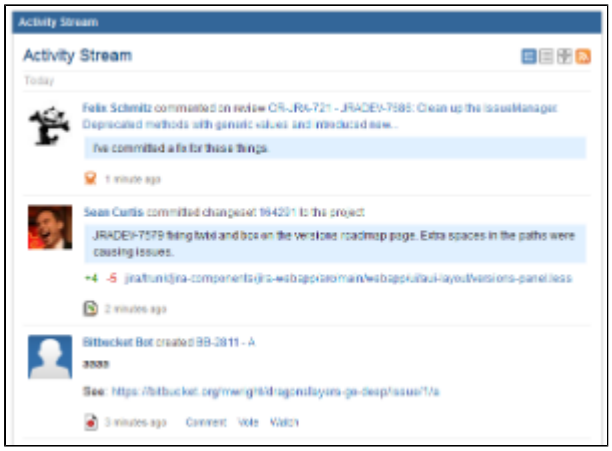
Also introduced in JIRA 4.4.3 was the ability of the **WAS** operator to work with the **Fix Version** field. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to **4.4**.

^Top

Activity streams now show activity from other applications



Screenshot: Activity stream gadget showing activity from other Atlassian applications

Screenshot: Activity stream gadget showing activity from non-Atlassian applications

We have expanded the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

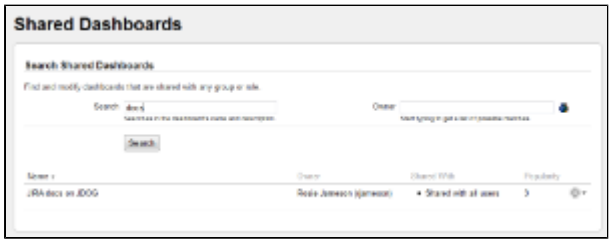
With these new features, you can:

- See Confluence page updates from your activity streams in JIRA, then drill down into those Confluence pages for more information
- See updates from another JIRA site. For example, activity streams on your development team's JIRA site (behind the firewall) can include activity on your support team's customer facing JIRA site.

^Top

6

Manage other users' shared filters and dashboards



Introduced in JIRA 4.4.1, JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboard. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

You can access these features by selecting **Administration > Users > Shared Filters** or **Shared Dashboards** (or using the keyboard shortcut **g + g +** start typing **shared filters** or **shared dashboard**).

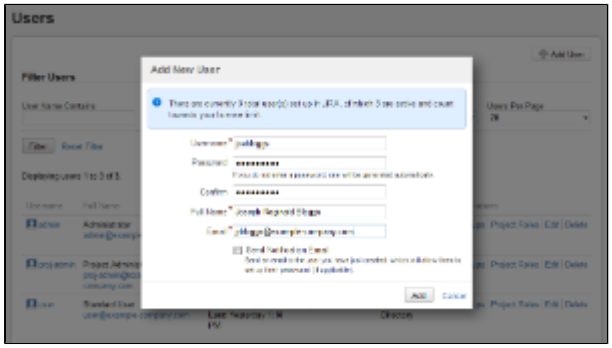
On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

^Top

7

Administration user interface improvements



Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 provides further improvements to the Administration UI by converting forms on various Administration pages to convenient dialog boxes.

For example, the form for adding users is now a dialog box, which is accessed by clicking **Add User** on the **Users** page of JIRA's Administration area.

In addition to the dialog box for adding a new user, the 'Attachments' and 'Workflows' pages have been redesigned and the forms associate with these pages have been converted into convenient dialog boxes too.

^Top

REST API (with tutorials) for working with issues in JIRA

JIRA's REST API has undergone a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues and editing existing ones.
- Delete existing issues and their subtasks.
- Create remote 'issue links'.
- Retrieve metadata from your favourite filters and dashboards.
- Retrieve metadata about your permissions.
- Almost all system fields and JIRA's built-in custom field types are supported.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2' (or '`latest`' to use the latest REST API version available with your version of JIRA).

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

You might also want to try out Atlassian's new REST API Browser, which is available in the Atlassian Plugin SDK and can be accessed by adding `/plugins/servlet/restbrowser#/user/search` to the end of the URL for accessing JIRA (e.g. `http://localhost:2990/jira/plugins/servlet/restbrowser#/user/search`) within the SDK environment.

^Top

Stable Java API

JIRA's Java API has undergone a significant number of changes and improvements to provide the following:

- More stability and compatibility with future versions of JIRA. See our Java API Policy for JIRA for details.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

If you are developing plugins for JIRA 5.0, please also check out our newly published set of JIRA plugin tutorials.

^Top

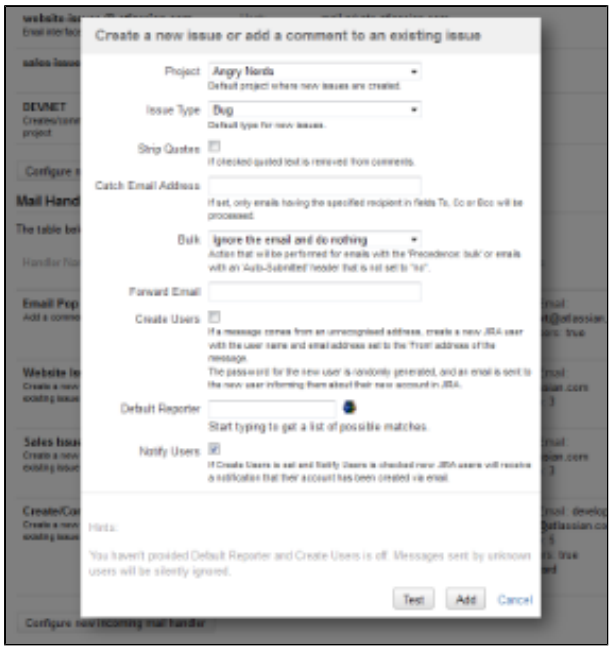
New troubleshooting and debugging tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

- A number of email debugging tools are now provided to System Administrators in the new **Logging and Profiling** page under **Troubleshooting and Support**
  - Enable or disable mail logging
  - Turn debug mail logging on or off
  - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New email handler wizard



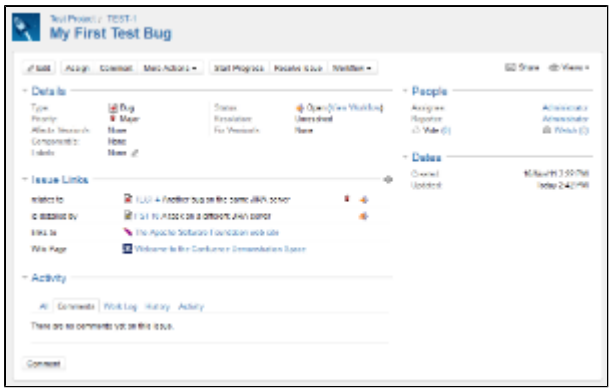
JIRA 5.0 incorporates a new mail handler wizard that greatly simplifies the process of configuring incoming mail handlers for creating issues or comments from email messages.

There is no longer a need to configure a JIRA service and enter a complex string of mail handler parameters to handle your email messages. Instead, simply you configure your mail handler through a convenient wizard.

Improvements have been made to the layout of JIRA's mail configuration options. The configuration options in the **Mail Servers** administration page have been separated into two separate pages — one for **Outgoing Mail** (SMTP) and another for **Incoming Mail** (POP/IMAP) configurations. The configuration options for mail handlers have been moved from the **Services** administration page and incorporated into the **Incoming Mail** page.

^Top

Enhancements to the 'view issue' page



The 'view issue' page has the following enhancements:

- The **Issue Links** section of the 'view issue' page has been redesigned to cater for remote issue links (above) and makes better use of space — the separate line used to group issue link types in earlier versions of JIRA has been removed.
- The right-hand side of the view issue page now uses a fixed width. Hence, when the 'view page' is maximised on large, high resolution monitors, the association between field names and values is not lost.
- In JIRA versions prior to 5.0, initially clicking any tab in the **Activity** section would reload the whole 'view issue' page. In JIRA 5.0, the content within each of these (non-selected) tabs only loads upon being clicked by a user and does so independently of the rest the 'view issue' page. This modified behaviour provides more rapid access to the information on these tabs.
- **For plugin developers** - the left hand side of the 'view issue' page can be customised via web panels. Hence, you can now insert your own custom panels anywhere below the operations bar on the 'view issue' page.
  The web panel location for the left-hand side of the 'view issue' page is `atl.jira.view.issue.left.context` — refer to the View Issue Page Locations page of the Web Fragments guide in our Atlassian Developers documentation site for more information. Also see our plugin tutorial for an example of customising the right-hand side of the 'view issue' page (a feature which became available in JIRA 4.4).

^Top

13

JIRA Add-Ons *(Plugins and Integrations)*

JIRA 5.0 contains a great deal of new capabilities for developers who want to integrate with JIRA, including new integration features like remote issue links and activity streams, a new REST API (link) and a stable Java API.

JIRA Customers will already see the benefits of these new capabilities:

- Over 100 integrations to JIRA 5.0 are available (at 5.0 release), so many of the plugins customers depend on are already JIRA 5.0 compatible.
- JIRA's stable Java and new REST APIs allow Add-On developers to build integrations to JIRA which will be forwards compatible with future releases of JIRA 5.x, so upgrades become even easier for JIRA customers.
- Many of these new integrations use the new activity streams and remote issue links features.

See the full list of JIRA 5.0 compatible integrations on the Atlassian Plugin Exchange.

^Top

Other enhancements and fixes

When you access your new or upgraded JIRA 5.0 installation, JIRA launches the **What's New in JIRA ...** dialog box, which provides brief overview of the new features available in that JIRA version.

This dialog box can be prevented from showing up whenever you access JIRA by selecting the **Don't show again** check box at the base

of the dialog box. However, you can access this dialog box again by choosing the **What's New** item from your user name drop down menu.

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

JIRA 5.0 Upgrade Notes

On this page:

- Upgrading from JIRA 4.4.x to 5.0
    - The 'trackbacks' feature has been removed
    - JIRA servers using Oracle databases require additional configuration
    - Mail handlers are no longer configured as a JIRA Service
    - GreenHopper plugin
    - JIRA Toolkit plugin
    - Other plugins
    - Translations
    - Licensing changes
    - Naming convention changes for JIRA distributions
  - Known issues
    - When creating an issue in Internet Explorer 8, the 'Issue Type' field behaves differently
    - Performance degradation when users view lists of issues sorted by date from large JIRA sites
- Upgrading from between JIRA versions 4.0.0 and 4.3.x to JIRA 5.0
- Upgrading to JIRA 5.0 from a version of JIRA prior 4.0.0

Upgrading from JIRA 4.4.x to 5.0

Please follow the instructions in the general Upgrading JIRA guide, as well as the JIRA 5.0-specific instructions in the sections below. The general 'Upgrading JIRA' guide contains important tasks that are essential for getting your upgraded JIRA installation to work correctly and if necessary, migrating existing configurations.

The 'trackbacks' feature has been removed

The trackback feature has been removed in JIRA 5.0 and its functionality has been replaced solely by remote issue links.

JIRA servers using Oracle databases require additional configuration

Due to Active Objects changes in JIRA 5.0, any customers with a JIRA server connected to an Oracle database will need to ensure that the database user account (through which JIRA connects to Oracle) has the **create sequence** and **create trigger** Oracle database permissions. To apply these permissions, execute the following Oracle SQL commands with the relevant database user account, *before* upgrading JIRA to version 5.0:

```
grant create sequence to <user>;
grant create trigger to <user>;
```

Mail handlers are no longer configured as a JIRA Service

The introduction of the mail handler wizard in JIRA 5.0 means that after upgrading to JIRA 5.0, any existing mail handlers or any new mail handlers created will no longer be configurable through JIRA's **Services** page. Instead, all mail handlers are created and edited through the **Mail Handlers** section of the **Incoming Mail** page. See Configure a mail handler for details.

GreenHopper plugin

Please note that JIRA 5.0 is only compatible with GreenHopper 5.9.

The GreenHopper plugin is no longer bundled with JIRA. Instead, please use the JIRA Universal Plugin Manager to update your GreenHopper plugin.

JIRA Toolkit plugin

From JIRA 5.0, version 0.21 or later of the JIRA Toolkit plugin is required.

Hence, if you are using any of the custom fields provided by this plugin and want to ensure that messages are always shown to users upon the creation of issues, please ensure that these fields are made 'required' in your field configurations:

- Message Custom Field (for edit)
- Velocity processed Message Custom Field (for edit)

Other plugins

PLEASE BE AWARE — JIRA 5.0 introduces several changes that may break existing plugins which are not bundled with JIRA.

If you have a developed a plugin, then please read the Preparing for JIRA 5.0 section of our developer documentation site for more details. This documentation describes changes in JIRA 5.0 which may affect the compatibility of your plugin with JIRA 5.0.

If you are using a plugin developed by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 5.0.

Translations

Please note that unlike earlier versions of JIRA, JIRA 5.0 ships with a reduced set of translations, which may not be the most up-to-date available. The most recent translations for French, German, Spanish, Japanese and other languages can be obtained easily via the Universal Plugin Manager by searching for "language pack". See Managing JIRA's Plugins for details on using this feature.

You can also download and install these language packs manually from our Plugin Exchange site.

Language packs for other languages which are not available through the Universal Plugin Manager or our Plugin Exchange site can be downloaded from our translation site (https://translations.atlassian.com/).

Licensing changes

Due to licensing changes in JIRA 5.0, some customers may wish to modify their JIRA license to one with a lower user count. Since the number of users that count towards your JIRA license is defined by the number of users with the **JIRA Users** global permission, some customers who upgrade to JIRA 5.0 may experience problems with their users being unable to create issues (due to license counts being exceeded).

Customers can overcome this issue by reducing the number of users that count towards their JIRA license. See How do I reduce my user count in JIRA for details on how to do this.

Naming convention changes for JIRA distributions

In versions of JIRA prior to 5.0, JIRA distributions had the following naming conventions:

- **JIRA Standalone** — JIRA distributions that were 'bundled' with their own Apache Tomcat application server and were installed from either an executable or archive file.
- **JIRA WAR** — JIRA distributions that could be customised, then built for deployment to a separate Apache Tomcat application server installation.

Several customers misconstrued the meaning of term **Standalone** which was used to describe JIRA distributions bundled with their own Apache Tomcat application server. Hence, from JIRA 5.0:

- The naming convention for JIRA WAR distributions will not change. The installation files for these distributions include 'WAR' or 'war' in their file names.
- JIRA distributions which are bundled with their own Apache Tomcat application server are referred to as 'recommended' JIRA distributions (in the documentation) and the installation files for these distributions do not use a distinct term to distinguish them from JIRA WAR distributions.
- The installation files for 'recommended' JIRA distributions include:
  - JIRA Windows 32-/64-bit installer
  - JIRA Linux 32-/64-bit installer
  - JIRA zip/tar.gz archives

For installation instructions for 'recommended' JIRA distributions, see Installing JIRA. For installation instructions for JIRA WAR distributions, see Installing JIRA WAR.

Known issues

Before you begin the upgrade, please check for known issues. Sometimes we find out about a problem with the latest version of JIRA after we have released the software. In such cases we publish information about the known issues in the JIRA Knowledge Base. Please check for known issues and follow the instructions to apply any necessary patches.

If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

When creating an issue in Internet Explorer 8, the 'Issue Type' field behaves differently

Due to existing behaviour in Internet Explorer 8, when creating an issue, JIRA does not select the default issue type for the specified project on the **Create Issue** dialog box or page if you have no history of having previously created issues. Instead, the issue type at the top of the dropdown list is selected if the user has not previously created an issue. See JRA-27029 for more information.

Performance degradation when users view lists of issues sorted by date from large JIRA sites

On large JIRA sites, users may experience performance degradation in the following situations:

- Viewing search results in either the issue navigator or issue filters, which retrieve more than 25,000 issues sorted by date.
- For projects with more than 25,000 issues, viewing sections of the summary page that list issues sorted by date - for example, the 'Issues: Updated recently' section.

See JRA-27206 for more information.

Upgrading from between JIRA versions 4.0.0 and 4.3.x to JIRA 5.0

In addition to the points listed above, please read the Important Version-Specific Upgrade Notes for every version of JIRA you are skipping.

Upgrading to JIRA 5.0 from a version of JIRA prior 4.0.0

Please be aware that you cannot directly upgrade to JIRA 5.0 from a version prior to 4.0.0 (for example from JIRA 3.10.3). Instead, you will need to upgrade your JIRA installation to a JIRA 4.x version first before you can upgrade to JIRA 5.0. The latest JIRA 4.4.x version is recommended for this process. The following resources in the JIRA 4.4.x documentation are good places to start for upgrading to the latest JIRA 4.4.x release:

- JIRA 4.4.5 Upgrade Notes
- JIRA 4.4 Upgrade Notes
- Upgrading JIRA guide

You can then follow the JIRA 5.0 Upgrade Notes (above) to upgrade your JIRA 4.x installation to JIRA 5.0.

Plugin Developer Notes for JIRA 5.0

This documentation is now on the Atlassian Developers wiki.

How to add activities to the Third Party feed (REST API)

This documentation is now on the Atlassian Developers wiki.

How to add activities to the Third Party feed (Java API)

This documentation is now on the Atlassian Developers wiki.

JIRA Remote Issue Links

This documentation is now on the Atlassian Developers wiki.

Remote Issue Links Field Guide

This documentation is now on the Atlassian Developers wiki.

Remote Issue Links - Issue Link Creation Dialog Guide

This documentation is now on the Atlassian Developers wiki.

Remote Issue Links - Issue Link Renderer Plugin Guide

This documentation is now on the Atlassian Developers wiki.

Remote Issue Links - REST API Guide

This documentation is now on the Atlassian Developers wiki.

JIRA REST API in JIRA 5.0

This documentation is now on the Atlassian Developers wiki.

Issue Fields in JIRA REST

This documentation is now on the Atlassian Developers wiki.

JIRA REST 5.0 - How to update an Issue ("Issue Mutation")

This documentation is now on the Atlassian Developers wiki.

The Shape of an Issue in JIRA REST 5.0

This documentation is now on the Atlassian Developers wiki.

Java API Policy for JIRA 5.0 onwards

This documentation is now on the Atlassian Developers wiki.

Activity Streams Demos

This documentation is now on the Atlassian Developers wiki.

Changes to markup, CSS and Decorators

This documentation is now on the Atlassian Developers wiki.

JIRA Java API Changes in JIRA 5.0

This documentation is now on the Atlassian Developers wiki.

Upgrading JIRA 3.x Data to JIRA 5.0

If you are upgrading from JIRA 3.x data (or earlier) to JIRA 5.0, you must upgrade to any JIRA 4.x release first, although the latest 4.4.x version is recommended for this process. The following resources in the JIRA 4.4.x documentation are good places to start for upgrading to the latest JIRA 4.4.x release:

- JIRA 4.4.5 Upgrade Notes
- JIRA 4.4 Upgrade Notes
- Upgrading JIRA guide

You can then follow the JIRA 5.0 Upgrade Notes to upgrade your JIRA 4.x installation to JIRA 5.0.

JIRA 4.4 Release Notes

2 August 2011

The Atlassian JIRA team is proud to bring you a brand new version of one of the world's favourite issue-trackers.

JIRA 4.4 brings you a visual Workflow Designer, a Workflow Viewer from the 'View Issue' page, simplified Administration and user-specific Time Zones. We are also very pleased to announce that the JIRA installation and upgrade processes have been improved and largely automated.

Upgrading to JIRA 4.4 is free for all customers with active JIRA software maintenance as of 2 August 2011.

Highlights of JIRA 4.4:

- For Users:
  - User Time Zones
  - New Email Style
  - Issue Linking when Resolving an Issue
  - Workflow Viewer on the 'View Issue' Screen
  - Multiple File Selection and Upload from the 'File Upload' Dialog Box
  - JQL Enhancements
  - New-look Activity Stream
  - Graph of Vote History
- For Administrators:
  - Visual Workflow Designer
  - New-Look Administration Area
  - Simplified Project Administration
  - Editable Options for Custom Fields
- Setup and Installation:
  - Improved Setup Wizard with Database Configuration
  - Improved JIRA Standalone Installer/Uninstaller and Automated Upgrade
- Platforms, APIs:
  - IE 9 and Firefox 5 Support
  - REST API improvements
- Other Enhancements and Fixes

Thank you for your feedback:

More than **50** new feature requests implemented

Over **1400** votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

Upgrading to JIRA 4.4

JIRA 4.4 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 4.4 Upgrade Notes.
If you are upgrading a JIRA WAR installation, please ensure you read the section on Upgrading a JIRA 4.3.x (or Earlier) WAR Installation, including the subsection on **Migrating Your Database Configuration**.

Note to developers: Please see the Plugin Developer Notes for JIRA 4.4.

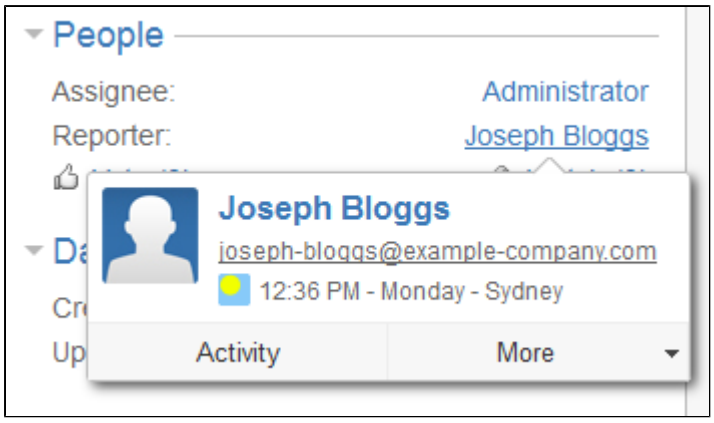
Highlights of JIRA 4.4

For Users:

User Time Zones

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: times will be displayed to a user in their local time zone, rather than the server's time zone.

You can set a default user time zone at an administration level, and individual users have the ability to choose their own time zone. Each user's time zone is displayed in their hover profile.

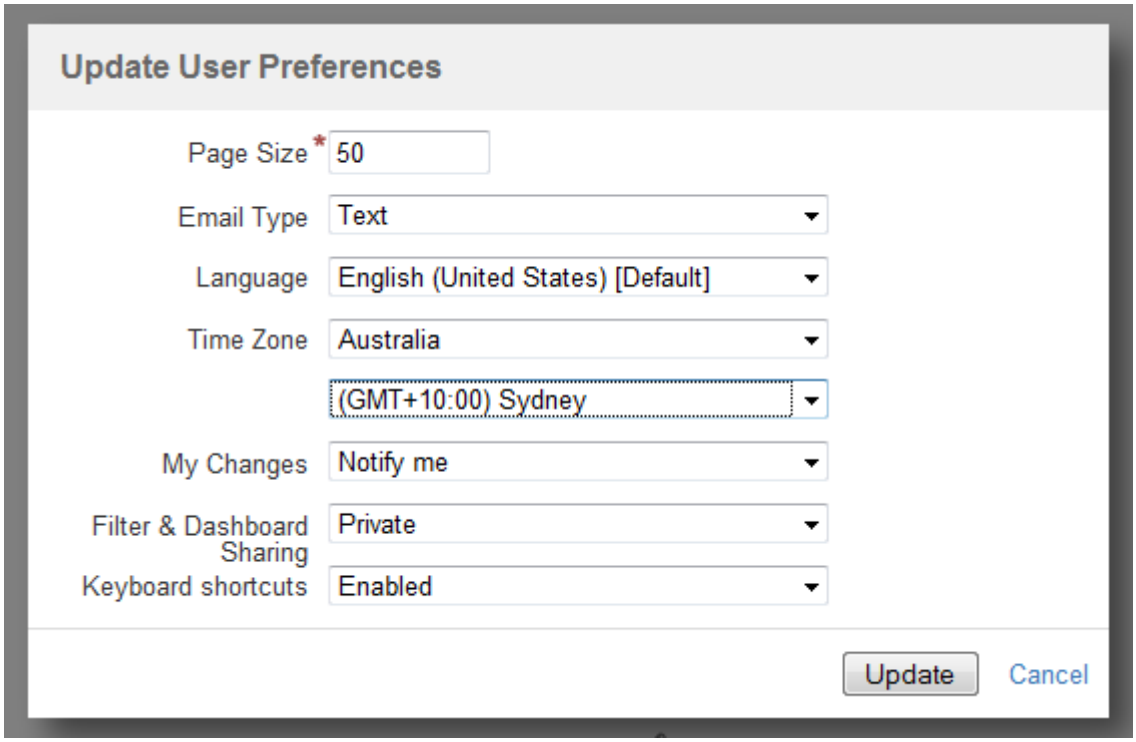


Time zone support has been implemented for quick searching, simple searching and advanced searching, chart and report gadgets, date/time-based custom fields, as well as issue histories, work logs and source code check-ins (via the JIRA FishEye Plugin).

Note: Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information).

JIRA administrators can change the default time zone by going to '**Administration**' > '**General Configuration**' (under '**Global Settings**'), and editing the '**Default user time zone**'.

Users can also change their individual time zone setting via their user profile:



Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

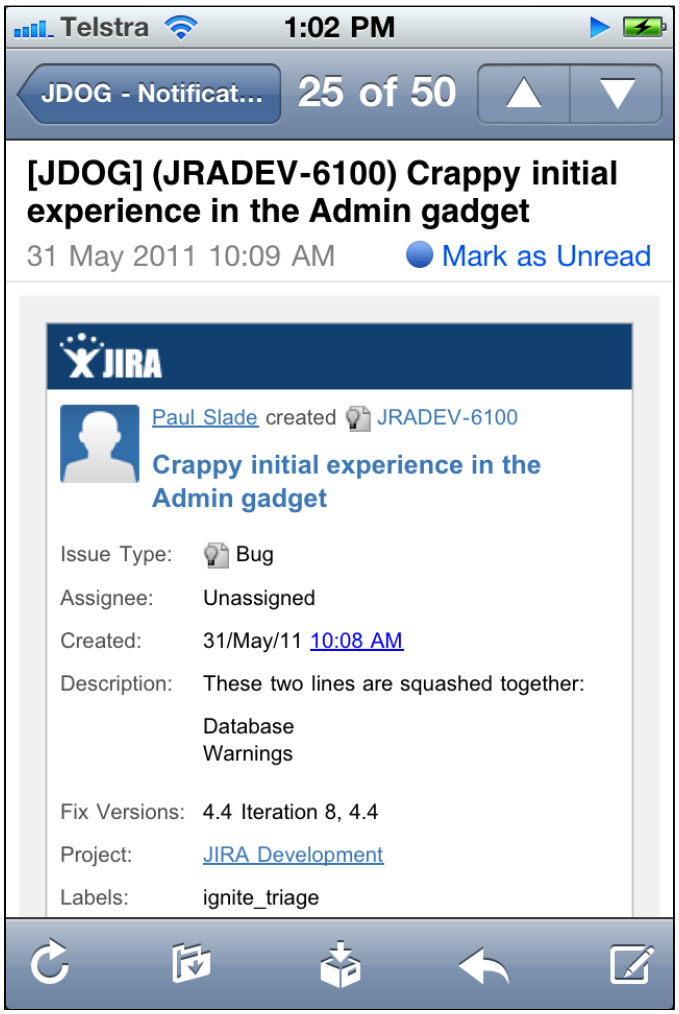
^Top

New Email Style

The HTML email templates have undergone a complete visual refresh.

They will also thread better in mail clients (such as Gmail), so all emails relating to one issue will thread together.

They will also render nicely in mobile mail clients:



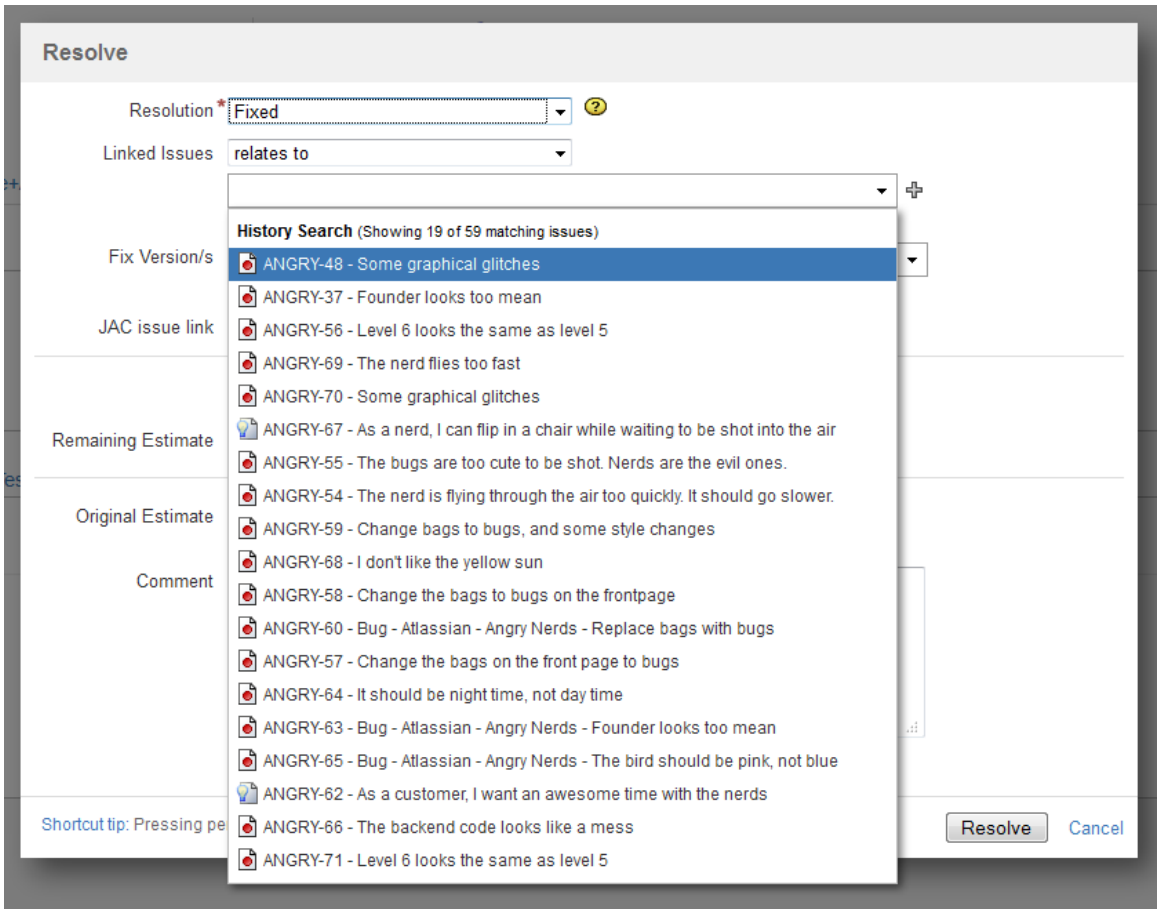
^Top

Issue Linking when Resolving an Issue

When resolving an issue, you can create links to other issues on an issue resolution screen. This is handy when you want to resolve an issue as a 'duplicate' of another *and at the same time* link to the duplicate issue.

For convenience, your recent issue links and resolutions are readily accessible from the 'Linked Issues' fields (in the screenshot below). The 'Linked Issues' fields can also be added to any JIRA screen via the new 'Issue Linking' field in JIRA 4.4. See Defining a Screen for more information.

If you are upgrading from an earlier version of JIRA, you need to configure this feature manually through JIRA's administration area. Please refer to the Upgrade Notes for details.



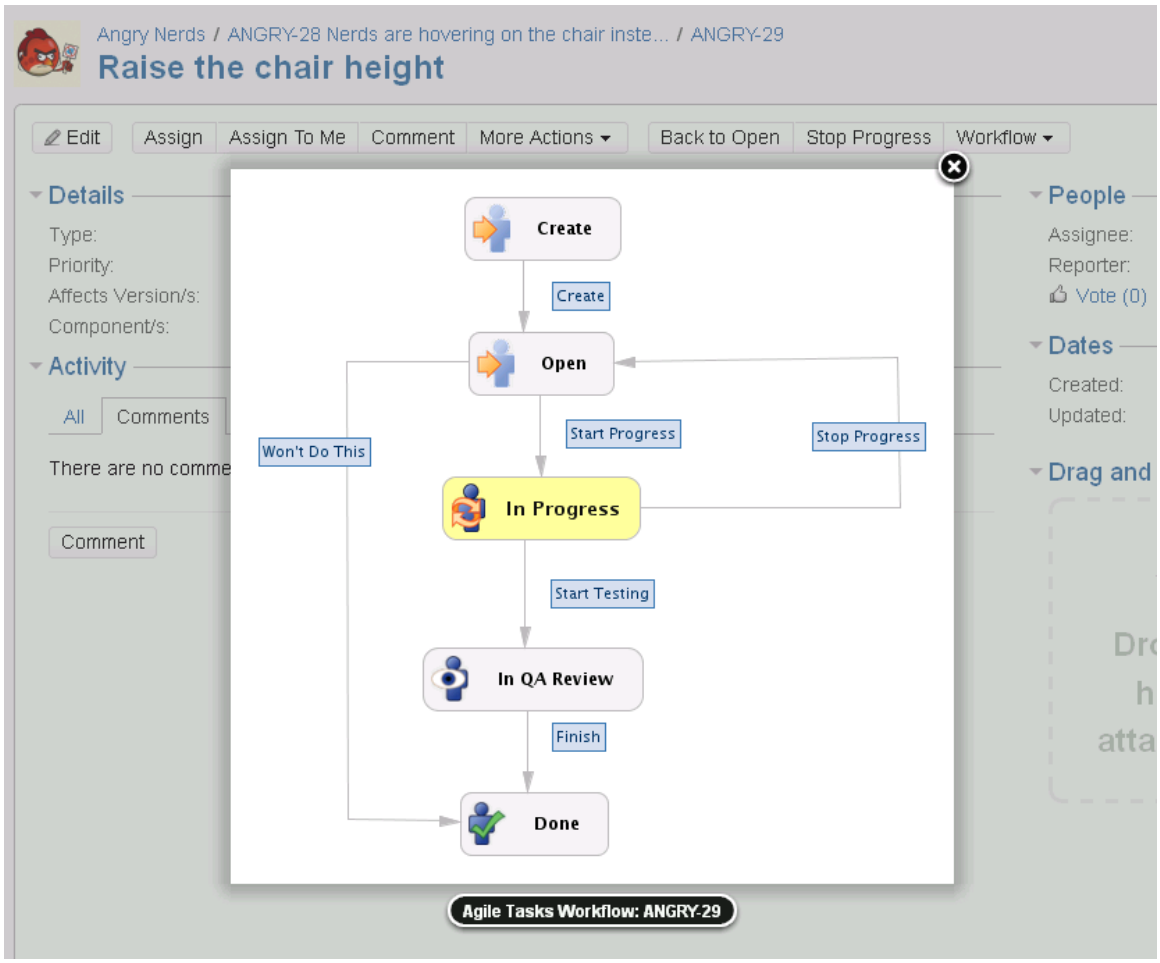
^Top

Workflow Viewer on the 'View Issue' Screen

You can also see a read-only view of the workflow from the 'View Issue' page — just click the '**View Workflow**' link against the 'Status' field. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View (Read-Only) Workflow' is required to access the workflow viewer feature from the 'View Issue' page.



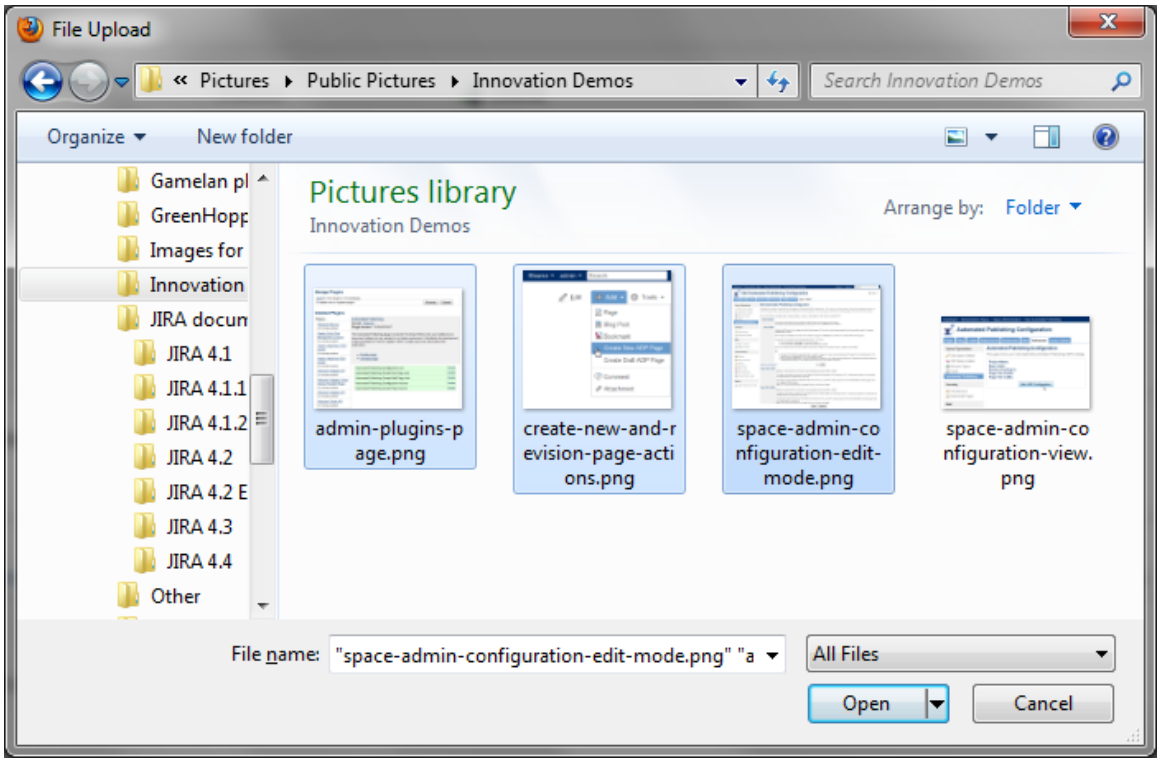
^Top

5

Multiple File Selection and Upload from the 'File Upload' Dialog Box

When using JIRA's 'Attach Files' dialog box, you can now select multiple files in the 'File Upload' dialog box that appears after clicking the 'Browse' button.

This feature currently works with Firefox 3.6 or later and Chrome web browsers.



^Top

JQL Enhancements

Do more with 'WAS'

The ability to search an issue's Change History was introduced in JIRA 4.3 and allowed you to search the history of the **Status** field.

In JIRA 4.4 you can now also search the history of:

- the **Assignee**field, e.g.:

```
assignee WAS "jsmith"
```

- the **Reporter**field, e.g.:

```
reporter WAS "djones"
```

You can also now search for a field that had a particular value:

- **ON**a given date — e.g. find issues that had a status of "closed" on May 31st:

```
status WAS "closed" ON "2011/05/31"
```

- **BEFORE**a given date — e.g. find issues that were assigned to jsmith before May 31st:

```
assignee WAS "jsmith" BEFORE "2011/05/31"
```

- **AFTER**a given date — e.g. find issues that were assigned to me after May 31st:

```
assignee WAS currentUser() AFTER "2011/05/31"
```

- **DURING**a given date range — e.g. find issues that were assigned to me during May:

```
assignee WAS currentUser() DURING ("2011/05/01","2011/05/31")
```

- set **BY**a particular user — e.g. find issues that were assigned to Fred by me:

```
assignee WAS "Fred" BY currentUser()
```

How many Watchers?

The new '**Watchers**' field allows you to search for issues with a specified number of watchers, e.g.:

```
watchers > 3
```

For more details please see Advanced Searching.

^Top

New-look Activity Stream

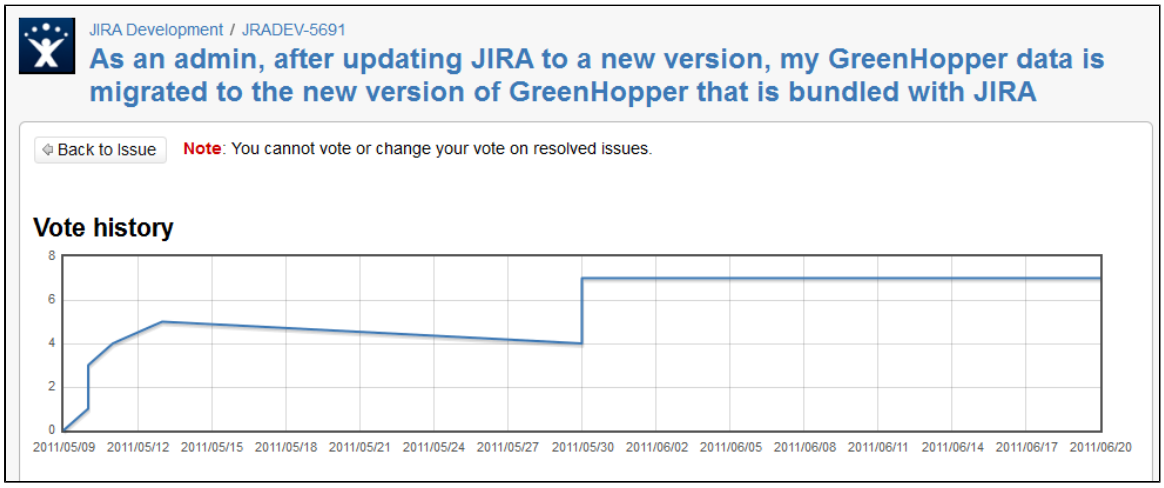
The Activity Stream has had a makeover. As well as looking prettier, it now lets you vote, start watching or comment on an issue with a single click:

Activity Stream
Activity Stream
Tuesday
Andrew Prentice attached one file to ANGRY-73 - bonfire test
iCloud
Tuesday at 5:23 AM Comment Vote Watch
Monday
Mike Cannon-Brookes commented on ANGRY-27 - Nerd flight path does not take into account gravitational pull of moon
Are we building the physics engine for Newtonian or Einsteinian physics?
Monday at 2:15 AM Reply Watch
Sunday
Edwin Wong linked 3 issues
ANGRY-67 - As a nerd, I can flip in a chair while waiting to be shot into the air
is related to
ANGRY-70 - Some graphical glitches and ANGRY-69 - The nerd flies too fast
Sunday at 12:13 AM Vote Watch
Edwin Wong updated 2 fields of ANGRY-67 - As a nerd, I can flip in a chair while waiting to be shot into the air
Logged '2h'
Changed the Remaining Estimate to '0h'
Sunday at 12:13 AM Comment Vote Watch
Edwin Wong attached one file to ANGRY-70 - Some graphical glitches
ANGRY NERDS
Sunday at 12:05 AM Comment Vote Watch
Show more...

^Top

Graph of Vote History

You can now graph an issue's votes over time:



^Top

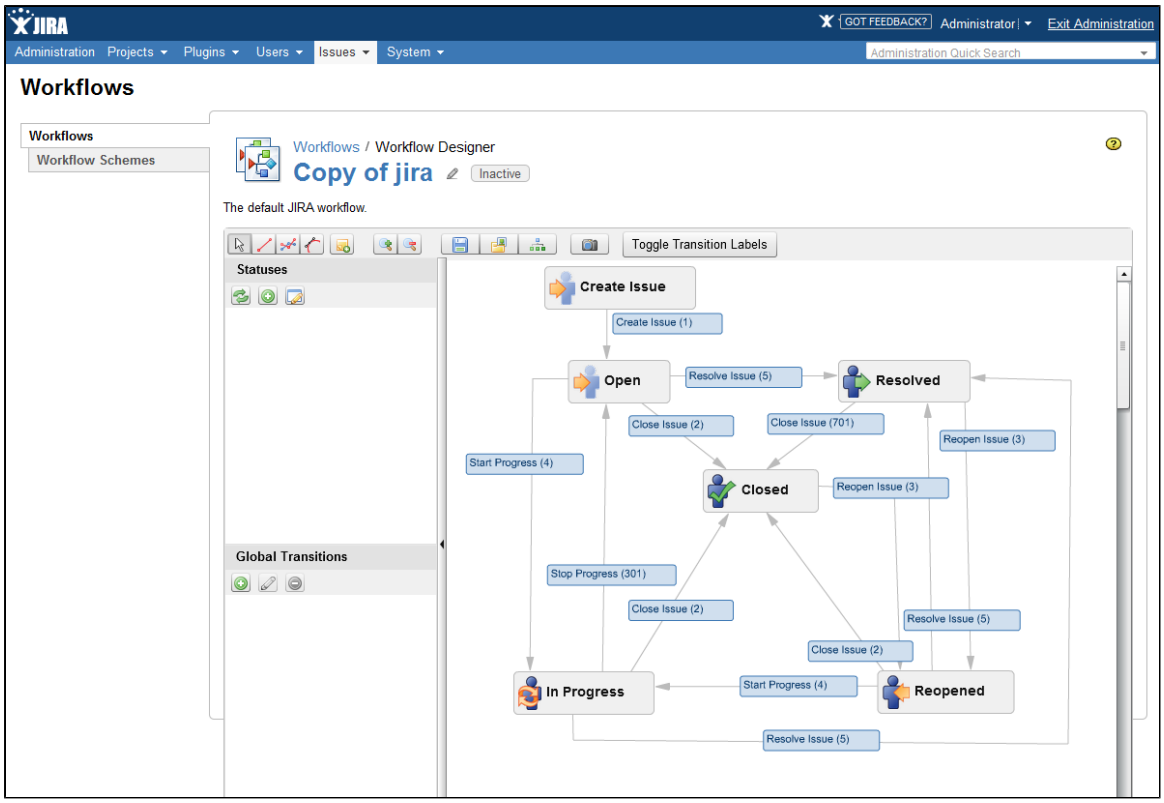
For Administrators:

Visual Workflow Designer

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to '**Workflows**' in JIRA administration as usual, and click the '**Design**' link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.

The layout of a workflow is preserved whenever you 'Copy' or 'Create a Draft' of an existing workflow.



^Top

10

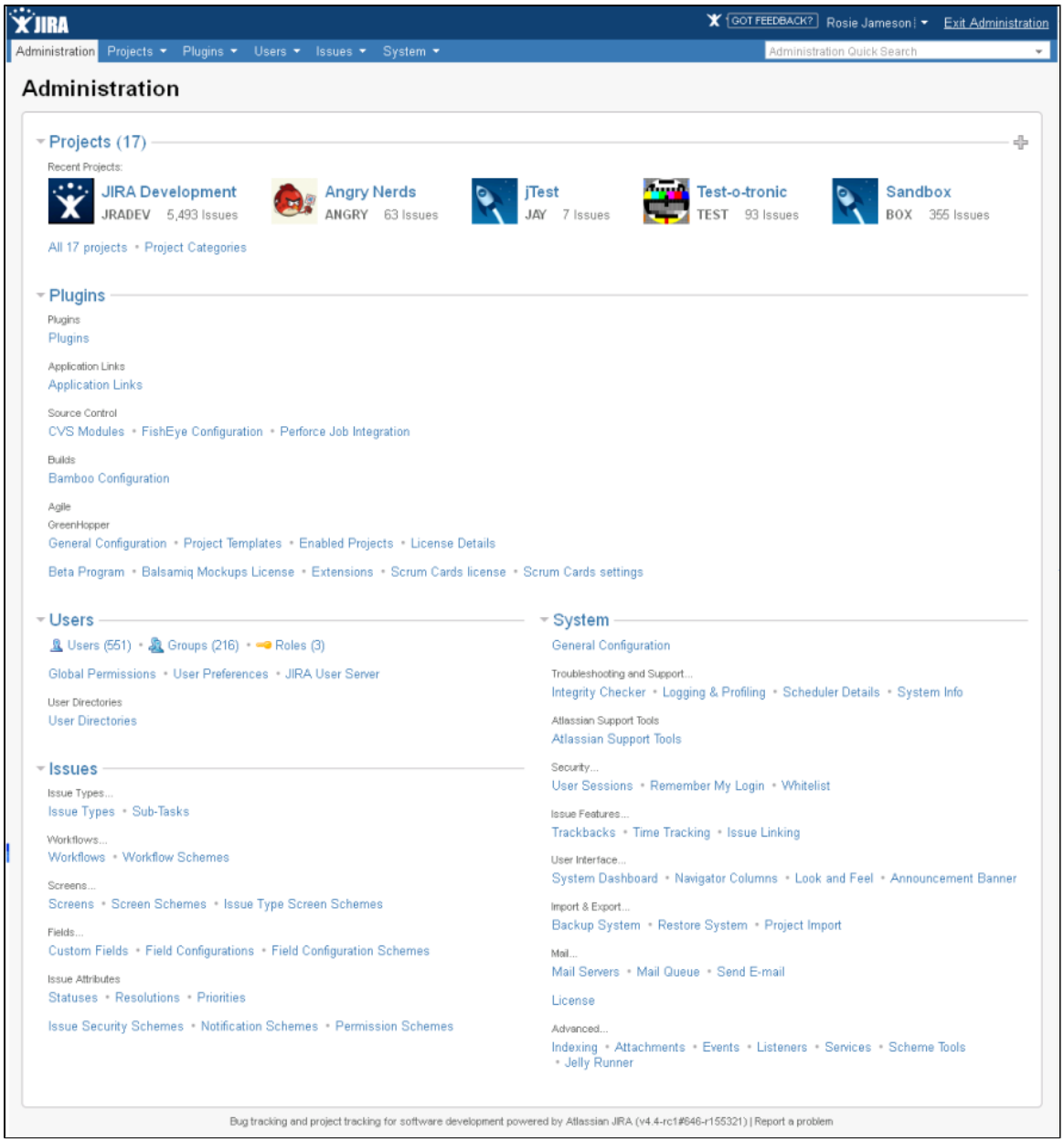
New-Look Administration Area

JIRA 4.4 brings you a dedicated 'administration mode', which replaces the left-hand column of the JIRA's administration console with a series of drop-down menus across the top navigation bar.

To find the new location of a menu item, type it into the **"Administration Quick Search"** box at the top right of the screen — or click the drop-down in the **"Administration Quick Search"** box to get a full list of admin options.

You can bring up the **"Administration Quick Search"** box from anywhere in JIRA by typing **g + g**. This has replaced the **A** keyboard shortcut.

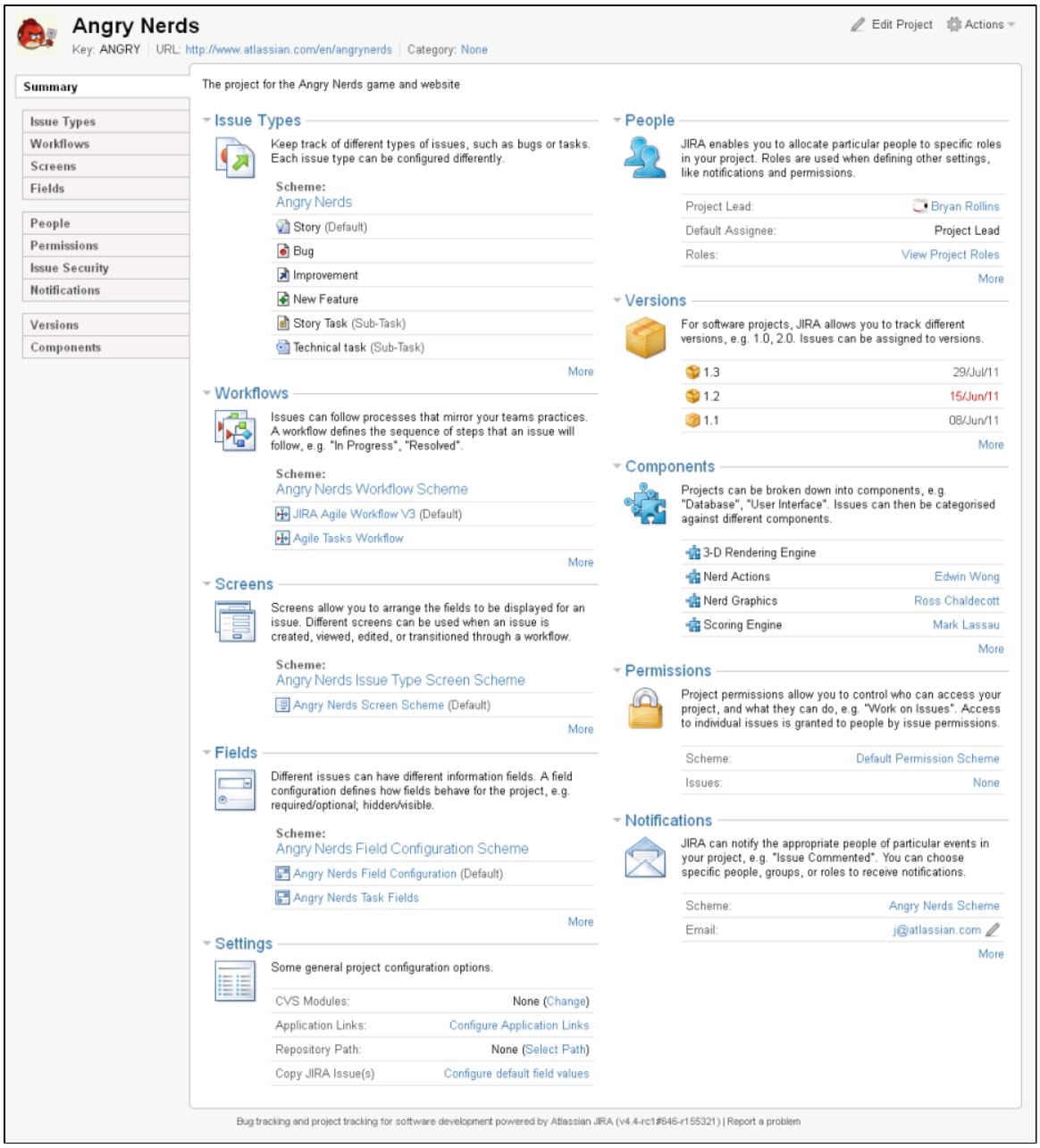
To leave JIRA's 'administration mode', click the '**Exit Administration**' link at the top-right of the screen to return JIRA to its standard user mode.



^Top

Simplified Project Administration

Project administration has become more visual and far simpler:



^Top

Editable Options for Custom Fields

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

^Top

Setup and Installation:

Improved Setup Wizard with Database Configuration

Our trusty Setup Wizard has had a makeover:

The new Setup Wizard

^Top

Database Configuration Now Part of the Setup Wizard

In JIRA 4.4, configuring a connection to an external database is now part of the standard Setup Wizard. Upon completing the Setup Wizard, JIRA will create a direct JDBC connection (whose entire configuration is stored within your JIRA home directory).

Here is the new database configuration step of JIRA's Setup Wizard:

^Top

14

Improved JIRA Standalone Installer/Uninstaller and Automated Upgrade

Linux Installer and Uninstaller

The Linux Installer provides a simple console (shell prompt) wizard that installs JIRA for Linux operating systems. The Linux Installer:

- Installs JIRA under a dedicated user account 'jira' with restricted write access to your JIRA installation directory.
- Can install JIRA as a service, so that JIRA automatically re-starts whenever your Linux operating system must be rebooted.

To install JIRA, simply download the Linux Installer (`.bin`) file and at a shell prompt, execute this file and follow the remaining prompts!

See Installing JIRA on Linux for details.

After using the Linux Installer, an executable file 'uninstall' (located in your JIRA Installation Directory) is available to conveniently uninstall JIRA from your Linux operating system.

See Uninstalling JIRA from Linux for details.

Windows Installer

The installation wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

See Installing JIRA on Windows for details.

Unattended Installation

When installing JIRA using the Linux or Windows Installers (above), a configuration file called `response.varfile` can be generated in the `.install4j` subdirectory of your JIRA Installation Directory.

The `response.varfile` file records all configuration options specified during your initial installation. This allows you to reinstall JIRA on

multiple server machines based on the same configuration without the need for any user input.

See the 'Performing an Unattended Installation' sections for Linux and Windows for details.

Automated Upgrade

The new Linux and Windows Installers (above) include an option that allows you to upgrade an existing JIRA 4.3.x or later Standalone installation.

This upgrade feature automates the following tasks for you:

1. Backs up the Installation and Home Directories of the existing JIRA installation to be upgraded.
2. Installs **JIRA 5.0.x** whilst migrating the following from your existing JIRA installation to the new **JIRA 5.0.x**installation:
    - Legacy database configurations defined as a datasource within the application server (used in JIRA 4.3.x and earlier) to the new database configuration used in JIRA 4.4 and later. See JIRA 4.4 Upgrade Notes for details.
    - TCP port values in your existing JIRA installation's `server.xml` file.
    - Custom values in your existing JIRA installation's `jira-application.properties`, including key customisations from the `setenv.sh` / `setenv.bat` files.

The upgrade feature detects and notifies you of any other files in the `atlassian-jira` subdirectory of your existing JIRA Installation Directory, which had been deleted, added or modified from a 'default' JIRA installation. This informs you of any customisations you will need to migrate manually over to your upgraded JIRA installation directory.

Also note that the JIRA Configuration Tool (bundled with JIRA Standalone) is now capable of changing JIRA's TCP Ports.

^Top

Platforms, APIs:

IE 9 and Firefox 5 Support

We are very pleased to announce that JIRA 4.4 supports Internet Explorer 9.0 and Firefox 5.0.

^Top

REST API improvements

There are new REST APIs for

- Listing and managing Project Components.
- Listing and managing Project Versions.
- Listing and managing Project Roles.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.4 Upgrade Notes

On this page:

- Upgrading from JIRA 4.3 to 4.4
  - Changes to the `jira-application.properties` file
  - Database Configuration Changes
  - Migrating Existing Configurations to Your New JIRA During the Upgrade
    - Upgrading a JIRA 4.3.x Standalone Installation on Linux or Windows
    - Upgrading a JIRA 4.2.x (or Earlier) Standalone or JIRA 4.3.x (or Earlier) WAR Installation
      - Migrating Your Database Configuration
      - Avoid Migrating Your Database Configuration This Way
  - JIRA Configuration Tool
  - New 'View (Read-Only) Workflow' permission
  - External Password Management
  - Issue Linking when Resolving an Issue
  - Old Issue Import Functionality Removed
  - XML Backups Always Generate a ZIP Archive File
  - 'Contact Administrators' Reintroduced
  - Earlier Versions of JIRA with the Visual Workflow Designer Plugin
  - Other Plugins
  - Translations
  - Java Platform Support
  - Known Issues
    - An Exception is Generated After Starting a New JIRA Installation
    - SOAP API Returns IDs Instead of Values for Multiselect Fields
    - Automated Upgrade Encounters Problems Reading the JIRA Installation Directory
    - Other Known Issues
- Upgrading from JIRA 4.2.x and Earlier

Please Note:

From JIRA 4.4, version-specific 'upgrade guides' are now referred to as 'upgrade notes'. This change was made to avoid potential naming ambiguity with the general Upgrading JIRA guide and to match the naming conventions used for the equivalent documents of other Atlassian products.

Upgrading from JIRA 4.3 to 4.4

Please follow the instructions in the general Upgrading JIRA guide, as well as the JIRA 4.4-specific instructions in the sections below. The general 'Upgrading JIRA' guide contains important tasks that are essential for getting your upgraded JIRA installation to work correctly and if necessary, migrating existing configurations.

If you are upgrading a JIRA WAR installation:

Please ensure you read through all content in the section below for Upgrading a JIRA 4.3.x (or Earlier) WAR Installation, including the subsection on **Migrating Your Database Configuration**.

Changes to the `jira-application.properties` file

In JIRA 4.4.0 and later, advanced configuration options are no longer stored in the `jira-application.properties` file. Instead, the default values for nearly all these JIRA properties are stored in the new `jpm.xml` file (located in the JIRA Installation Directory) and any properties whose values you have customised are defined in either the new `jira-config.properties` (located in the JIRA Home Directory) or the JIRA database (for more commonly edited properties). The values of any properties defined in the `jira-config.properties` and JIRA database override those values stored in the `jpm.xml` file. For more information, see Advanced JIRA Configuration.

When upgrading and migrating your existing configurations to JIRA 4.4.x (most of which is handled automatically if you use the new 'automated upgrade' tools), any properties whose values you had customised in the `jira-application.properties` file of your earlier JIRA installation, will be read and migrated across to the new `jira-config.properties` file or JIRA database of your upgraded JIRA installation.

In JIRA 4.4.x, the `jira-application.properties` file is still required, although only the `jira.home` property is used.

Database Configuration Changes

JIRA 4.4.0 and later uses a new file called `dbconfig.xml` (located within your JIRA Home Directory), which defines all JIRA database connection information for your JIRA installation.

Here is an example of a `dbconfig.xml` file defining a *direct JDBC connection* to an external PostgreSQL database:

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>postgres72</database-type>
  <schema-name>public</schema-name>
  <jdbc-datasource>
    <url>jdbc:postgresql://dbserver:5432/jiradb</url>
    <driver-class>org.postgresql.Driver</driver-class>
    <username>jiradbuser</username>
    <password>password</password>
    <pool-size>15</pool-size>
  </jdbc-datasource>
</jira-database-config>
```

A *direct JDBC connection* means that JIRA's entire database configuration information is encapsulated within the `dbconfig.xml` file, with no references to a 'datasource'.

Tip: A `dbconfig.xml` file defining a *direct JDBC connection* to a JIRA database, effectively replaces values which defined the JIRA 'datasource' and 'database' connections from the `<jira-application-dir>/WEB-INF/classes/entityengine.xml` and `conf/server.xml` (JIRA Standalone only) files, respectively (within the JIRA Installation Directory of version 4.3.x or earlier). For JIRA WAR version 4.3.x or earlier, the JIRA database connection is typically defined in the `jira.xml` file within the `conf` subdirectory of the Tomcat installation running JIRA (instead of `server.xml`).

From JIRA 4.4.0, the `entityengine.xml` file is no longer used to store any information about your JIRA database/datasource. However, this file is still required by JIRA for other purposes.

Migrating Existing Configurations to Your New JIRA During the Upgrade

Upgrading a JIRA 4.3.x Standalone Installation on Linux or Windows

Use the upgrade feature of the Linux and Windows Installers to upgrade JIRA.

If you are upgrading a JIRA 4.3.x installation on Solaris, do not use the Linux Installer to upgrade JIRA. Instead, use either the manual (i.e. 'in-place' database upgrade) or migration (i.e. XML export/import) procedures and refer to the additional notes below.

This upgrade feature automates the following tasks for you:

1. Backs up the Installation and Home Directories of the existing JIRA installation to be upgraded.
2. Installs **JIRA 5.0.x** whilst migrating the following from your existing JIRA installation to the new **JIRA 5.0.x**installation:
    - Legacy database configurations defined as a datasource within the application server (used in JIRA 4.3.x and earlier) to the new database configuration used in JIRA 4.4 and later. See Database Configuration Changes (above) for details.
    - TCP port values in your existing JIRA installation's `server.xml` file.
    - Custom values in your existing JIRA installation's `jira-application.properties` and `setenv.sh` / `setenv.bat` files.
        Be aware that in the `setenv.sh` / `setenv.bat`file, only the following values are migrated:
        - `JVM_SUPPORT_RECOMMENDED_ARGS`
        - `JVM_MINIMUM_MEMORY`
        - `JVM_MAXIMUM_MEMORY`
        - `JIRA_MAX_PERM_SIZE`

The upgrade feature detects and notifies you of any files (other than `jira-application.properties` and `setenv.sh` / `setenv.bat`) in the `atlassian-jira` subdirectory of your existing JIRA Installation Directory, which had been deleted, added or modified from a 'default' JIRA installation. This informs you of any customisations you will need to migrate manually over to your upgraded JIRA installation directory.

Please Note:

- The upgrade process requests that you conduct a backup of your database using your database's backup utilities. If your database does not support online backups, you can stop the upgrade process, shut down JIRA, perform your database backup and then restart the upgrade process to continue on.
- If you have made customisations to your `seraph-config.xml` file or any other files in your JIRA installation directory, these must be migrated manually.
- If your attachments and index files are located outside your JIRA Home Directory, then backups of these directories must be performed manually.

Upgrading a JIRA 4.2.x (or Earlier) Standalone or JIRA 4.3.x (or Earlier) WAR Installation

Use either the manual (i.e. 'in-place' database upgrade) or migration (i.e. XML export/import) procedures to upgrade your existing JIRA installation.

When migrating your existing JIRA configurations over to your new JIRA installation, ensure that any specific configurations to the following

files of your existing JIRA installation have been *re-applied* to the equivalent files in your new JIRA installation *before you start your upgraded JIRA installation*:

- `jira-application.properties` (located in the `<jira-application-dir>/WEB-INF/classes` subdirectory of your JIRA Installation Directory.)
- Standalone distributions only — `server.xml` (located in the `conf` subdirectory JIRA Installation Directory).

Migrating Your Database Configuration

Both the manual and migration upgrade procedures require you to have properly configured a `dbconfig.xml` file in your JIRA Home Directory (which defines a *direct JDBC connection* as shown in the example above), *before you start your upgraded JIRA installation*.

If you are creating this `dbconfig.xml` file manually and you are not changing your JIRA database, it may help to migrate specific parameter values from within the following areas of your old JIRA Installation Directory:

- The `<datasource name="defaultDS"/>` element of the `<jira-application-dir>/WEB-INF/classes/entityengine.xml` file
- The `<Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"/>` element of the `conf/server.xml` file.

For JIRA WAR, the `<Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"/>` element is typically located in the `jira.xml` file within the `conf` subdirectory of the Tomcat installation running your old JIRA server.

The following table shows which element values in the `dbconfig.xml` file for JIRA 4.4.x (or later) match the parameter values in the `entityengine.xml` and `server.xml` (or `jira.xml`) files of your JIRA 4.3.x (or earlier) installation:

Element in `dbconfig.xml` for JIRA 4.4.x	**Parameter in `entityengine.xml` from JIRA 4.3.x or earlier**	**Parameter in `server.xml` or `jira.x` from JIRA 4.3.x or earlier**
`<name>defaultDS</name>`	`name="defaultDS"`	-
`<delegator-name>default</delegator-name>`	`jndi-server-name="default"` (within the child `<jndi-jdbc/>` element)	-
`<database-type>hsql</database-type>`	`field-type-name="hsql"`	-
`<schema-name>PUBLIC</schema-name>`	`schema-name="PUBLIC"`	-
`<url>jdbc:hsqldb:C:\Program Files\Atlassian Application Data\JIRA/database/jiradb</url>`	-	`url="jdbc:hsqldb:C:\Program Files\Atlassian Application Data\JIRA/database/jiradb"`
`<driver-class>org.hsqldb.jdbcDriver </driver-class>`	-	`driverClassName="org.hsqldb.j`
`<username>sa</username>`	-	`username="sa"`
`<password></password>`	-	`password=""`
`<pool-size>15</pool-size>`	-	`maxActive="15"`
`<validation-query>select 1</validation-query>`	-	`validationQuery="select 1"`
`<min-evictable-idle-time-millis>4000 </min-evictable-idle-time-millis>`	-	`minEvictableIdleTimeMillis="4`
`<time-between-eviction-runs-millis>5000 </time-between-eviction-runs-millis>`	-	`timeBetweenEvictionRunsMillis=`
`<connection-properties>SetBigStringTryClob= true</connection-properties>`	-	`connectionProperties= "SetBigStringTryClob=true"`

Avoid Migrating Your Database Configuration This Way

If you do not create a `dbconfig.xml` file but instead, re-apply database configurations (in the `entityengine.xml` and `server.xml`/`jira.xml` files) from your old JIRA installation to your new JIRA installation, then upon starting your new JIRA installation, JIRA will write a new `<JIRA Home Directory>/dbconfig.xml` file defining your existing database configuration. However, this `dbconfig.xml` file will actually define a *datasource* connection to JIRA's database (defined in your `server.xml`/`jira.xml` file), as opposed to a *direct JDBC connection* shown in the example above. ***We strongly recommend avoiding this database migration approach***, as it will complicate future upgrades of JIRA.

JIRA Configuration Tool

The JIRA Configuration Tool (included with JIRA 4.4 Standalone) can only work with and modify JIRA 4.4 database configurations (above).

New 'View (Read-Only) Workflow' permission

Users require a new project permission called 'View (Read-Only) Workflow' to access the workflow viewer feature from the 'View Issue' page. On new installations of JIRA, this permission is granted to users automatically. However, for JIRA 4.4 upgrades, JIRA administrators will need to assign this permission to their users manually.

External Password Management

The 'External Password Management' option in the 'General Configuration' area of JIRA has been removed as this is now handled through User Directories.

Issue Linking when Resolving an Issue

On clean installations of JIRA 4.4, when a user resolves or closes an issue which utilises JIRA's default workflow, the user will be able to link that issue to another before submitting the transition.

If you have upgraded to JIRA 4.4, this feature will not automatically be available and it must be configured manually through the JIRA administration console.

To add this feature to JIRA's default workflow, add the new **Linked Issues** field to the **Resolve Issue Screen**. (See Defining a Screen for more information.)

To add this feature to any workflow transition of any other custom workflow, add the **Linked Issues** field to the appropriate screen used by that workflow transition.

Old Issue Import Functionality Removed

The old issue import functionality has been removed and has been completely replaced by the JIRA importers plugin feature, which is included with JIRA.

XML Backups Always Generate a ZIP Archive File

JIRA's XML backup utility now always generates a ZIP archive file. As a result of this change, the option to 'Backup As Zip' has been removed from the user interface.

'Contact Administrators' Reintroduced

The 'Contact Administrators' link has been reintroduced in JIRA 4.4. It operates a little differently from previously — please see the following two options on the 'General Configuration' page:

Contact Administrators Form	Provides an email form for users to fill in when they click the 'Contact Administrators' link (which appears when necessary throughout the JIRA interface, e.g. on the Login screen). Applies only if outgoing email is enabled. Can be used with or without the custom 'Contact Administrators Message' below. *Default: OFF*
Contact Administrators Message	Displays a custom message when users click the 'Contact Administrators' link (which appears when necessary throughout the JIRA interface, e.g. on the Login screen). The 'Contact Administrators Message' will be displayed at the top of the 'Contact Administrators Form', if the form is enabled (see above), or by itself if the form is not enabled.

Earlier Versions of JIRA with the Visual Workflow Designer Plugin

Since the Visual Workflow Designer feature is now included with JIRA 4.4, if you used the JIRA Workflow Designer plugin with your earlier JIRA version, please remove it from your JIRA Home Directory before upgrading JIRA.

If you forget to do this before commencing the upgrade, then conduct the following after upgrading JIRA:

- Shut down your upgraded JIRA installation.
- Remove the old version of the Visual Workflow Designer plugin from your JIRA Home Directory.
- Restart your upgraded JIRA installation.

Other Plugins

JIRA 4.4 introduces several changes that may break existing plugins which are not bundled with JIRA.

If you have a developed a plugin, then please read the Plugin Developer Notes for JIRA 4.4 guide. This guide describes changes in JIRA 4.4 which may affect the compatibility of your plugin with JIRA 4.4.

If you are using a plugin developed by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.4.

Translations

Please note that JIRA 4.4 may not ship with the very latest translations of all languages at this time. However, French, German and other translations can be obtained or updated easily via the Universal Plugin Manager by searching for "language pack". See Managing JIRA's Plugins for details on using this feature.

You can also download and install these language packs manually from our Plugin Exchange site.

✅ Language packs for other languages which are not available through the Universal Plugin Manager or our Plugin Exchange site can be downloaded from our translation site (https://translations.atlassian.com/).

Java Platform Support

Please be aware that Oracle JDK / JRE 1.6 update 18 or later is now required to run JIRA. See Supported Platforms for more information.

Known Issues

An Exception is Generated After Starting a New JIRA Installation

Due to a bug (JRA-24818), an `ArrayIndexOutOfBounds` exception is generated after starting a new JIRA 4.4 installation. However, this exception will not cause any issues with JIRA's functionality.

SOAP API Returns IDs Instead of Values for Multiselect Fields

Due to a bug (JRA-25034), people using the SOAP API to get values for multiselect fields will now just see the "ids" (numbers) of each multiselect value, instead of their human-readable display name. Note that these fields can still be set using values (as opposed to ids).

Our proposed solution is here — please watch the issue for progress and resolution.

Automated Upgrade Encounters Problems Reading the JIRA Installation Directory

The automated upgrade feature of the Windows and Linux installers may report the error: "Problems reading the installation directory." This is usually caused by a missing HTTP connector definition in the `conf/server.xml` file of the JIRA Installation Directory. For more information on how to resolve this issue, please refer to our JIRA Knowledge Base article Unable to Upgrade to JIRA 4.4 due to Problems Reading the Installation Directory.

Other Known Issues

Before you begin the upgrade, please check for known issues. Sometimes we find out about a problem with the latest version of JIRA after we have released the software. In such cases we publish information about the known issues in the JIRA Knowledge Base. Please check for known issues and follow the instructions to apply any necessary patches.

If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Upgrading from JIRA 4.2.x and Earlier

In addition to the points listed above, please read the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

Plugin Developer Notes for JIRA 4.4

On this page:

- Introduction
- PluggableProjectOperation
- Formatting and Parsing Dates Using the Appropriate Time Zone
  - Notes About `OutlookDate` and its Deprecation
- Single- and Multi-Select Custom Field Changes
  - Database Changes
  - Effect on Existing Plugins
- Restrictions on the Alias Names of plugin Webwork Actions
- View Issue Content & Project Admin Summary is now Pluggable via Web Panels
  - Adding action icons to the header
  - Adding links to the dropdown.
  - Adding sections to the dropdown
- Web-fragments and the administration navigation changes
  - Tab navigation in admin section
- Plugging in to the new Project Administration
- Version-related Atlassian Events
- Gadget Web-Resources

- As of JIRA 4.4, *all* plugin modules are now dynamically reloadable
- REST API Changes in JIRA 4.4 (from JIRA 4.3)
  - Retrieving a List of Groups
  - Deleting a Watcher from an Issue
  - Create, Read, Update or Delete Project Components
  - Create, Read, Update or Delete Project Versions
  - Date Format Improvements

Introduction

JIRA 4.4 introduces several changes that may break existing plugins which are not bundled with JIRA.

If you are using or have been involved in the development of such a plugin, it may need to be updated to work with JIRA 4.4. Please read through the information below to see if any of this content is relevant to your plugin.

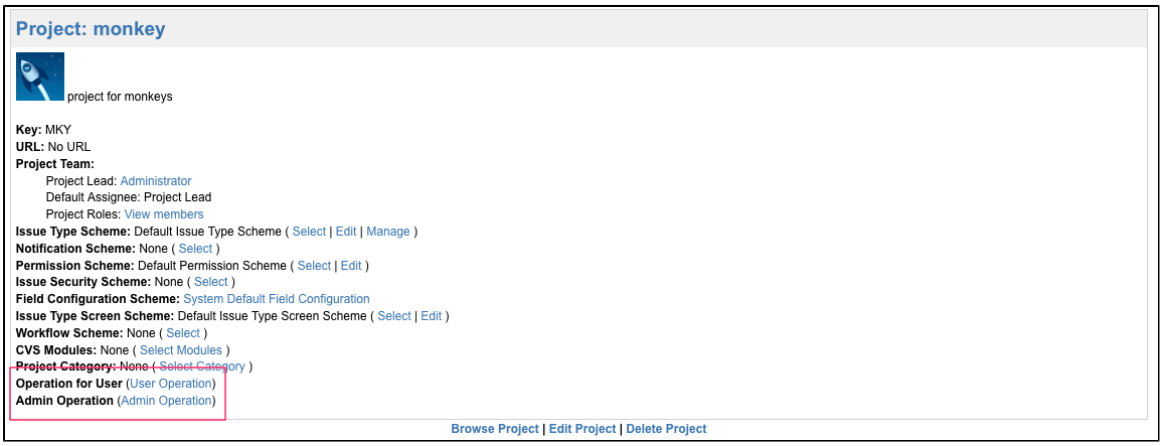
If you are using a plugin developed by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.4.

Please Note:

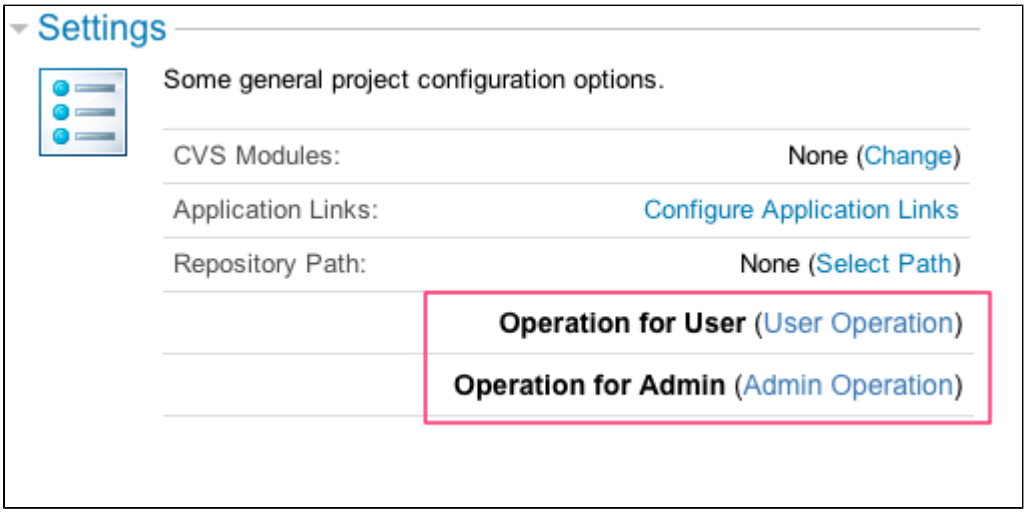
- *This is not the complete list of changes for JIRA 4.4 — it only describes changes in JIRA 4.4 that will impact plugin developers.*
- For details about which versions of Atlassian's Plugin Development Platform and its components (such as the Plugin Framework, Shared Access Layer (SAL), Atlassian User Interface (AUI) and the Atlassian REST Plugin) are included in JIRA 4.4, please refer to Plugin Development Platform Version Matrix.

PluggableProjectOperation

The `PluggableProjectOperation` interface allows plugin developers to add links to the project admin section. For example:



These plugin points are now rendered in a different location within JIRA. While the plugins will continue to render they will not look very nice. For example:



To make the plugins fit into the new panel the markup they generate will have to change. The new template looks something like:

```
<span class="project-config-list-label">Operation for User</span>
<span class="project-config-list-value"><a href="#">User Operation</a></span>
```

This markup is encapsulated in `DefaultPluggableProjectOperation`. You can extend this class and provide implementations of the `getLabelHtml` and `getContentHtml` methods in your plugin. For example:

```
public class ExampleOperation extends DefaultPluggableProjectOperation
{
    public boolean showOperation(Project project, User user)
    {
        return true;
    }

    protected String getLabelHtml(Project project, User user)
    {
        return descriptor.getI18nBean().getText("com.example.plugin.projectop.label");
    }

    protected String getContentHtml(Project project, User user)
    {
        return descriptor.getHtml("view", ImmutableMap.of("some", "parameter"));
    }
}
```

This class should isolate you from small changes to the markup necessary to show your operation on the project summary page.

Formatting and Parsing Dates Using the Appropriate Time Zone

JIRA 4.4 introduces the concepts of **user time zone** and **default user time zone**. JIRA will attempt to use the **user time zone** when displaying dates to a user and similarly when interpreting dates entered by the user. If a user has not specified a time zone in their user profile, JIRA will fall back to the **default user time zone**, which can be configured by a JIRA administrator.

(Note: Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information), and therefore remain in the default server timezone. If plugins are displaying these values, they might be inconsistent with JIRA otherwise, especially if they use OutlookDate which applies the user timezone.)

In order to provide a consistent user experience, plugins that target JIRA 4.4 should be mindful of the time zone that is in use when parsing and formatting dates. It is not enough to instantiate a `java.text.SimpleDateFormat` or a `org.joda.time.format.DateTimeFormatter` class, as these classes will use the default JVM time zone and locale, which may not necessarily match the user's specified time zone in JIRA.

From JIRA 4.4, the recommended way of formatting and parsing dates is to use a `com.atlassian.jira.datetime.DateTimeFormatter`. Here is a plugin class example that creates a couple of formatters when the plugin is started, for use at a later point in time:

```
class MyPluginClass
{
    private final DateTimeFormatter defaultFormatter;
    private final DateTimeFormatter userFormatter;

    public MyPluginClass(DateTimeFormatterFactory factory)
    {
        defaultFormatter = factory.formatter().withStyle(DateTimeStyle.DATE);
        userFormatter = defaultFormatter.forLoggedInUser();
    }

    public String getDateInDefaultTimeZoneAndLocale()
    {
        // uses the default JIRA locale and time zone
        return defaultFormatter.format(new Date());
    }

    public String getDateForLoggedInUser()
    {
        // looks up the currently logged in user, and uses his/her time zone and locale
        return userFormatter.format(new Date());
    }
}
```

Notes About `OutlookDate` and its Deprecation

The `OutlookDate` class has been retrofitted to account for the new **user time zone** and **default user time zone** concepts. Hence, plugins that use this class to display and parse time zones will automatically take on this new user time zone behaviour. While this is generally the

desired behaviour, plugin developers can override the time zone and locale used in the `com.atlassian.jira.datetime.DateTimeFormatter` class.

`OutlookDate` has been deprecated and will be removed in a future version of JIRA. We encourage plugin writers to port their plugins over to the new `com.atlassian.jira.datetime.DateTimeFormatter` API as soon as possible.

Single- and Multi-Select Custom Field Changes

Since the following single- and multi-select custom fields are now editable, you should be aware of the changes below which may affect your plugins.

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

See JRA-2983 for more information about this JIRA improvement.

Database Changes

There are two database changes that result from this improvement:

1. We now store the ID (as opposed to the literal value) of the custom field option in the `customfieldvalue` table.
2. We have added a 'disabled' flag to the `customfieldoption` table. When this flag is set to 'true' for a given option, that option is not available for selection when creating or editing an issue. You should honour this behaviour when using or extending your single- and multi-select custom field types. Disabled values are still available for searching.

Effect on Existing Plugins

This improvement will affect plugins that provide custom fields (of type select or multi-select) which reuse or extend the following classes:

- `com.atlassian.jira.issue.customfields.impl.SelectCFType` and
- `com.atlassian.jira.issue.customfields.impl.MultiSelectCFType`

If your plugin extends these types, you may need to adjust its behaviour. Velocity templates for editing and searching should be adjusted to accept and return the custom field option ID in the `value` attribute.

If your plugin consumes the output of the com.atlassian.jira.issue.Issue.getCustomFieldValue() you should be aware that this method now returns a com.atlassian.jira.issue.customfields.option.Option object rather than a String for Select or MultiSelect custom fields. You can use the Option.toString() method to make your pugin compatible with both JIRA 4.4 and earlier versions.

Please note:

- Only select and multi-select custom field types are affected by this change. Any custom fields of these types will continue to operate correctly, as the database will be updated during the upgrade to 4.4. Third party custom field types that extend the select and multi-select custom field types may break, but that depends upon their implementation, for example if they provided different templates for viewing or editing these would need to be updated.
- Plugins that provide custom field types that extend select and multi-select custom field types, can build a plugin that is compatible with both JIRA 4.3 and JIRA 4.4 but that may not be practical in some cases and it may be easier to have separate versions.

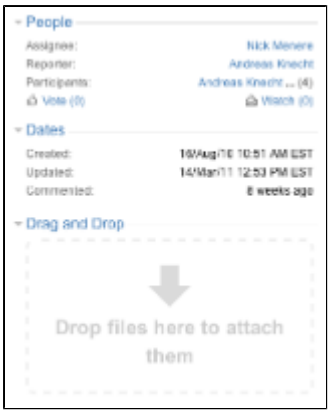
Restrictions on the Alias Names of plugin Webwork Actions

In JIRA 4.4 we have restricted the alias names that you can use for your Webwork actions in a small way. It can't start with 'webwork.'. So for example you cant have an action aliased as 'webwork.MyAction.jspa'.

We did this because for performance reason Webwork calls down to all layers to find webwork properties such as 'webwork.action.prefix' and so on as well as trying to find action aliases. This results in hundreds of calls to plugins for an answer we never intended that they answer. So we have limited the times they will be asked via this string matching rule.

View Issue Content & Project Admin Summary is now Pluggable via Web Panels

We have converted the right-hand side of the View Issue page and the entire Project Admin Summary page to use Web Panels for rendering content. This has enabled us to make this a plugin point for plugin writers.
E.g.



The location for the webpanel needs to be: "atl.jira.view.issue.right.context" for the View Issue page. In JIRA 5.0 we aim to make the left hand side pluggable as well.

✅ For the Admin Project Summary page use "webpanels.admin.summary.left-panels" or "webpanels.admin.summary.right-panels".

Here is how the Time Tracking block is created:

```
<!-- Time tracking web panel -->
    <web-panel key="timetrackingmodule" location="atl.jira.view.issue.right.context"
weight="60">
        <context-provider
class="com.atlassian.jira.plugin.viewissue.TimeTrackingViewIssueContextProvider"/>
        <resource name="view" type="velocity" location="timetracking/timetracking.vm"/>

        <label key="timetracking.title" />
        <condition
class="com.atlassian.jira.plugin.webfragment.conditions.IsFieldHiddenCondition" invert="true">
            <param name="field">timetracking</param>
        </condition>
        <condition
class="com.atlassian.jira.plugin.webfragment.conditions.TimeTrackingEnabledCondition"/>
    </web-panel>
```

In this example:

- The context provider, TimeTrackingViewIssueContextProvider, populates the context for the velocity template.
- The velocity template, timetracking.vm, is responsible for rendering the block.
- The label provides the label for the block.
- The conditions are evaluated to determine if we should show the block.

The Web Panel is responsible for providing the content inside the "module" chrome. The module chrome provides the block heading (from the Web Panel label), the collapsable states and further plugin points for the header. If you do not wish to have this chrome rendered for you (e.g. you may just want to include some javascript on the page), you need to specify the following inside your module descriptor:

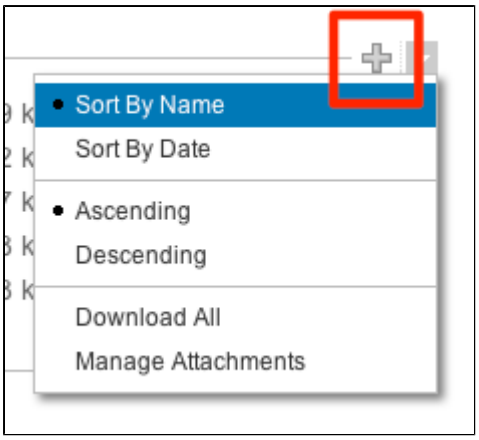
```
<param name="headless">true</param>
```

This will just put the exact output of your Web Panel into the page.

The additional plugin points the common chrome provides are:

Adding action icons to the header

These are actually just styled up Web Items.



The location is: "<web-panel-full-key>/header".

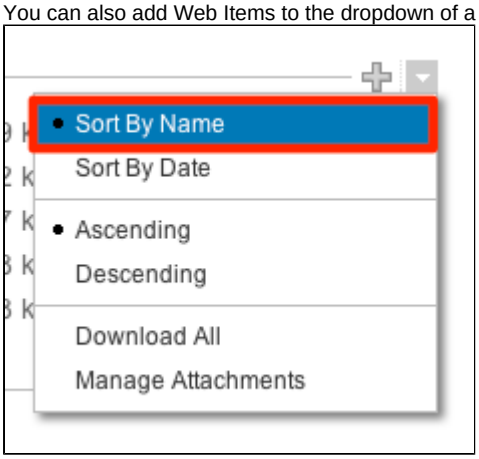
Here is how we render the "Add Attachment icon":

```
<!--  Add Attachment link -->
    <web-item key="add-attachments-link"
i18n-name-key="webfragments.view.issue.attachments.create" name="Create an Attachments"
section="com.atlassian.jira.jira-view-issue-plugin:attachmentmodule/header" weight="1">
        <label key="admin.issue.operations.plugin.attach.file.name"/>
        <tooltip key="admin.issue.operations.plugin.attach.file.name"/>
        <styleClass>issueaction-attach-file icon icon-add16</styleClass>
        <link linkId="add-attachments-link">
            /secure/AttachFile!default.jspa?id=${issue.id}
        </link>
        <condition
class="com.atlassian.jira.plugin.webfragment.conditions.CanAttachFileToIssueCondition"/>
    </web-item>
```

Note the styleClass in this example as it is responsible for styling the link as an icon.

Adding links to the dropdown.

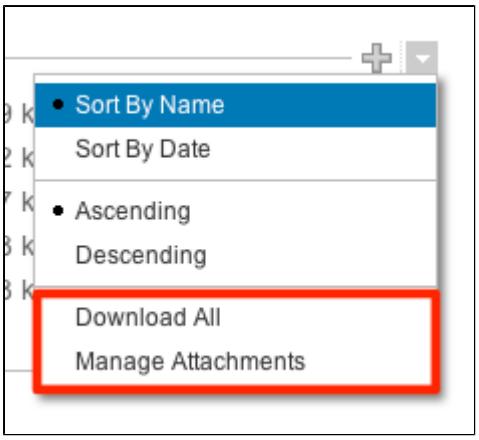
You can also add Web Items to the dropdown of a block.



The default location of these are: "<web-panel-full-key>/drop/default".
E.g. "com.atlassian.jira.jira-view-issue-plugin:attachmentmodule/drop/default"

Adding sections to the dropdown

You can also define Web Sections for the dropdown, to group Web Items together.



The location for these are: "<web-panel-full-key>/drop".
E.g. "com.atlassian.jira.jira-view-issue-plugin:attachmentmodule/drop"
To add Web Items to these sections, use the location: "<web-panel-full-key>/drop/<section-key>".
E.g. "com.atlassian.jira.jira-view-issue-plugin:attachmentmodule/drop/attachment-sorting-options"

Web-fragments and the administration navigation changes

Traditionally navigation in the administration part of JIRA was done in a huge list of links down the left hand side of the page. We have moved to having an administration mode with menus at the top of the page. These menus follow a new, simpler way to generate themselves and web-items and web-sections in existing plugins will need to be updated to fit with the new structure. Unchanged web-items will still show in the menus, all together at the bottom of the "Plugins" menu.

The new structure follows the rules:

- Any level of web-section can have both or either web-items and web-sections in it.
- The *location* attribute for a web-section is the web-section it is inside of.
- The *section* attribute of a web-item is the locationOfTheSection/section.

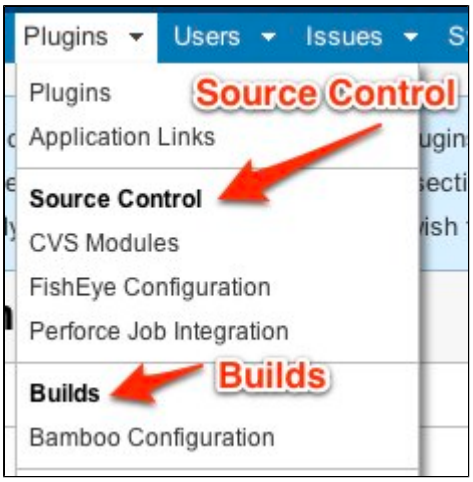
The new structure has the new location of *system.admin.top.navigation.bar* and a section for each menu at that location. There are 2nd level web-sections for each menu under that. 3rd level web-sections for each section within a menu and 4th level web-sections so that a set of web-items that will appear as tabs at the side will appear as a single item in the top level menus, this menu item will link to the web-item with the lowest weight in that web-section.

It is **strongly** recommended to put all plugins web-sections and web-items in the Plugin menu (location *admin_plugins_menu*) and we have created some predefined sections to put you to use:

Section Name	web-item's section attribute	web-section's location attribute
Source Control	admin_plugins_menu/source_control	source_control
Builds	admin_plugins_menu/builds_section	builds_section
Agile	admin_plugins_menu/agile_section	agile_section
Testing	admin_plugins_menu/testing_section	testing_section
Requirements	admin_plugins_menu/requirements_section	requirements_section
Timetracking	admin_plugins_menu/timetracking_section	timetracking_section
Integrations	admin_plugins_menu/integrations_section	integrations_section
Workflow	admin_plugins_menu/workflow_section	workflow_section
Drawing	admin_plugins_menu/drawing_section	drawing_section

Adding web-items and sections to these new sections means they will also be rendered in the administration summary page.

The Source Control and Builds sections of the Plugins menu are shown below:



And example of adding a web-item for Bamboo Configuration to the Builds section would be:

```
    <web-item key="bambooConfigLink" name="Bamboo Config Web Item"
section="admin_plugins_menu/builds_section" weight="10">
        <condition
class="com.atlassian.jira.plugin.webfragment.conditions.JiraGlobalPermissionCondition">
            <param name="permission">admin</param>
        </condition>
        <label key="bamboo.config.title"/>
        <link linkId="bamboo_config">/secure/admin/jira/ViewBambooApplicationLinks.jspa</link>
    </web-item>
```

If you want to keep your plugin compatible with versions of JIRA prior to 4.4, you can add a condition to your existing web-item for older versions of JIRA so that it will only show up for them. For example in the Bamboo Plugin we have a web-item:

```
<web-item key="bambooConfigLink-old" name="Bamboo Config Web Item"
section="system.admin/globalsettings" weight="19">
   <condition
class="com.atlassian.jira.plugin.webfragment.conditions.JiraGlobalPermissionCondition">
      <param name="permission">admin</param>
   </condition>
   <condition class="com.atlassian.jira.plugin.ext.bamboo.conditions.IsPriorToJiraVersion">
      <param name="majorVersion">4</param>
      <param name="minorVersion">4</param>
   </condition>
   <label key="bamboo.config.title"/>
   <link linkId="bamboo_config">/secure/admin/jira/ViewBambooApplicationLinks.jspa</link>
</web-item>
```

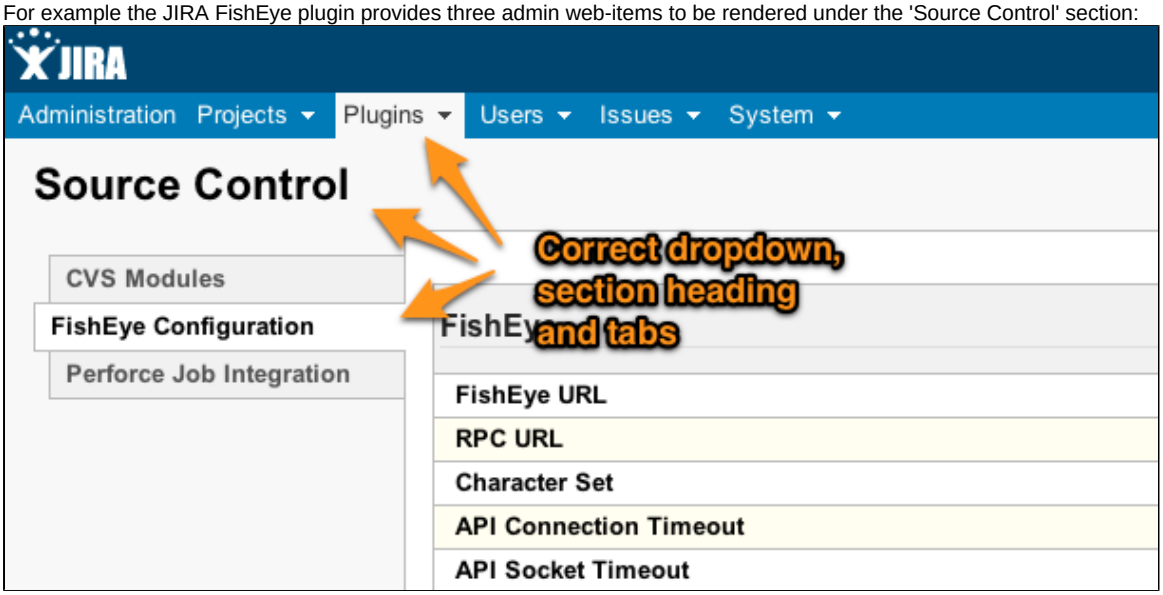
With the IsPriorToJiraVersion is:

```
public class IsPriorToJiraVersion implements Condition {

    private int maxMajorVersion;
    private int maxMinorVersion;
    private int majorVersion;
    private int minorVersion;

    public IsPriorToJiraVersion(ApplicationProperties applicationProperties) {
        String versionString = applicationProperties.getVersion();
        String versionRegex = "^(\\d+)\\.(\\d+)";
        Pattern versionPattern = Pattern.compile(versionRegex);
        Matcher versionMatcher = versionPattern.matcher(versionString);
        versionMatcher.find();
        majorVersion = Integer.decode(versionMatcher.group(1));
        minorVersion = Integer.decode(versionMatcher.group(2));
    }

    public void init(final Map<String, String> paramMap) throws PluginParseException {
        maxMajorVersion = Integer.decode(paramMap.get("majorVersion"));
        maxMinorVersion = Integer.decode(paramMap.get("minorVersion"));
    }

    public boolean shouldDisplay(final Map<String, Object> context) {
        return (majorVersion < maxMajorVersion) || (majorVersion == maxMajorVersion) &&
(minorVersion < maxMinorVersion);
    }
}
```

Tab navigation in admin section

To ensure that the new admin decorator can highlight the correct dropdown in the header and render the correct tabs on the left hand side, the page being decorated needs to tell the decorator which admin section it belongs to. This is done through the use of `<meta/>` tags.

For example the JIRA FishEye plugin provides three admin web-items to be rendered under the 'Source Control' section:



In order for this to work correctly, the page source for the FishEye Configuration page has to include the following `<meta/>` tags to tell the new admin decorator which tabs to render:

```
<meta name="admin.active.section" content="admin_plugins_menu/source_control">
<meta name="admin.active.tab" content="fisheye_config">
```

These meta tags will have to be included on **every** one of your admin pages and will have to refer back to the web-items you defined earlier.

Plugging in to the new Project Administration

This is fairly similar to the integrating with the new project tab look and feel.

To add content to the Summary tab, see above.

Instead it is a simple web-section//web-item, for example:

```
<web-section key="yoursection" name="Your Config Group" location="atl.jira.proj.config"
i18n-name-key="ayoukey" weight="50"/>
```

Or you can use the inbuilt sections - projectgroup1, projectgroup2, projectgroup3, projectgroup4
Then add a web-item to it.

```
<web-item key="view_project_your_tab" name="Your Tab"
section="atl.jira.proj.config/yoursection" i18n-name-key="yourkey" weight="10">
  <label key="your_label" />
  <link linkId="view_project_your_tab">a_link_to_your_content</link>
</web-item>
```

That is enough to add your tab.

To then wrap your content with the project config decorator and have the right tab selected, your content should have the following in its header:

```
<head>
  <title>$title</title>
  <meta name="decorator" content="admin"/>
  <meta name="projectKey" content="$project.key"/>
  <meta name="projectId" content="$project.id"/>
  <meta name="admin.active.tab" content="id_of_your_tab"/>
  <meta name="admin.active.section" content="atl.jira.proj.config"/>
</head>
```

First create a tab group (web-section) you want to add to:

```
<web-section key="your_section_key" name="Your Config Group" location="atl.jira.proj.config"
i18n-name-key="your.i18n.key" weight="50"/>
```

or you can use an existing section (weights are in brackets after sections & items):

- **projectgroup1** (10) - Contains Summary(10)
- **projectgroup2** (20) - Contains Issue Types(10), Workflows(20), Screens(30) & Fields(40)
- **projectgroup3** (30) - Contains People(10), Permissions(20), Issue Security(30) & Notifications(40)
- **projectgroup4** (40) - Contains Versions(10) & Components(20)

Add a web-item to your section.

```
<web-item key="your_tab_key" name="Your Tab" section="atl.jira.proj.config/your_section_key"
i18n-name-key="your.i18n.key" weight="10">
  <label key="your_label" />
  <link linkId="view_project_your_tab">a_link_to_your_content</link>
</web-item>
```

Now you will have a tab in Project Configuration. The link should point to some content. This can be provided by a WebWork Action, REST resource or Servlet.

To then wrap your content with the project config decorator and have the right tab selected, your content should have the following in its header:

```
<head>
    <title>Your Page Title</title>
    <meta name="decorator" content="admin"/>
    <meta name="projectKey" content="key_of_active_project"/>
    <meta name="projectId" content="id_of_active_project"/>
    <meta name="admin.active.tab" content="view_project_your_tab"/>
    <meta name="admin.active.section" content="atl.jira.proj.config"/>
</head>
```

These meta tags will have to be included on **every** one of your project admin pages

Version-related Atlassian Events

The following new Atlassian Events are available in JIRA 4.4:

- `VersionMergeEvent`
- `VersionDeleteEvent`
- `VersionCreateEvent`
- `VersionReleaseEvent`
- `VersionUnreleaseEvent`
- `VersionArchiveEvent`
- `VersionUnarchiveEvent`
- `VersionMoveEvent`

See JIRA-specific Atlassian Events for a complete list.

Gadget Web-Resources

As of JIRA 4.4 gadgets should no longer depend on the `com.atlassian.jira.gadgets:common` web resource. This change was introduced in JIRA 4.3, however due to some changes to the gadgets framework in JIRA 4.4 gadgets still depending on this web-resource will now break.

Instead gadgets should only need to rely on the more lightweight `com.atlassian.jira.gadgets:common-lite` web resource.

For more information, please refer to JRA-25039.

As of JIRA 4.4, *all* plugin modules are now dynamically reloadable

Traditionally some of the modules were bearing the `@RequiresRestart` annotation, which indicated that installing plugins containing such modules required a full restart of JIRA. As of JIRA 4.4 *each* plugin module is capable of being dynamically added and removed from the running JIRA instance. This has some important implications for the plugin developers, encapsulated by the following recommendations:

- avoid using static caches or statically accessible components (the *singleton* pattern) in your plugins - use component module type instead
- avoid manipulating / storing class loaders and threads in your plugin components - this *may* potentially lead to memory leaks given your plugin is installed and taken down from running JIRA instance multiple times
- the following life-cycle interfaces are available for your components to implement and will be called upon relevant plugin life cycle events:
    - `org.springframework.beans.factory.InitializingBean` - its `afterPropertiesSet()` method will be called each time the Spring context for given plugin is created, i.e. each time the plugin is enabled. **NOTE** this method is called immediately after given component instance is created and fully initialized, and not after the plugin has been fully enabled. In fact, when the whole plugin system is starting (as opposed to just your plugin being enabled), this is likely to be called *before* the plugin system has been initialized. Therefore you cannot rely on the plugin system state when implementing this interface. The preferred way in such case is to listen to Atlassian plugin framework events and use this method to only to register itself with the `EventPublisher`
    - `org.springframework.beans.factory.DisposableBean` - its `destroy()` method will be called when the component is being destroy as its Spring context is being removed. Similarly to the `InitializingBean`, implementing classes *cannot* rely on the state of the plugin system (e.g. calling other components within `destroy()` may cause exceptions as those components may have already be removed from the destroyed context). Please use this method sparingly, e.g. to unregister the component from the `EventPublisher`, and otherwise use the plugin framework events instead
    - com.atlassian.sal.api.lifecycle.LifecycleAware (see javadoc) - called when the plugin system starts up
- avoid implementing `com.atlassian.jira.extension.Startable` in you plugins, it is meant to be used by internal JIRA components only and its support for plugins will soon be removed.

REST API Changes in JIRA 4.4 (from JIRA 4.3)

Retrieving a List of Groups

You can now retrieve a list of all groups in a JIRA installation, as well as a filtered list of groups matching a specified 'query' substring using the following HTTP GET action on:

- `http://hostname/rest/api/2.0.alpha1/groups/picker`

For example, `http://hostname/rest/api/2.0.alpha1/groups/picker?query=admin` will retrieve a list of groups containing the 'admin' substring. Refer to the REST API documentation for more information.

Deleting a Watcher from an Issue

To delete a watcher from an issue, the REST API call format has changed.

For example, to delete a user with username 'fred' from issue 'PROJ-123', you would use the following formats:

- In JIRA 4.3 (and earlier) — `http://hostname/rest/api/2.0.alpha1/issue/PROJ-123/watchers/fred`
- In JIRA 4.4 (and later) — `http://hostname/rest/api/2.0.alpha1/issue/PROJ-123/watchers?fred`

Refer to the REST API documentation for more information.

Create, Read, Update or Delete Project Components

You can create, read, update or delete project components using the following REST calls:

Action	REST API call
Create a project component	HTTP POST on `http://hostname/rest/api/2.0.alpha1/component` http://docs.atlassian.com/jira/REST/4.4/#id2475780
Read/get a project component along with the total number of issues with that component	HTTP GET on `http://hostname/rest/api/2.0.alpha1/component/{id}/relatedIssueCounts` http://docs.atlassian.com/jira/REST/4.4/#id2475988
Read/get a detailed list of information about a project component	HTTP GET on `http://hostname/rest/api/2.0.alpha1/component/{id}` http://docs.atlassian.com/jira/REST/4.4/#id2475844
Modify a project component	HTTP PUT on `http://hostname/rest/api/2.0.alpha1/component/{id}` http://docs.atlassian.com/jira/REST/4.4/#id2475844
Delete a project component	HTTP DELETE on `http://hostname/rest/api/2.0.alpha1/component/{id}?moveIssuesTo` http://docs.atlassian.com/jira/REST/4.4/#id2475844 (You can assign any issues associated with the project component being deleted (i.e. `{id}`) to another component specified as the value of the `moveIssuesTo` parameter.)

Create, Read, Update or Delete Project Versions

You can create, read, update or delete project versions using the following REST calls:

Action	REST API call
Create a project version	HTTP POST on `http://hostname/rest/api/2.0.alpha1/version` http://docs.atlassian.com/jira/REST/4.4/#id2474955
Read/get a list of information about a project version	HTTP GET on `http://hostname/rest/api/2.0.alpha1/version/{id}` http://docs.atlassian.com/jira/REST/4.4/#id2475026
Read/get a project version along with the total number of issues fixed and affected in that version	HTTP GET on `http://hostname/rest/api/2.0.alpha1/version/{id}/relatedIssueCounts` http://docs.atlassian.com/jira/REST/4.4/#id2475180
Read/get a project version along with the total number of unresolved issues in that version	HTTP GET on `http://hostname/rest/api/2.0.alpha1/version/{id}/unresolvedIssueCount` http://docs.atlassian.com/jira/REST/4.4/#id2475234
Modify a project version's sequence within the project	HTTP PUT on `http://hostname/rest/api/2.0.alpha1/component/{id}/move` http://docs.atlassian.com/jira/REST/4.4/#id2475287
Delete a project version	HTTP DELETE on `http://hostname/rest/api/2.0.alpha1/version/{id}` http://docs.atlassian.com/jira/REST/4.4/#id2475026

Date Format Improvements

Due dates and custom field dates are now only parsed/presented in a simple year, month and day format. Additional timezone-specific content is no longer required, nor expected.

JIRA 4.4.5 Release Notes

22 February 2012

The Atlassian JIRA team announces the release of **JIRA 4.4.5**. This point release contains several updates and fixes.

Upgrading to JIRA 4.4.5 is free to all customers with active JIRA software maintenance.

Don't have JIRA 4.4 yet?

Take a look at all the new features in the JIRA 4.4 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.4.5 Upgrade Notes.

Updates and Fixes in this Release

JIRA 4.4.5 includes the following updates and bug fixes:

JIRA Issues (22 issues)

Type	Key	Summary	Priority	Status
	JRA-22256	Performance Issue with JQL functions		Resolved
	JRA-26846	"Select List" custom field values are not escaped in CDATA in issue XML view, leading to invalid XML		Resolved
	JRA-26707	SOAP progress issue fails with custom fields updates for select like field types		Resolved
	JRA-26954	Attaching files on Create Issue Page does not work.		Resolved
	JRA-26635	"Error importing data: java.lang.NullPointerException" when importing in JIRA		Resolved
	JRA-26422	Issue Navigator inline Actions not performed on selected Issue		Resolved
	JRA-25915	Deleting User from AD breaks searches in JIRA, but doesn't break the existing filters with the same search		Resolved
	JRA-25904	Issue link dialog jumps to issue instead of returning to the issue navigator when linking from the navigator		Resolved
	JRA-25622	Admin menu keyboard navigation doesn't work correctly		Resolved
	JRA-25405	REST login and REST logout is broken because Seraph wont keeps its nose out of the way		Resolved
	JRA-25284	module-descriptor reloadabiity in CustomFieldSearcherModuleDescriptor		Resolved
	JRA-24979	The gadget directory code will wait forever on the gadget provider and this can hang the ability to add gadgets		Resolved
	JRA-20445	Some Gadget configurations can't be saved when running JIRA and selecting Turkish as region format		Resolved
	JRA-20054	Total issue count is not matching for pie chart gadget configured with filter and statistic type as multivalue field.		Resolved
	JRA-26483	com.atlassian.jira.rest.api.util.ValidationError has non-public constructor		Resolved
	JRA-26377	Exception getting message body from Velocity		

				Resolved
	JRA-26167	When a user is deleted from AD and is the value of a user custom field the field does not show for that issue		Resolved
	JRA-25760	HTTP/HTTPS security warning popups on IE 8 when moving issue between projects with different Issue Type Schemes		Resolved
	JRA-25572	Search Original Estimate using unit "d" doesn't work in Advance Search		Resolved
	JRA-25178	Autocomplete on issue links, user pickers, etc. should be cancelled when form field is exited		Resolved
	JRA-24074	Gadgets - Can't configure Average Number of Times in Status gadget		Resolved
	JRA-21586	Prevent duplicate form actions from happening		Resolved

JIRA 4.4.5 Upgrade Notes

Upgrading from JIRA 4.4.x to 4.4.5

Please follow the instructions in the general upgrading JIRA documentation.

Upgrading from JIRA 4.3.x and earlier

In addition to the above, please read the JIRA 4.4 Upgrade Notes and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.4.4 Release Notes

13 December 2011

The Atlassian JIRA team announces the release of **JIRA 4.4.4**. This point release contains several updates and fixes, plus an important bug fix (JRA-26172). This bug lead to a database lock, which typically occurred when custom plugins create issues in a high-load environment (specifically inside the same thread where another issue is being updated).

JIRA 4.4.4 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.4 yet?

Take a look at all the new features in the JIRA 4.4 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.4.4 Upgrade Notes.

Updates and Fixes in this Release

JIRA 4.4.4 includes the following updates and bug fixes:

JIRA Issues (32 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-24139	JIRA's Profiling and Logging pages should allow you to add new logger configurations on the fly		Resolved
	JRA-26240	Need to upgrade AO to a version that fixes AO-217		Resolved
	JRA-25981	Fix compatibility changes introduced by backporting 5.0 mail handler code to 4.4		Resolved
	JRA-25794	On some flavours of Linux, when the automated JIRA Linux installer is executed with root-level privileges, the 'jira' user account is not created.		Resolved
	JRA-26362	Stalker bar does not work on FF8 windows (works on OSX)		Resolved

	JRA-26184	GH data is left in the database after a restore.		Resolved
	JRA-26183	Cannot use GreenHopper RapidBoards 5.7.x or 5.8.x on Oracle11g		Resolved
	JRA-25880	View Status page takes a long time to load if you have lots of statuses and lots of Workflows		Resolved
	JRA-25468	Configuring the 'jira.issue.actions.order' property on JIRA's Advanced Settings page (AdvancedApplicationProperties.jspa) has no effect.		Resolved
	JRA-25319	Fix source distribution plan so that a source distribution build is tested correctly.		Resolved
	JRA-26172	Creating an issue from a transition can lock the database under load.		Resolved
	JRA-26070	Slow down in the rendering of issue navigator when the logged in user is not in the first user directory		Resolved
	JRA-26061	Firefox 7 crashes when trying to create subtasks in Jira 4.4.3		Resolved
	JRA-26030	On the Project Summary page (in JIRA admin mode), change the 'Configure Application Links' link wording (which is misleading and incorrect) to 'Configure Project Links'.		Resolved
	JRA-25928	Buttons in the Projects, Issues in progress, Assigned to me gadgets do not work in Firefox 7.0.1		Resolved
	JRA-25907	Cannot change password on Internal Users with Global setting "External User Management = true"		Resolved
	JRA-25817	Update STP (support tools plugin) to 2.9.3		Resolved
	JRA-25797	Ampersand Does Not Render Appropriately in View Issue Screen		Resolved
	JRA-25588	Logging Confusion with New Implementaion of Mail Templates		Resolved
	JRA-25574	Assignee characters are messed up in email		Resolved
	JRA-25570	Broken link to the SimpleDateFormat JDK documention on the View and Edit Look and Feel admin pages		Resolved
	JRA-25558	IssueInputParameters API is not compatible with the timetracking changes made in JIRA 4.2		Resolved
	JRA-25480	XML RPC API method getIssuesFromTextSearchWithProject does not allow empty search terms		Resolved
	JRA-24161	osuser.xml migration fails if multiple ldap repositories defined		Resolved
	JRA-26228	Update AO to 0.17.2.5 to fix AO-204		Resolved
	JRA-26049	NPE when updating JIRA's logo or fav icon under a Turkish locale		Resolved
	JRA-26048	NPE when trying to evaluate the IsBrowser condition under a Turkish locale		Resolved
	JRA-25933	Upgrade bundled JIM to 3.5.2		Resolved
	JRA-25892	The XsrfVulnerabilityDetectionSQLInterceptor throws a RuntimeException when logging mutating actions that are not protected by @RequiresXsrfCheck annotations		Resolved
	JRA-25882	The changelog in notification emails is clumsy when accents are within the changes		Resolved
	JRA-25841	The logs are full of missing jira-styles web resource messages		Resolved

	JRA-25834	Using the documented procedure to move JIRA data from one database to another can fail if you have users where their name only differs in case, e.g 'admin' and 'Admin'		Resolved

JIRA 4.4.4 Upgrade Notes

Upgrading from JIRA 4.4.x to 4.4.4

Please follow the instructions in the general upgrading JIRA documentation.

Notes for plugin developers with custom mail handlers

JIRA 4.4.4 includes a dependency on `atlassian-mail` version `1.25` which is not 100% backward compatible with `atlassian-mail 1.24` included in previous releases of JIRA.

JIRA 4.4.4 also ships with `javamail 1.4.4`, which we cannot guarantee is 100% compatible with `javamail 1.4.1` (shipped with previous releases of JIRA).

Hence, there are two significant points to note as a result of these changes:

- The `com.atlassian.mail.server.MailServer.getSession()` method returns objects of the `javax.mail.Session` class rather than `alt.javax.mail.Session`
- The classes from `mockobjects-j1.3-j2ee1.3.jar` are no longer exported (i.e. made available) to Atlassian version 2 plugins. This `jar` is no longer distributed with JIRA.

Potential Impact and Solution

We believe that very few (if any) 3rd party plugins will be affected by the two changes mentioned above and that such plugins will be broken as a result of these changes.

- If your plugin uses the `com.atlassian.mail.server.MailServer.getSession()`method, doing either of the following should be sufficient to get it working with JIRA 4.4.4:
  - Recompile your plugin against 4.4.4
  - Use a higher level class like `MailQueueItem`.
- If your plugin depends on `mockobjects-j1.3-j2ee1.3.jar` (which is highly unlikely and could be considered questionable), we encourage you to either:
  - Remove this dependency or
  - If it is not possible to remove this dependency, bundle this `jar` with the plugin.

According to the changelog for `javamail` (http://www.oracle.com/technetwork/java/javaee/changes-220864.txt), we believe that these changes should not negatively impact current plugins. On the contrary, many bug fixes and compatibility improvements in this new version of `javamail` should make existing JIRA mail handlers more robust.

Rationale

JIRA 4.4.4 comes with substantial mail logging improvements and capabilities, which should be valuable to customers and support personnel troubleshooting problems related to incoming and outgoing mail.

Part of this feature work required the need to upgrade both the `atlassian-mail` and `javamail` libraries to the versions mentioned above. The anticipated small negative impact resulting from these upgraded libraries led us to believe that there was little or no business justification for supporting their older versions.

Upgrading from JIRA 4.3.x and earlier

In addition to the above, please read the JIRA 4.4 Upgrade Notes and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.4.3 Release Notes

JIRA 4.4.3 fixes a critical issue in JIRA 4.4.2 (JRA-25914 which may cause data corruption in a high-load environment).

If you upgraded to JIRA 4.4.2, please upgrade to JIRA 4.4.3 as soon as possible. JIRA 4.4.3 includes an upgrade task that will fix any data corruption resulting from JRA-25914. See the JIRA 4.4.3 Upgrade Notes for more information.

19 October 2011

The Atlassian JIRA team announces the release of **JIRA 4.4.3**. This point release contains new features that give users more JQL enhancements to filter issues based on system fields that possessed a specified value at some point in the past, in addition to several updates and fixes.

Features and improvements in JIRA 4.4.3:

- More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements
- Other Improvements

Thank you for your feedback:

More than **three** new features/improvements implemented

Over **420** votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

Download Latest Version

JIRA 4.4.3 is of course free to all customers with active JIRA software maintenance.

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.4.3 Upgrade Notes

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements

Enhancements were introduced in JIRA 4.4.0 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

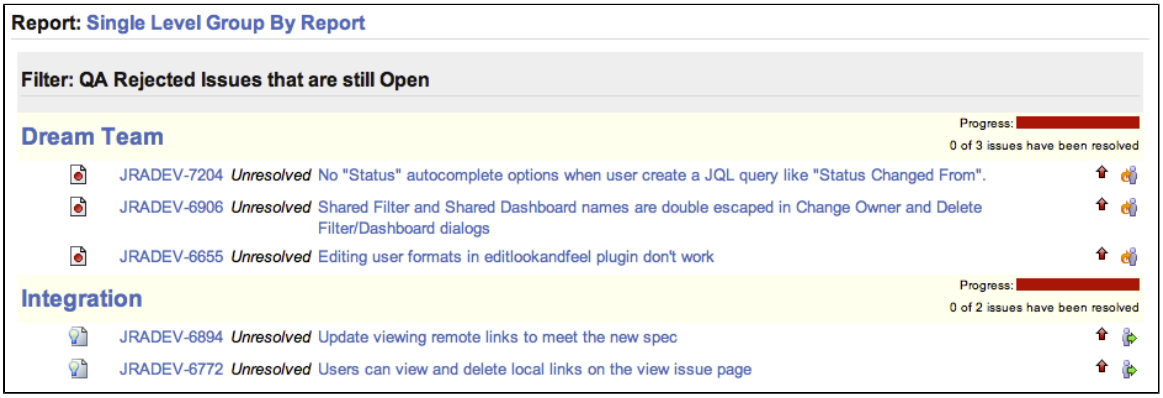
In JIRA 4.4.3, JQL supports the new "CHANGED" operator, which can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status changed FROM "In QA Review" to "QA Rejected" BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo', and after the start and before the end of the current week.

You can use complex queries such as these to generate the following 'Single Level Group By Report', which in the example below, shows grouping by 'Team'.



The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

The "WAS" operator can be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

Other Improvements

- Mixed-case LDAP usernames are now treated as the same user when conducting searches in JIRA (JRA-24558).

- When a new user is added to JIRA, JIRA will only add that user to groups which explicitly have the '**JIRA Users**' global permission. JIRA no longer adds new users to nested groups of a group that has the '**JIRA Users**' global permission, unless those nested groups also explicitly have this permission (JRA-25554).
- The Application Links Link Rendering plugin, which was broken when JIRA 4.4 was released, is now fixed and is compatible with all JIRA 4.4.x versions (JRA-25594).
- GreenHopper 5.7.4 is now bundled with JIRA 4.4.3.

Don't have JIRA 4.4 yet?

Take a look at all the new features in the JIRA 4.4 Release Notes and see what other features you are missing out on!

Updates and Fixes in this Release

JIRA 4.4.3 includes the following updates and bug fixes:

JIRA Issues (40 issues)		
Type	**Key**	**Summary**
	JRA-5536	Search the change history
	JRA-24558	JIRA using mixed-case username from LDAP corrupt other features
	JRA-25914	Two issues created with the same pkey value...
	JRA-25406	JIRA 4.4 Custom field multiuser picker will generate error in XML views
	JRA-25547	Issue statistics are now incorrect for multi-select fields
	JRA-24650	Expand "WAS" JQL operator to support fixVersion
	JRA-23861	An admin can lock themselves out of JIRA by changing the order of the User Directories.
	JRA-25688	Big database performance hit in JIRA 4.4.1
	JRA-25412	The "Event" is no longer displayed in the subject of text format emails.
	JRA-25395	Invalid Date Format Error At Issue Navigator After Change User Prefer Language
	JRA-23696	German translation for work ratio overflows in Searchview sidebar
	JRA-25819	The pie chart doesn't work correctly when using a multi select custom field
	JRA-25668	Progress Indicator when converting custom fields during 4.4 upgrade
	JRA-25594	Update Applinks-linker to work with JIRA 4.4
	JRA-25554	When creating a new user, new account is added to all nested group of jira-users
	JRA-25497	failed upgrade to new user management due to missing osuser.xml
	JRA-25288	Email received in different language if the sender has different language preference
	JRA-15819	record issue resolver
	JRA-25611	Unable to modify group memberships for re-created users
	JRA-25541	Infinite loop in group picker
	JRA-24918	'Test Settings' in User management needs perform more complete connection validation
	JRA-24425	issue settings translation not working
	JRA-27305	SQL syntax error importing data with PosgreSQL
	JRA-26268	Somehow track of fix version for issues
	JRA-26133	Project Import fails with error: "Existing issue found for key"
	JRA-25765	Bundle latest QE/QC plugin with JIRA 4.4.2
	JRA-25751	Add warning to the atlassian-feedback-plugin that feedback goes to Atlassian
	JRA-25750	Update AO plugin to 0.17 in 4.4.2
	JRA-25744	Attaching certain JPGs causes Attachments section of issue view to not render

	JRA-25692	Upgrade fisheye plugin to 3.4.5 to ship in JIRA 4.4.2
	JRA-25571	Uploading a plugin update through UPM does not run upgrade tasks
	JRA-25358	round trips between advanced and simple search with multiple label custom field search
	JRA-25297	Add net.java.ao.sql to log4j.properties in JIRA
	JRA-25237	JPG image causes JIRA to not display attachments in web UI and REST API
	JRA-25192	The list of roles in a project can be invalid after a project import
	JRA-24978	Remove the Sybil code in AbstractIssueSelectAction that causes multiple reads to the database when we have already done it
	JRA-24946	Link LDAP integration errors to a cac page helping customers to troubleshoot integration issues
	JRA-24945	Include Directory Configuration Summary from User Directories in System Info
	JRA-24634	Delegated LDAP UI does not include the "Follow Referrals" option
	JRA-17466	DefaultPortalPageManager should use index to find portal pages by owner and name

JIRA 4.4.3 Upgrade Notes

JIRA 4.4.3 fixes a critical issue in JIRA 4.4.2 (JRA-25914 which may cause data corruption in a high-load environment).

Upgrading from JIRA 4.4.x to 4.4.3

Please follow the instructions in the general upgrading JIRA documentation.

If you upgraded to JIRA 4.4.2, please upgrade to JIRA 4.4.3 as soon as possible. JIRA 4.4.3 includes an upgrade task that will fix any data corruption resulting from JRA-25914 (i.e. the creation of issues with duplicate keys in a high-load environment).

Upgrading from JIRA 4.3.x and earlier

In addition to the above, please read the JIRA 4.4 Upgrade Notes and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.4.2 Release Notes

JIRA 4.4.2 has a critical issue (JRA-25914) which may cause data corruption. You may wish to wait until JIRA 4.4.3 is released before upgrading JIRA.

12 October 2011

The Atlassian JIRA team announces the release of **JIRA 4.4.2**. This point release contains new features that give users more JQL enhancements to filter issues based on system fields that possessed a specified value at some point in the past, in addition to several updates and fixes.

Features and improvements in JIRA 4.4.2:

- More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements
- Other Improvements

Thank you for your feedback:

More than **three** new features/improvements implemented

Over **420** votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

Download Latest Version

JIRA 4.4.2 is of course free to all customers with active JIRA software maintenance.

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.4.2 Upgrade Notes

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements

Enhancements were introduced in JIRA 4.4.0 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

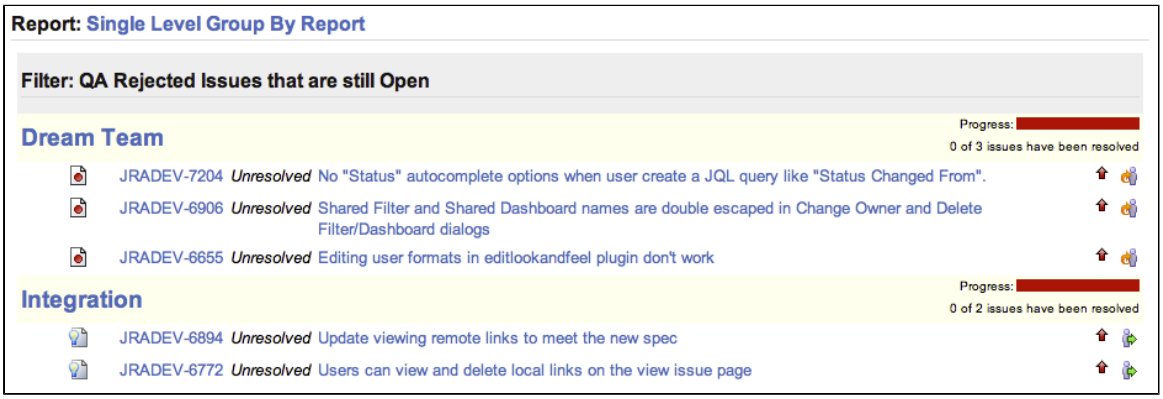
In JIRA 4.4.2, JQL supports the new "CHANGED" operator, which can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status changed FROM "In QA Review" to "QA Rejected" BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo', and after the start and before the end of the current week.

You can use complex queries such as these to generate the following 'Single Level Group By Report', which in the example below, shows grouping by 'Team'.



The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

The "WAS" operator can be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

Other Improvements

- Mixed-case LDAP usernames are now treated as the same user when conducting searches in JIRA (JRA-24558).
- When a new user is added to JIRA, JIRA will only add that user to groups which explicitly have the '**JIRA Users**' global permission. JIRA no longer adds new users to nested groups of a group that has the '**JIRA Users**' global permission, unless those nested groups also explicitly have this permission (JRA-25554).
- The Application Links Link Rendering plugin, which was broken when JIRA 4.4 was released, is now fixed and is compatible with all JIRA 4.4.x versions (JRA-25594).
- GreenHopper 5.7.4 is now bundled with JIRA 4.4.2.

Don't have JIRA 4.4 yet?

Take a look at all the new features in the JIRA 4.4 Release Notes and see what other features you are missing out on!

Updates and Fixes in this Release

JIRA 4.4.2 includes the following updates and bug fixes:

JIRA Issues (34 issues)

Type	Key	Summary
	JRA-5536	Search the change history
	JRA-24558	JIRA using mixed-case username from LDAP corrupt other features
	JRA-25406	JIRA 4.4 Custom field multiuser picker will generate error in XML views
	JRA-25547	Issue statistics are now incorrect for multi-select fields
	JRA-24650	Expand "WAS" JQL operator to support fixVersion
	JRA-23861	An admin can lock themselves out of JIRA by changing the order of the User Directories.
	JRA-25688	Big database performance hit in JIRA 4.4.1
	JRA-25412	The "Event" is no longer displayed in the subject of text format emails.
	JRA-25395	Invalid Date Format Error At Issue Navigator After Change User Prefer Language
	JRA-23696	German translation for work ratio overflows in Searchview sidebar
	JRA-25668	Progress Indicator when converting custom fields during 4.4 upgrade
	JRA-25594	Update Applinks-linker to work with JIRA 4.4
	JRA-25554	When creating a new user, new account is added to all nested group of jira-users
	JRA-25497	failed upgrade to new user management due to missing osuser.xml
	JRA-25288	Email received in different language if the sender has different language preference
	JRA-15819	record issue resolver
	JRA-25611	Unable to modify group memberships for re-created users
	JRA-25541	Infinite loop in group picker
	JRA-24918	'Test Settings' in User management needs perform more complete connection validation
	JRA-24425	issue settings translation not working
	JRA-25765	Bundle latest QE/QC plugin with JIRA 4.4.2
	JRA-25751	Add warning to the atlassian-feedback-plugin that feedback goes to Atlassian
	JRA-25750	Update AO plugin to 0.17 in 4.4.2
	JRA-25692	Upgrade fisheye plugin to 3.4.5 to ship in JIRA 4.4.2
	JRA-25571	Uploading a plugin update through UPM does not run upgrade tasks
	JRA-25358	round trips between advanced and simple search with multiple label custom field search
	JRA-25297	Add net.java.ao.sql to log4j.properties in JIRA
	JRA-25237	JPG image causes JIRA to not display attachments in web UI and REST API
	JRA-25192	The list of roles in a project can be invalid after a project import
	JRA-24978	Remove the Sybil code in AbstractIssueSelectAction that causes multiple reads to the database when we have already done it
	JRA-24946	Link LDAP integration errors to a cac page helping customers to troubleshoot integration issues
	JRA-24945	Include Directory Configuration Summary from User Directories in System Info
	JRA-24634	Delegated LDAP UI does not include the "Follow Referrals" option
	JRA-17466	DefaultPortalPageManager should use index to find portal pages by owner and name

JIRA 4.4.2 Upgrade Notes

JIRA 4.4.2 has a critical issue (JRA-25914) which may cause data corruption. You may wish to wait until JIRA 4.4.3 is released before upgrading JIRA.

Upgrading from JIRA 4.4.x to 4.4.2

Please follow the instructions in the general upgrading JIRA documentation.

Upgrading from JIRA 4.3.x and earlier

In addition to the above, please read the JIRA 4.4 Upgrade Notes and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.4.1 Release Notes

12 September 2011

The Atlassian JIRA team announces the release of **JIRA 4.4.1**. This point release contains new features that give administrators the ability to manage other users' shared filters and shared dashboards, in addition to several updates and fixes.

Features and improvements in JIRA 4.4.1:

- Manage Other Users' Shared Filters and Dashboards
- Other Improvements

Thank you for your feedback:

More than **four** new feature requests implemented

Over **672** votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

Download Latest Version

JIRA 4.4.1 is of course free to all customers with active JIRA software maintenance.

Upgrading from a Previous Version of JIRA

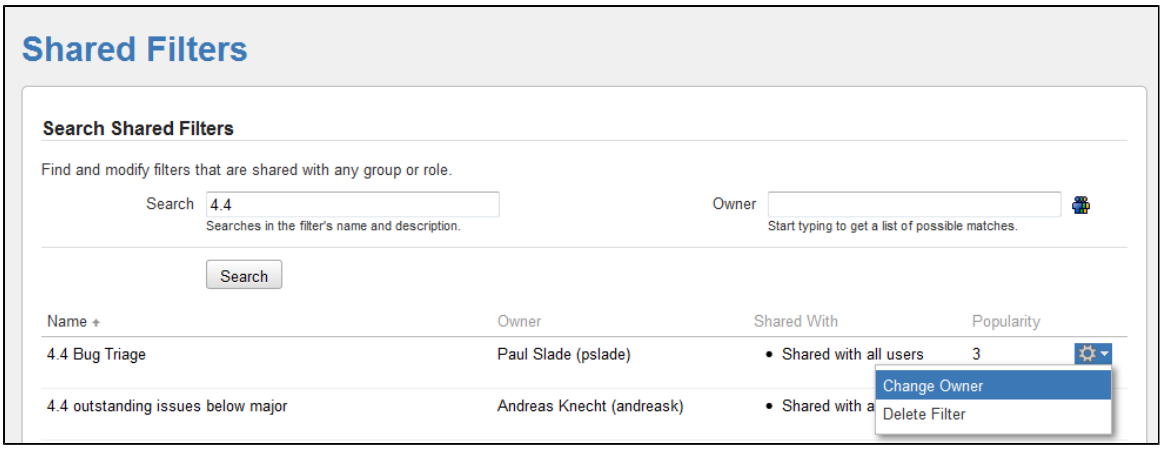
If you are upgrading, please read the JIRA 4.4.1 Upgrade Notes.

Manage Other Users' Shared Filters and Dashboards

JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboards. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.



Other Improvements

- The 'Advanced Settings' page of JIRA's Administration area now has improved validation (JRA-24974).
- GreenHopper 5.7.1 now bundled with JIRA 4.4.1.
- When configuring JIRA to connect to a MySQL database via the JIRA setup wizard or JIRA Configuration Tool, the `sessionVariables=storage_engine=InnoDB` property is now specified by default in the database URL and no longer needs to be added manually (JRA-24857).
- Improvements to JIRA's issue importers.

Don't have JIRA 4.4 yet?

Take a look at all the new features in the JIRA 4.4 Release Notes and see what other features you are missing out on!

Updates and Fixes in this Release

JIRA 4.4.1 includes the following updates and bug fixes:

JIRA Issues (58 issues)

Type	Key	Summary	Priority	Status
	JRA-25849	Change Dashboard ownership		Resolved
	JRA-25315	Remove dependency on mindprod CSV parser		Closed
	JRA-24974	Validation and UI improvements of the Advanced configuration screen.		Resolved
	JRA-24823	Dialog after clicking on "Create your first project" link is confusing		Resolved
	JRA-19780	Change Dashboards and Filters ownership		Resolved
	JRA-9997	Allow administrators to manage filters owned by other users		Resolved
	JRA-8303	Ability for global ownerity of filters (or at least changing the filter owner)		Resolved
	JRA-6561	Administrators should be able to delete all filters		Resolved
	JRA-25539	User can't tab out of the "Linked Issues" dropdown field to next item in "Create Issue" form in IE9.		Resolved
	JRA-25309	Viewing XML view of some issues or filters results in net.java.ao.sql.ActiveObjectSqlException		Resolved
	JRA-25374	Cannot update version custom field via SOAP on 4.4		Resolved
	JRA-25362	ActiveObjects cannot encode value for unsupported column type "ntext" on MS SQL Server 2008		Resolved
	JRA-25300	The issue page scrolls when user scrolls through listing in frother control.		Resolved

	JRA-25298	SOAP search service should have hard limit on results		Resolved
	JRA-25034	Multi-select custom field value handling incompatible with previous releases		Resolved
	JRA-24996	You should not be able to disable the "Atlassian JIRA - Plugins - Project Config Plugin"		Resolved
	JRA-24990	webwork's ValueStack caches class objects, preventing the GC of a plugin's classloader		Resolved
	JRA-24952	When Syncing with Crowd fails due to a bad user it fails for everyone after that user, this should be more robust.		Resolved
	JRA-24893	Webwork Configuration mechanism induces unnecessary SQL statements and hence hurts performance		Resolved
	JRA-23051	Auto-complete in Create Issues is painfully slow when you have many components		Resolved
	JRA-25348	Upgrade task for Select Custom fields in upgrade to JIRA 4.4 wipes out customer data.		Resolved
	JRA-25316	Non canonical time zone id causes IllegalArgumentExceptions		Resolved
	JRA-25299	Usability: Frother control doesn't scroll to end of the list to show the message.		Resolved
	JRA-25134	The edit pencil next to scheme names in new edit pages is ambiguous		Resolved
	JRA-25066	Decorator atl.userprofile is displaying incorrectly		Resolved
	JRA-25036	error in edit-multicheckboxes.vm		Resolved
	JRA-25009	Loading the Gadget Directory can block indefinitely waiting for the news gadget to download		Resolved
	JRA-24944	Extend transaction support to BulkMove and MoveIssue		Resolved
	JRA-24942	Administration Search does not work for freshly installed plugin - JIRA restart required		Resolved
	JRA-24784	The .dot dialog drop down is not positioned properly if user is lot of columns configured in issue navigator.		Resolved
	JRA-24554	Upgrade task doesn't like Crowd servers with "services" in the hostname		Resolved
	JRA-23520	Confusing Log Work screen freezes on navigate away		Resolved
	JRA-19963	Editing a gadget while in canvas mode causes the gadget iframe to be clipped		Resolved
	JRA-19893	Roadmap gadget is blank when it tries to render a version that has no description (on Oracle)		Resolved
	JRA-19522	No horizontal scrollbar when editing filter		Resolved
	JRA-10088	Calendar in date picker gets week numbers wrong		Resolved
	JRA-25389	Upgrade bundled greenhopper to 5.7.1		Resolved
	JRA-25270	JIRA 4.4 does not work with schema name 'jira' in SQL Server		Resolved
	JRA-25239	MultiSelect "too many options" suggestion incorrect HTML		Resolved

	JRA-25047	Deleting a gadget while in canvas mode causes craziness		Resolved
	JRA-25042	Please fix typo on JIRA 4.4 setup wizard - administrator account		Resolved
	JRA-24857	Create MySQL tables using InnoDB engine by default - session variable		Resolved
	JRA-24435	Get a "dirty form warning" from sub-task quick create even when the user chooses cancel		Resolved
	JRA-24014	Buttons are cramped in IE8		Closed
	JRA-23775	The Version/Componenents selector info message does not render well on Move/Bulk-Edit Screens		Closed
	JRA-23410	Icons display over the top of Time Tracking border		Resolved
	JRA-23106	Using long project keys causes display issues in Firefox when linking/relating issues		Closed
	JRA-22913	Time tracking progress bars stretch out while a dialog is open in IE7		Closed
	JRA-22432	Funny icons appear on Project/Version page when Version description is very long		Closed
	JRA-21992	Recently Created gadget visually truncated tabular data after 'Back' from Issue Navigator		Closed
	JRA-21658	Subtask order icons not showing up in IE 7.x		Resolved
	JRA-21390	Multi User Picker remains with a z-index of 9000 after autocomplete.		Closed
	JRA-21240	Font size is super big under printable area on a ticket in IE7		Closed
	JRA-20839	Horizontal scrollbar appears on Reset Password screen		Closed
	JRA-20806	Bulk Operations need a visual refresh		Closed
	JRA-20358	JiraKeyUtils.isPartOfUrl doesn't recognize line breaks as whitespace		Resolved
	JRA-20226	In 1024x768 resolution, cannot scroll horizontally in Fisheye report		Closed
	JRA-20032	HTML formatted project description in project summary view is not correctly displayed		Closed

JIRA 4.4.1 Upgrade Notes

Upgrading from JIRA 4.4 to 4.4.1

Please follow the instructions in the general upgrading JIRA documentation.

Changes to the CSV Importer

Customers using the CSV importer feature and JIRA's Jelly Tags should be aware that the `CSVReader` class is now based on `au.com.bytecode.opencsv.CSVReader` rather than `com.mindprod.csv.CSVReader`. As a result of this change, the backslash character ('\') is now handled differently. If the backslash is found inside quoted text it will be ignored (removed) unless it precedes another backslash or a double quote. In case it does the following character will be kept but the first backslash will be omitted. See JRA-25315 and JIM-596 for details.

Fixes to SOAP and XML/RPC compatability

In JIRA 4.4 there was an accidental break compatibility of the SOAP and XML/RPC apis. Basically "Select" and "Multi-Select" custom fields were sending and expecting option ids instead of data values. This has been corrected and the SOAP and XML/APIs should now behave as they did for JIRA 4.3 and earlier. This is fully documented in issue JRA-25374. If you made changes to handle this compatibility breakage, you may need to revert your changes.

Changes to the default value of the `jira.search.views.max.limit` property

The `jira.search.views.max.limit` property sets a 'hard' limit on the number of issues returned. See Limiting the number of issues returned from a search view such as an RSS feed for details.

In JIRA 4.4.0, this value was disabled by default, by being either absent from the `jira-config.properties` file or present in this file but disabled with a preceding '#'. From JIRA 4.4.1, however, this property has a default value of 1000. See JRA-25298 for details about why this changed.

Upgrading from JIRA 4.3.x and earlier

In addition to the above, please read the JIRA 4.4 Upgrade Notes and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.3 Release Notes

16 March, 2011

The Atlassian JIRA team is proud to bring you a brand new version of one of the world's favourite issue-trackers.

Identity management comes of age in JIRA 4.3, with complete LDAP integration. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins, and a new data importer for easier migration from your old systems. Additionally, a raft of new JQL functions give you many powerful new searching options, such as the ability to search an issue's change history.

Dashboard Publish/Subscribe with Confluence (i.e. Gadgets 2.0) is also included, so if you are using Confluence as well as JIRA, your Confluence gadgets will now appear in the JIRA Gadget Directory.

Note to developers: JIRA 4.3 includes Unified Application Links (UAL) 3.2, Gadgets 2.0, and the Atlassian Plugin Framework version 2.6.

Upgrading to JIRA 4.3 is free for all customers with active JIRA software maintenance as of March 15, 2011.

Highlights of JIRA 4.3:

- Full Integration with LDAP and Active Directory
- Easy Management of User Directories via JIRA Admin UI
- New Plugin Management System
- Add Another Application's Gadgets to your JIRA Gadget Directory
- Improved Importer
- Dashboard Performance Improvements
- JIRA Now Supports 'In-place Database Upgrades'
- Search for Issue Changes, Relative Dates and Relative Versions with JQL
- Quick Search Enhancements
- Revamped User Avatars
- Improvements to Issue Links
- Remembered Assignees
- Mail Server Configuration Improvements
- Security Enhancements
- Support Tools Plugin Now Bundled
- Application Links: Connecting Applications Together
- Support for Chrome and Safari 5 Browsers
- REST API Improvements
- Other Enhancements and Fixes

Thank you for your feedback:

More than **20** new feature requests implemented

Over **1000** votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

Upgrading to JIRA 4.3

JIRA 4.3 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 4.3 Upgrade Guide.

Highlights of JIRA 4.3

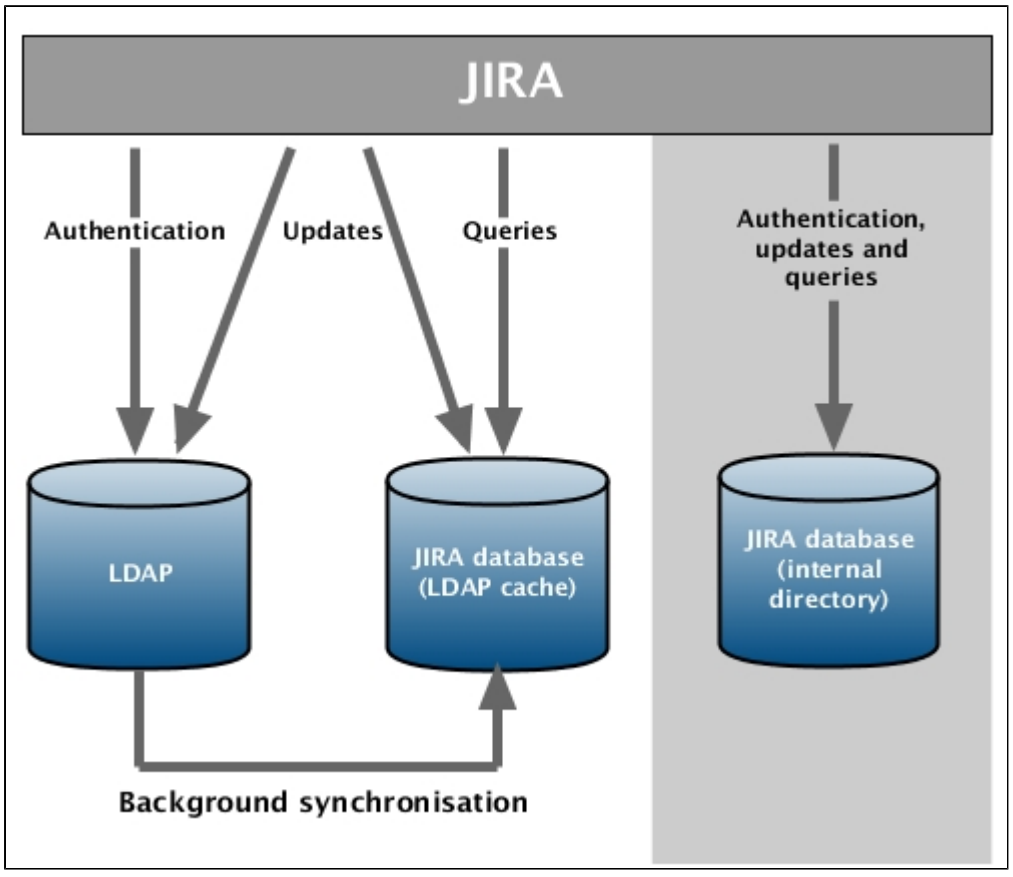
1

Full Integration with LDAP and Active Directory

The way users and groups are stored and accessed in JIRA has been totally rewritten in this release, giving you the ability to connect to an LDAP server — including Microsoft Active Directory — for *all* user information. Your options include:

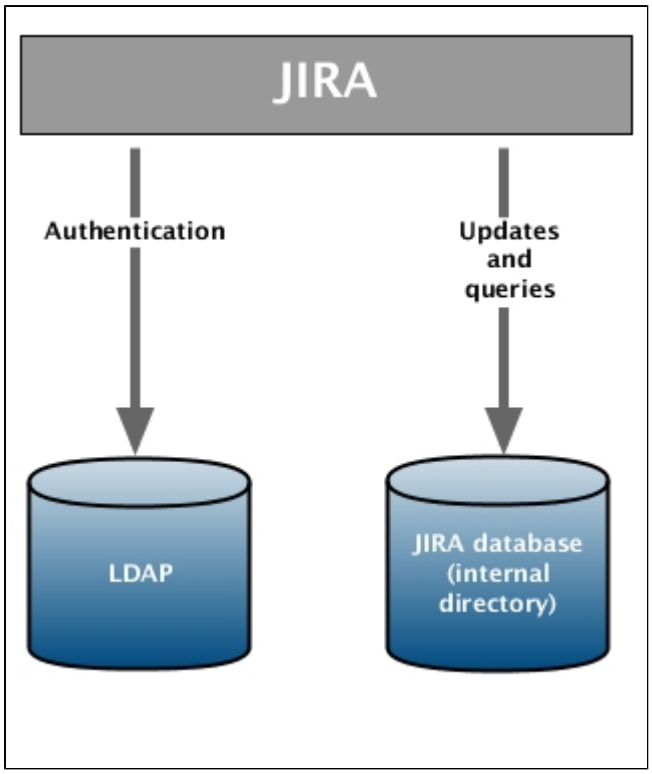
Integrate JIRA with LDAP/Active Directory

Update your user details in either JIRA *or* LDAP/Active Directory and they will automatically populate to the other repository:



Use LDAP for authentication only

This was the old way of integrating JIRA with LDAP, prior to JIRA 4.3. You can still do this, but much more easily (see below).



Connect to a Crowd server for user management

For larger and more complex installations, you may need to install Atlassian Crowd for user management and single sign-on. (See our guide to limitations and recommendations.) When integrating earlier versions of JIRA and Crowd, you had to manually edit a number of configuration files. JIRA 4.3 offers the following new features:

- Simple and quick setup via the JIRA and Crowd administration consoles.
- Clever synchronisation and caching to ensure the best response times.
- More...

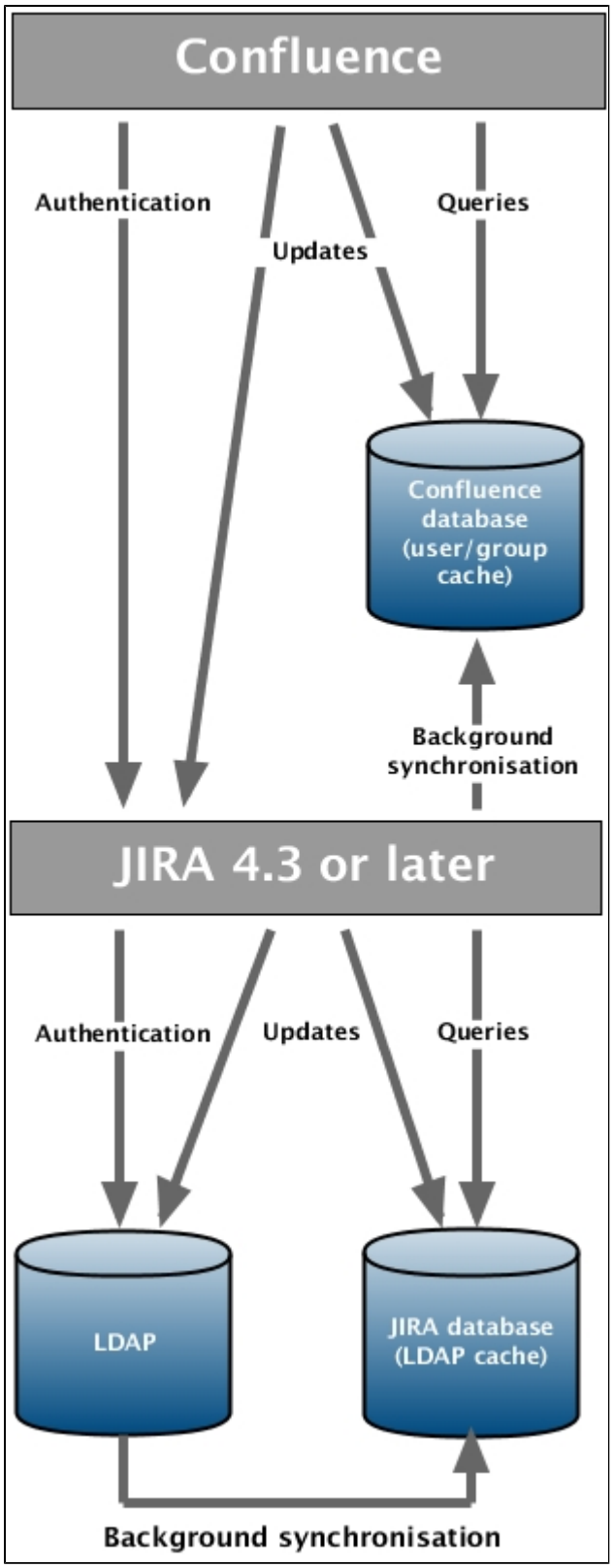
Use multiple LDAP and/or Crowd servers simultaneously

If you have multiple directories, you can now simply connect JIRA to all of them.

Connect Confluence to your JIRA User Directory

JIRA 4.3 can act as the directory manager for your Confluence site, interacting with one or more user directories and ensuring that you have the same set of users and groups across both applications.

The Confluence 3.5 administration UI can connect directly to JIRA 4.3. Clever synchronisation and caching ensure the best response times for your directory searches.



Earlier versions of Confluence can continue to use a direct JDBC connection to JIRA 4.2 and earlier.

^Top

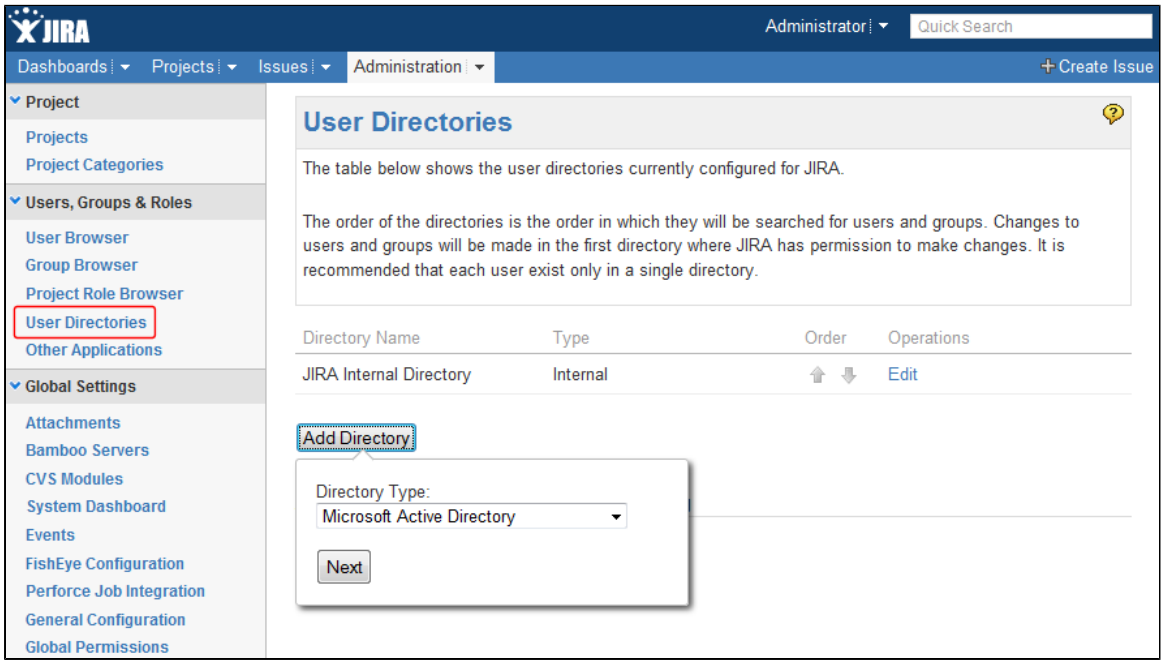
2

Easy Management of User Directories via JIRA Admin UI

In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.

JIRA 4.3 brings a simple, powerful and flexible directory management interface:

- Choose from a list of supported directory types, including Microsoft Active Directory and ten other popular LDAP schemas.
- Configure all your LDAP settings via the JIRA Administration interface: permissions, server and schema settings. We pre-populate the fields with default values depending on your choice of directory type.
- Choose the LDAP permissions to suit your needs: read/write, read only, local groups or authentication only.
- Make use of the caching and copy-as-required configurations to optimise the performance of your LDAP searches.
- Add as many directory servers as you need. Connect directory servers of different types, including the JIRA internal directory (default), LDAP, Crowd and/or another JIRA server.
- Turn on support for nested groups.
- Configure your LDAP connection pool.

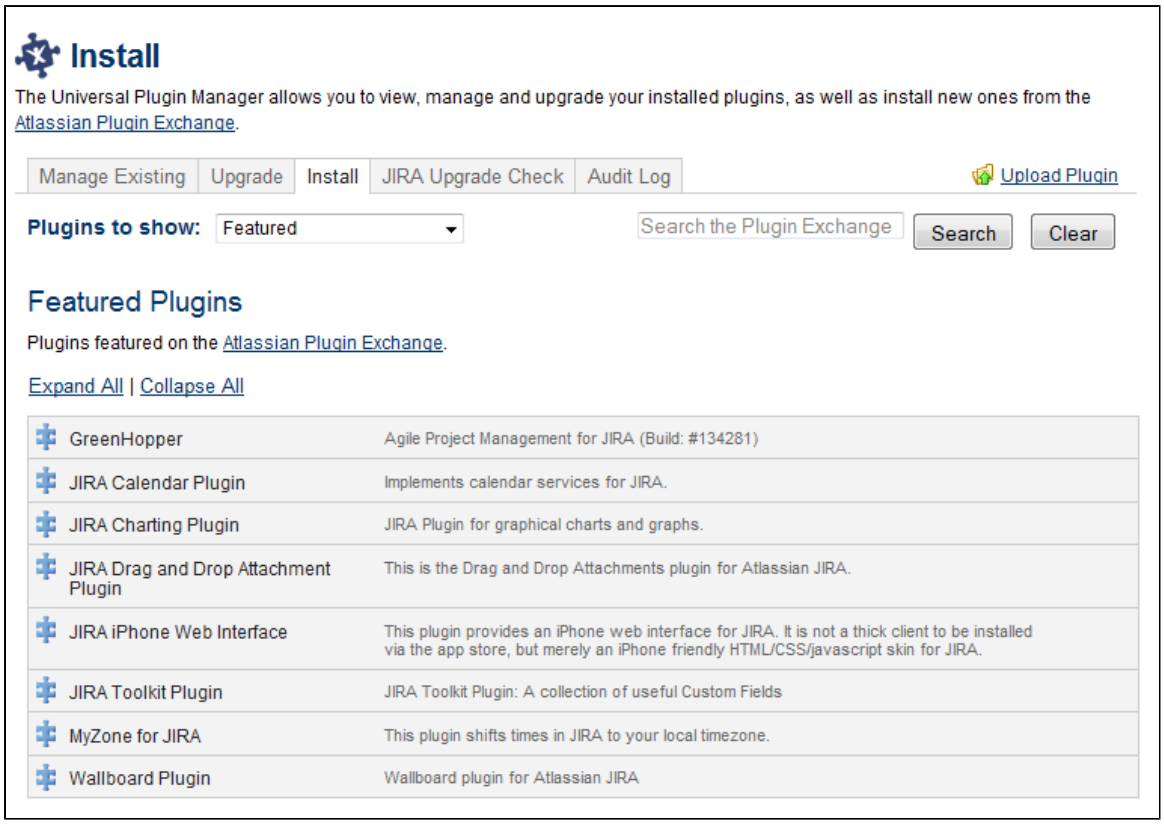


^Top

New Plugin Management System

JIRA 4.3 includes the Universal Plugin Manager (UPM), which provides you with a simple way of adding and managing plugins:

- Auto-discovery of available plugins (see **Featured Plugins** in the screenshot below)
- Point-and-click installation — you no longer need to download JAR files and shut down JIRA to install them.



See Managing JIRA's Plugins.

^Top

Add Another Application's Gadgets to your JIRA Gadget Directory

JIRA 4.3 includes Atlassian Gadgets 2.0 with Gadget Subscriptions. This allows you to quickly add all gadgets from your Confluence, Bamboo, FishEye or Crucible instance — or from another JIRA instance — to your JIRA Gadget Directory, for easy addition to your JIRA dashboard:

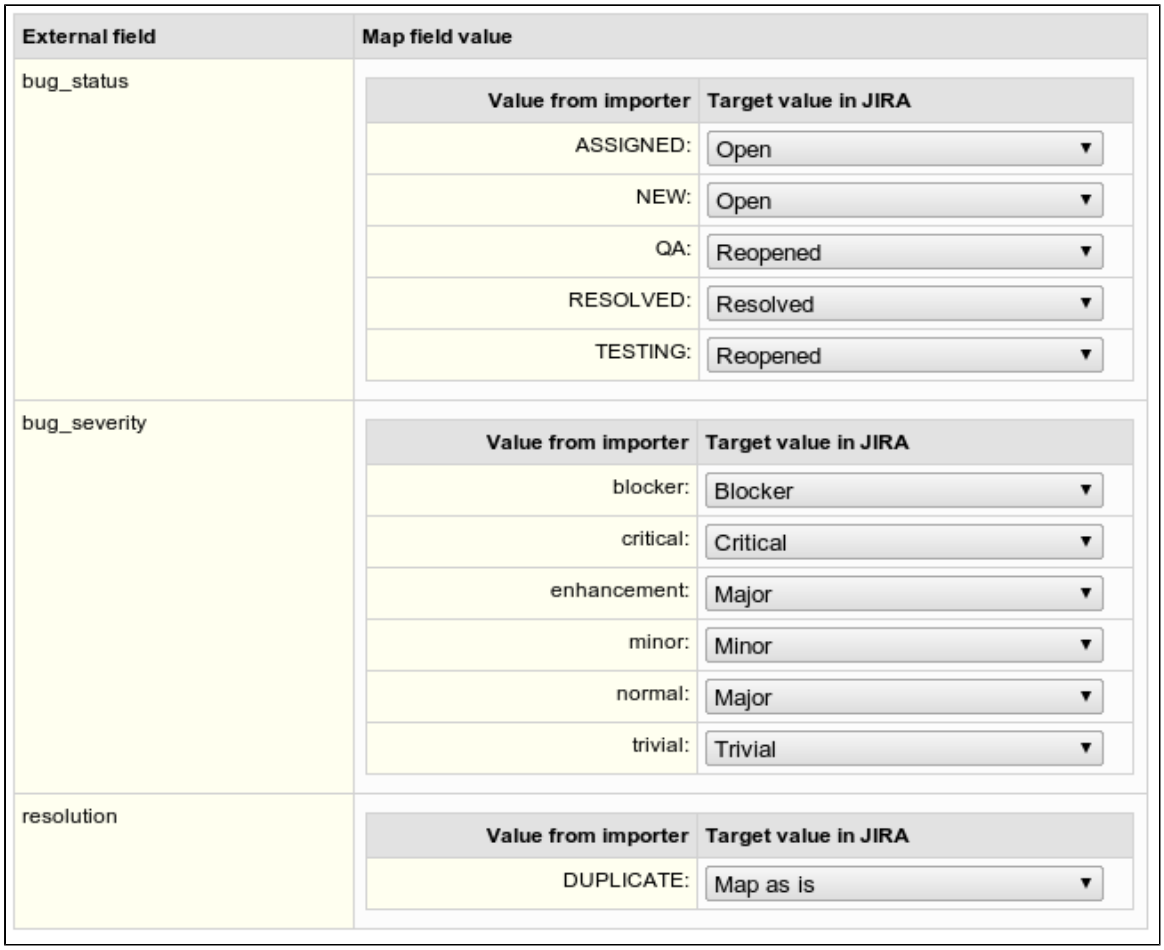
See Subscribing to Another Application's Gadgets.

^Top

Improved Importer

JIRA 4.3 provides a greatly improved Importer for Bugzilla, Mantis, FogBugz and CSV files, by bundling the JIRA Importers Plugin.

The web-based import wizard makes it easy to map fields — and individual field values — from your old bug-tracker to your new JIRA system:



^Top

Dashboard Performance Improvements

We are extremely pleased to announce that you should see a noticeable reduction in the time it takes to load a Dashboard.

- On average, dashboard performance has improved by 29%
- For large dashboards, the performance improvement is 35%

We are also extremely pleased to announce that you should see a noticeable reduction in the time it takes to load the Gadget Directory (depending on how many external gadgets you have configured).

^Top

JIRA Now Supports 'In-place Database Upgrades'

JIRA 4.3 now officially supports 'in-place database upgrades', when upgrading from JIRA 4.0.0 or later.

This method requires much less downtime during the JIRA upgrade process, especially if you operate a large JIRA installation. You no longer need to export your existing JIRA data to an XML backup and then restore this data into your new JIRA version. Instead, we now support simply 'pointing' your new version of JIRA at your existing JIRA database!

See the documentation for more details.

^Top

Search for Issue Changes, Relative Dates and Relative Versions with JQL

JIRA 4.3 incorporates new several functions and operators. You can now:

Search the Change History with JQL 'WAS' Operator

JIRA 4.3 introduces the ability to search the Change History of issues. In this release you can search for changes to the Status field, and there's much more functionality to come in future releases.

For example, the following will return all issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

The `WAS` operator can be used with the `NOT`, `IN` and `NOT IN` operators, e.g. to find issues that have never had a status of 'In Progress' or 'Resolved':

```
status WAS NOT IN ("In Progress","Resolved")
```

Search for relative dates and versions with JQL 'startOfDay', 'endOfDay', 'earliestUnreleasedVersion', 'latestReleasedVersion'

JIRA 4.3 introduces the ability to search for issues relative to the current day, month, week or year. For example, to find issues that have been created today:

```
created > startOfDay()
```

Or to find issues that are due by the end of this month:

```
due < endOfMonth()
```

You can also perform searches based on the earliest unreleased version (i.e. the next version that is due to be released) of a specified project:

```
earliestUnreleasedVersion(project)
```

Or on the most recently released version of a specified project:

```
latestReleasedVersion(project)
```

^Top

Quick Search Enhancements

JIRA 4.3 incorporates several enhancements to Quick Search. You can now:

Use a wild card when searching the 'Fix Version' field with Quick Search

When using Quick Search, you can now use the wildcard symbol: "`*`" to find issues that matches a core part of a Fix Version. For example, "`ff:3.2*`" will match any issue whose Fix For Version is:

- 3.2
- 3.2-beta
- 3.2.1
- 3.2.x

Use the `r:` prefix with Quick Search to find issues reported by a specific user

With Quick Search, you can find issues reported by you, another user or with no reporter, using the prefix "`r:`" followed by a specific reporter term such as "`me`", a username or "`none`", such that:

- "`r:me`" — finds issues reported by you.
- "`r:samuel`" — finds issues reported by the user whose username is "samuel".
- "`r:none`" — finds issues with no reporter.

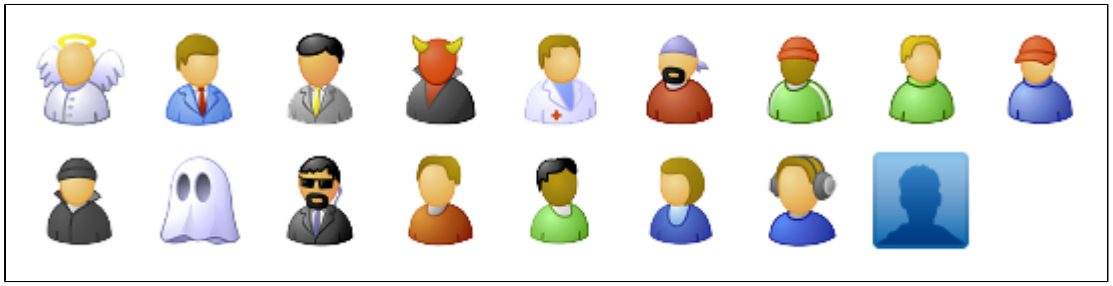
^Top

Revamped User Avatars

JIRA 4.3 introduces the new-look Atlassian avatars:

- User avatars are displayed as the icon for your profile, and to illustrate your comments on an issue. See the documentation on Adding a User Avatar.

We were fond of the old avatars, but think you'll agree they were looking a little dated by comparison:

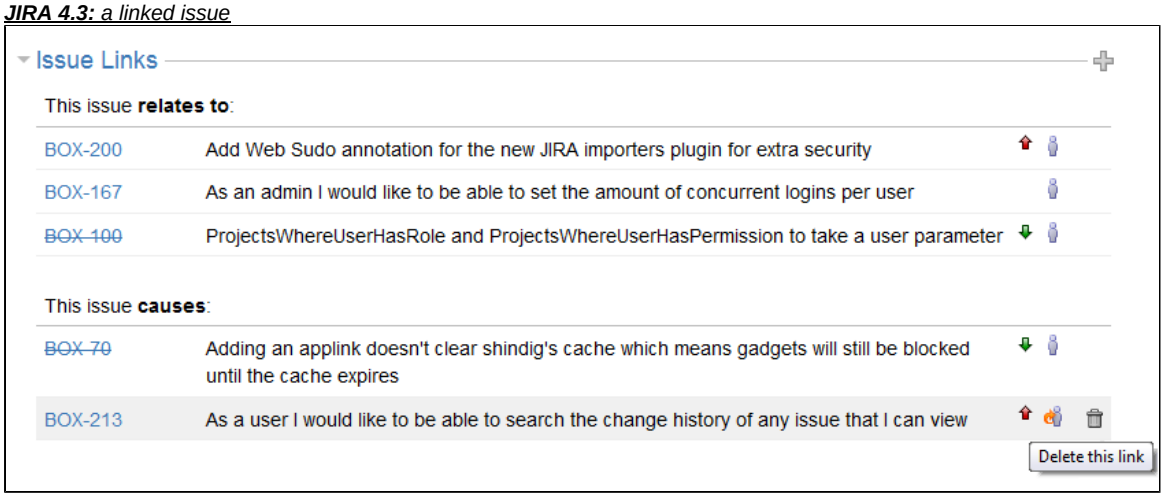


^Top

Improvements to Issue Links

The 'Linked Issues' section of the 'View Issue' screen has been streamlined to make it both more compact and quicker to use. You can now delete links directly from this screen upon mouse hover:

JIRA 4.3: *a linked issue*



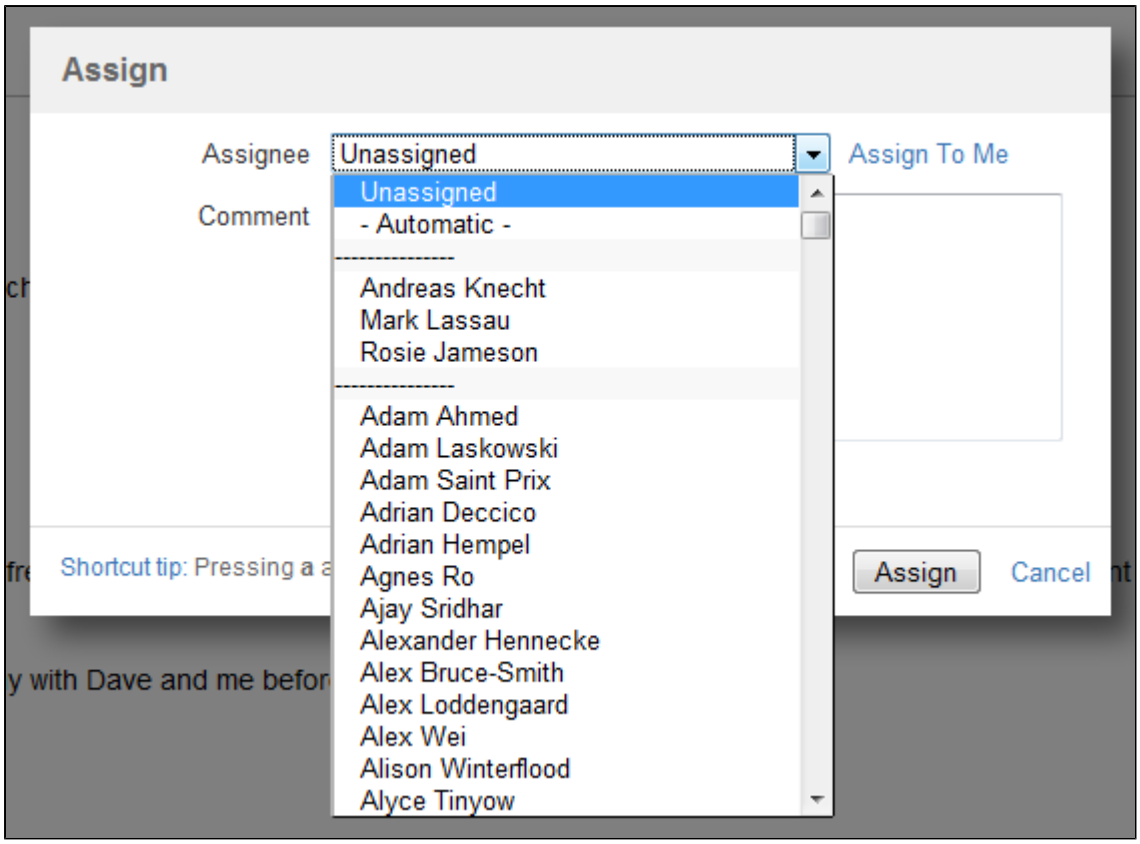
Previously, to delete a link you needed to click the 'cog' icon (which took you to a separate screen):

JIRA 4.2: *a linked issue*

^Top

Remembered Assignees

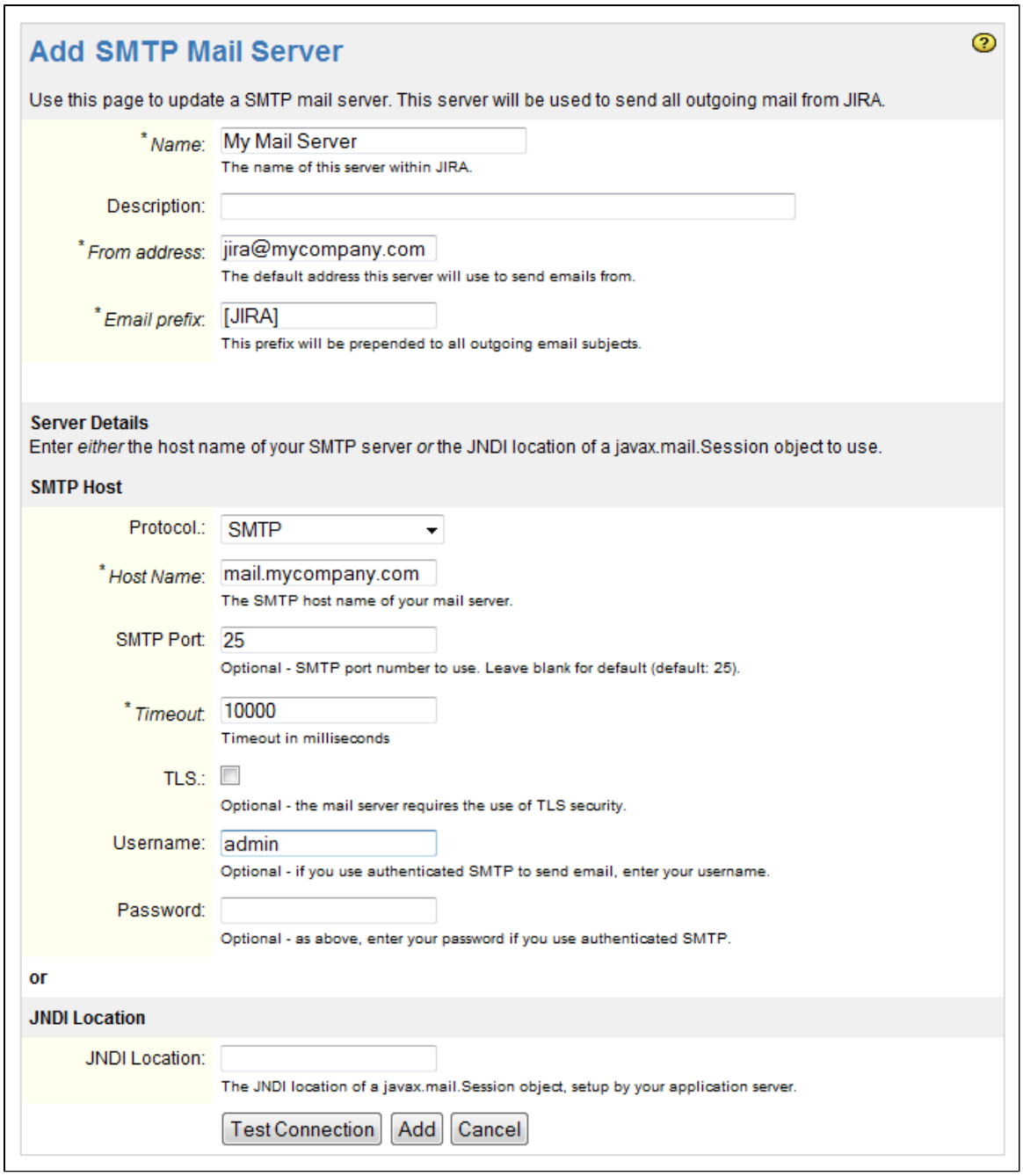
JIRA now makes it easier to assign users to issues. The top of the assignee dropdown list shows the last five people you assigned issues to, as well as the reporter of the issue and all participants involved in the issue.



^Top

Mail Server Configuration Improvements

In JIRA 4.3, administrators can now test their mail server configuration with the "**Test Connection**" button.



A "**Timeout**" field has also been added.

^Top

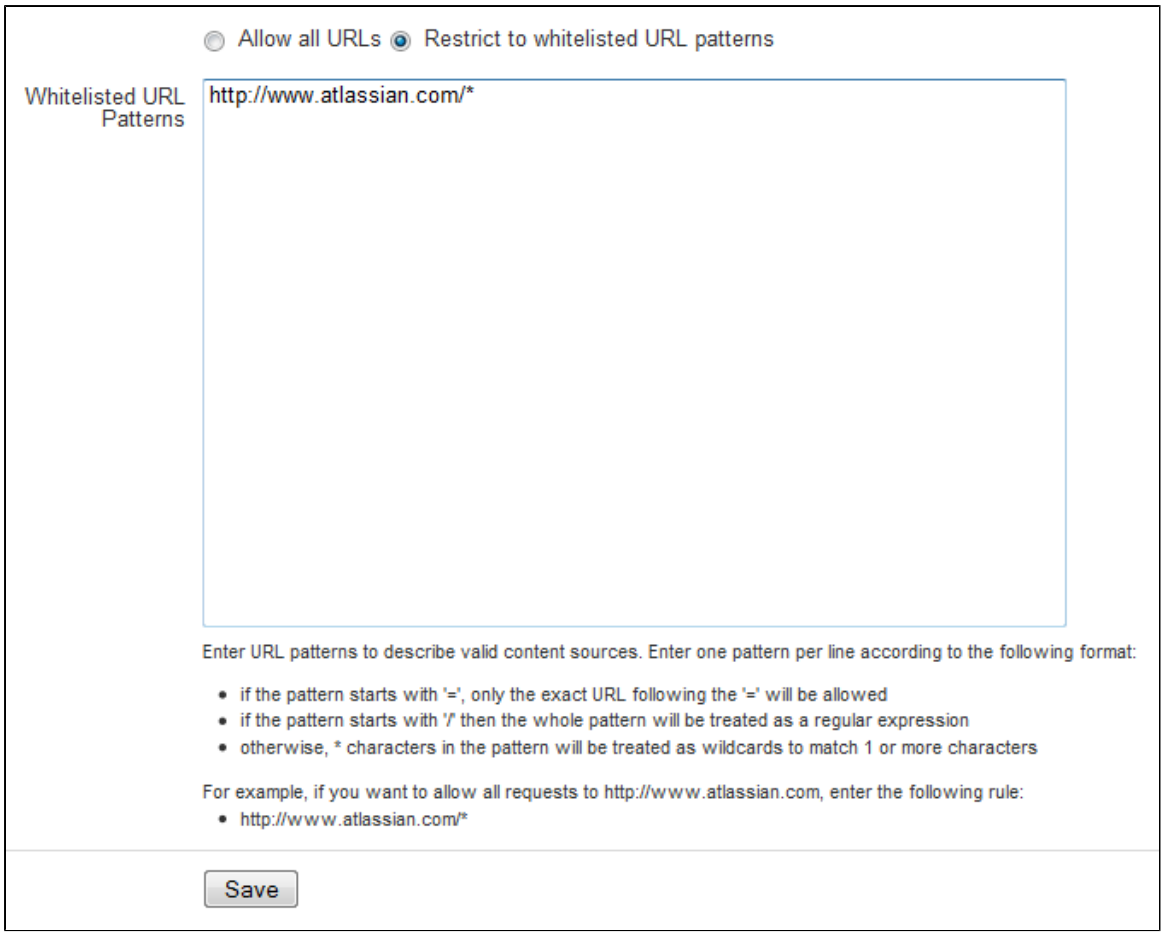
Security Enhancements

For enhanced security:

'Whitelist' for External URLs

For security reasons, an administrator may wish to limit the URLs from which users can source content that is displayed on their JIRA site (e.g. via an External Gadget).

JIRA 4.3 allows you to create a 'Whitelist' of URLs — or URL patterns — whose content you wish to make available to users of your JIRA site:



See the documentation for more details.

User Email Change is now Password Protected

Users will now be prompted to enter their password when changing their email address.

JIRA provides Secure Administration Sessions

Access to JIRA's administration features and functions is now password-protected.

See the documentation for more details.

^Top

Support Tools Plugin Now Bundled

We are very pleased to announce that the Atlassian Support Tools Plugin is now bundled with JIRA. The Atlassian Support Tools Plugin provides tools to help you troubleshoot issues with Atlassian products and get help from Atlassian Support.

See the documentation for more details.

^Top

Application Links: Connecting Applications Together

JIRA 4.3 includes Application Links, which makes it easy to connect your Atlassian applications together.

Application Links allows you to link your JIRA, Confluence, FishEye, Crucible, Bamboo and Subversion applications.

Linking two applications allows you to share information and access one application's functions from within the other. For example, if you linked a Confluence server to a JIRA server, you can create, find and insert JIRA issues directly onto a Confluence page or blog post using the new 'Insert JIRA Issue' option in Confluence 3.5.

You can even choose to associate your JIRA project with the 'entities' of other Atlassian applications (i.e. Confluence spaces, FishEye repositories or projects, Crucible projects, Bamboo projects, or other JIRA projects). The 'Configure Application Links' option on your JIRA project configuration screen, allows you to easily associate your JIRA project with these entities.

Applications Links is bundled with JIRA 4.3, Confluence 3.5 and FishEye 2.4.

See the documentation for more details.

^Top

Support for Chrome and Safari 5 Browsers

We are very pleased to announce that Chrome and Safari 5 are supported for use with JIRA 4.3.

^Top

REST API Improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance. Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3, click here.

^Top

JIRA 4.3 Upgrade Guide

On this page:

- Upgrading from JIRA 4.2 to 4.3
  - General upgrade instructions
    - 'In-place database upgrade' is now the recommended method
  - Changes in `jira-application.properties`
  - Changes in `seraph-config.xml` and Crowd Integration
  - Gadgets can only access External URLs that are on the Whitelist
  - Changes to user management in JIRA
    - Upgrade considerations
      - For customers with internally managed users
      - For customers using LDAP for authentication
      - For customers connecting to Crowd for user management
      - For customers with a pre-Confluence 3.5 installation that uses JIRA for user management (IMPORTANT!)
      - For customers who have written a custom provider for user management

        - For customers with non-standard configurations
    - Other considerations
        - Duplicate groups MUST BE DELETED before upgrading
        - Database tables have changed
        - Usernames are not case sensitive
        - Passwords no longer imported when importing a project
    - Migrating to LDAP user management
        - Considerations
    - Migrating from an Internal Directory with LDAP Authentication to a full LDAP directory.
        - Background
        - What happens when upgrading to JIRA 4.3
        - Migrating to full LDAP
            - Migrating from *Internal with LDAP authentication* to LDAP
            - Migrating from *Internal Directory* to LDAP
    - Known LDAP issues when upgrading to JIRA 4.3
- Additional JARs required when running JIRA WAR on Tomcat
    - Tomcat 5
    - Tomcat 6
- Upgrading JIRA connected to a MySQL database
- Upgrading JIRA running on a 64-bit Windows operating system
- GreenHopper
- Updated Toolkit Plugin for JIRA 4.3
- Upgrading from JIRA 4.2 with the Universal Plugin Manager installed
- Other Plugins
- Older Browsers are no longer supported
- Unsupported Modes
- Other Known Issues

- Upgrading from JIRA 4.1 and Earlier

Upgrading from JIRA 4.2 to 4.3

General upgrade instructions

Please follow the instructions in the general upgrading JIRA documentation, as well as the JIRA 4.3-specific instructions in the sections below. The general upgrade guide contains important tasks that are essential for getting your upgraded JIRA instance to work correctly (e.g. merging `jira-application.properties` customisations from the old instance to the upgraded instance).

'In-place database upgrade' is now the recommended method

JIRA 4.3 now officially supports 'in-place database upgrades' (when upgrading from JIRA 4.0.0 or later). This method requires much less downtime during the JIRA upgrade process, especially if you operate a large JIRA installation. You no longer need to export your existing JIRA data to an XML backup and then restore this data into your new JIRA version. Instead, we now support simply 'pointing' your new version of JIRA at your existing JIRA database!

Changes in `jira-application.properties`

If you are merging your old and new configuration files, as described in the Upgrade Guide, the following tables list the changes which have been made to the `jira-application.properties` file in JIRA 4.3.

The purpose of each *new* property is documented in the `jira-application.properties` file itself.

New properties in `jira-application.properties`
`jira.websudo.is.disabled = false`
`jira.websudo.timeout = 10`

As a result of changes in JIRA 4.3 to the 'Issue Links' section of the 'view issue' page, the `jira.table.cols.linkedissue` property is no longer recognised by JIRA. Modifying the value of this property in the `jira-application.properties` file will not influence the order of columns represented in this section of the 'view issue' page.

Changes in `seraph-config.xml` and Crowd Integration

When merging your old and new configuration files, as described in the Upgrade Guide, please take extra care with the `seraph-config.xml` file, since this file contains a few changed entries in JIRA 4.3.

If you simply copy your old `seraph-config.xml` to your new 4.3 installation, then:

- Due to Crowd integration changes in JIRA 4.3 (reflected in `seraph-config.xml`), your users may have authentication problems when attempting to log in to JIRA.
- If any user attempts to view a JIRA page URL which they do not have permission to access, JIRA will not explicitly indicate a permission access problem.

The following table lists the changes to the `seraph-config.xml` file in JIRA 4.3:

Elements in `seraph-config.xml` prior to JIRA 4.3	Change in JIRA 4.3

`<param-name>login.url</param-name>`
`<param-value>/login.jsp?os_destination=${originalurl}</param-value>`
	`<param-value>/login.jsp?os`
	
	`<param-value>/login.jsp?pe`

`<param-name>invalidate.session.exclude.list</param-name>`
`<param-value>ASESSIONID</param-value>`
	`<param-value>ASESSIONID</p`
	
	`<param-value>ASESSIONID,ji`

The following section
 	
`<!-- CROWD:START - If enabling Crowd SSO integration uncomment the following`
`JIRAAuthenticator and comment out the DefaultAuthenticator below -->`
`<!--`
`<authenticator class=`
`"com.atlassian.crowd.integration.seraph.v22.JIRAAuthenticator"/>`
`-->`
`<!-- CROWD:END -->`
 	
`<!-- CROWD:START - The authenticator below here will need to be commented out`
`for Crowd SSO integration -->`
`<authenticator class=`
`"com.atlassian.jira.security.login.JiraOsUserAuthenticator"/>`
`<!-- CROWD:END -->`
	Has been changed in JIRA 4.3 to
	
	`<!-- CROWD:START - If enab`
	`SSOSeraphAuthenticator`
	`<!--`
	`<authenticator class=`
	`"com.atlassian.jira.se`
	`-->`
	`<!-- CROWD:END -->`
	
	`<!-- CROWD:START - The aut`
	`for Crowd SSO integrat`
	`<authenticator class=`
	`"com.atlassian.jira.se`
	`<!-- CROWD:END -->`

`<interceptor class="com.atlassian.jira.portal.PortalPageInterceptor"/>`	Removed as this `<interceptor>`

Gadgets can only access External URLs that are on the Whitelist

Due to a security enhancement in JIRA 4.3, any external gadgets (or gadgets that make requests to external URLs) will be disabled until you add the relevant external URLs to your Whitelist.

When you first log in to JIRA 4.3 as an administrator, a message will be displayed at the top of the screen, containing a link to the 'Whitelist' page. This page will also list your external gadgets. You can either delete these gadgets, or confirm that you wish to add the relevant external URLs to your whitelist.

For more details, please see Configuring the Whitelist.

Changes to user management in JIRA

The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3. This has provided a number of additional capabilities, mainly the ability to use an LDAP server (including Microsoft Active Directory) for all user information.

When you start up JIRA 4.3, the upgrade process will automatically upgrade your user data. The sections below describe the upgrade considerations for each supported configuration type.

Upgrade considerations

For customers with internally managed users

For users that are not currently connecting to Crowd or LDAP then there are no actions required on upgrade.

For customers using LDAP for authentication

If you had previously connected JIRA to an LDAP server for authentication (using the standard method), then this configuration will automatically be acquired by JIRA when upgrading to JIRA 4.3 (or later). However, the following must be observed:

- Prior to JIRA version 4.3, the `osuser.xml` file was used to configure the connection to an LDAP server. For JIRA 4.3 (or later) to acquire these configurations automatically, your existing `osuser.xml` file **MUST be available to JIRA 4.3 (or later)** before it is started.
- If some of your users' passwords are stored in an LDAP directory but other users' passwords are stored in JIRA's internal user directory, you should **not** upgrade at this time (see JRA-23858). Please wait for JIRA 4.3.1.
- Regardless of which method you use to upgrade JIRA, when migrating your existing JIRA configurations to your new JIRA installation at the configuration migration step of either the 'in-place database upgrade' or migration procedures, ensure that you copy the `osuser.xml` file from the `atlassian-jira/WEB-INF/classes` directory of your old installation to the `atlassian-jira/WEB-INF/classes` directory of the new installation.

If you upgrade JIRA without the osuser.xml file in place, then the upgrade will proceed, but will not configure a connection to LDAP and there is no way to connect the migrated users to work with authentication via LDAP, without performing the upgrade again from a backup of JIRA

or direct manipulation of the database, which is unsupported by Atlassian.

For customers connecting to Crowd for user management

JIRA 4.3 (or later) will only connect to Crowd 2.1 or higher. If you are using an earlier version of Crowd and wish to use this Crowd configuration in your upgraded JIRA 4.3 (or later) installation, you must upgrade Crowd to version 2.1 before you upgrade JIRA.

After upgrading JIRA, you will need to wait until a synchronisation task has copied your user and group information from Crowd to JIRA's internal cache before you can log in to JIRA. If a JIRA user attempts to log in to JIRA before this synchronisation task has finished, the user's authentication will fail.

If you had previously connected JIRA to a Crowd server (using the standard method), then this configuration will automatically be acquired by JIRA when upgrading to JIRA 4.3 (or later). However, the following must be observed:

- Prior to JIRA version 4.3, the `osuser.xml` and `crowd.properties` files were used to configure the connection to Crowd. For JIRA 4.3 (or later) to acquire these configurations automatically, your existing `osuser.xml` and `crowd.properties` files **MUST be available to JIRA 4.3 (or later)** before it is started.
- Regardless of which method you use to upgrade JIRA, when migrating your existing JIRA configurations to your new JIRA installation at the configuration migration step of either the 'in-place database upgrade' or migration procedures, ensure that you copy the `osuser.xml` and `crowd.properties` files from the `atlassian-jira/WEB-INF/classes` directory of your old installation to the `atlassian-jira/WEB-INF/classes` directory of the new installation.

*If you upgrade JIRA without the osuser.xml file in place, then the upgrade will **NOT** proceed. JIRA has a table, EXTERNAL_ENTITIES, that contains some information regarding users maintained in a Crowd server. If there are entries in this table, but no osuser.xml file present, then the upgrade will stop and write a message to the log file.*

For customers with a pre-Confluence 3.5 installation that uses JIRA for user management (IMPORTANT!)

If you are a customer with a Confluence installation that uses JIRA for user management, please **do not upgrade to JIRA 4.3 *until you have first upgraded to Confluence 3.5***.

JIRA 4.3 possesses a significantly different database schema and exposes the Crowd REST interface, which Confluence will depend on for continued JIRA user management. If you upgrade to JIRA 4.3 before upgrading to Confluence 3.5, your Confluence users will no longer be able to log in until you upgrade to Confluence 3.5.

For customers who have written a custom provider for user management

Custom directory types are **not** possible and not supported in JIRA 4.3 and later.

Please see if one of the following solutions will work for you:

- If you have written a custom provider to provide LDAP support for JIRA 4.2 or earlier, you may no longer need your custom directory connector. Please check the supported LDAP schemas to see if you can use the new LDAP connectors supplied with JIRA 4.3.
- If you have written a custom provider to support nested groups, you can now use the new directory connectors supplied with JIRA 4.3. See Managing Nested Groups.
- If you have written a custom provider to connect to your own database, please consider loading the data into JIRA via the REST API instead. If you need to use an external authentication source, consider migrating to a Seraph authenticator.

If you need to keep the custom directory connection, please consider whether Atlassian Crowd meets your requirements. See the documentation on developing a custom directory connector for Crowd.

For customers with non-standard configurations

If you have a non-standard configuration and the upgrade has stopped, please contact http://support.atlassian.com.

Other considerations

Duplicate groups MUST BE DELETED before upgrading

If your JIRA instance has two groups that have the same name, but differ only by case (eg "sydney" and "Sydney"), the JIRA 4.3 upgrade will fail. You need to remove any duplicates in your current JIRA instance before upgrading. That is, you should delete one of the groups and move any users and permissions to the other group or to a new group.

(Note: There will not be a problem if your JIRA instance is connected to an external Crowd instance, and the duplicate groups are in Crowd or in an LDAP directory connected to JIRA via Crowd.)

Database tables have changed

For customers who have written programs or used other tools that access the JIRA database directly, the tables used to hold user data have changed.

Old table	New table	Information about new table
userbase	cwd_user	Holds information about the user. This now includes full name and email address.

groupbase	cwd_group	Hold information about groups.
membershipbase	cwd_membership	

These tables are indexed by directory and contain both details of all local users and groups and also cached details of external users and groups.

Usernames are not case sensitive

In version 4.3, JIRA makes no distinction between two or more usernames that only differ by case. Furthermore, username searches are case-insensitive. These behaviours are in compliance with the LDAP specification.

In version 4.2 and earlier, JIRA's handling of usernames is case-sensitive. While JIRA's user interface prevents the creation of usernames containing upper-case characters, the use of data migration or other tools may lead to a JIRA database containing mixed-case usernames.

During migration, if two users exist whose usernames differ only by case, one user will be dropped and an entry placed in the log file to record this.

JIRA's behaviour has always been undefined when two users had usernames differing only by case. There may be some side effects of the dropping of the second user including:

- A user may not be able to log in because their password was attached to the dropped user. The user will need to reset their password.

Passwords no longer imported when importing a project

When a single project is imported (see Restoring a Project from Backup), users who do not exist already in the target system are created. In releases 4.2 and earlier of JIRA, the users' passwords were also set to the passwords in the exported XML file. This is no longer the case. Users will be given randomly allocated passwords and will need to use the 'Forgotten password' link to have their passwords reset.

Note: this only relates to the 'Project Import' feature, that is, when you import a single Project from a second JIRA instance into this JIRA instance. When doing a normal full import (see Restoring Data), the passwords are preserved as usual.

Migrating to LDAP user management

Some customers may wish to migrate to managing their users in LDAP. This may particularly appeal to customers whose JIRA instance is internal, and all users are already managed in the company LDAP Directory or Microsoft Active Directory.

Considerations

- **Are user names the same?**
  - If users have the same name in the LDAP directory as they do in the internal JIRA directory, then you can simply configure a new LDAP directory and then disable the internal directory. This would also apply if you are currently using LDAP authentication.
  - If users have different names in the LDAP directory to that in JIRA, then you could configure an LDAP directory to be used for new users and leave the internal directory in place for current JIRA users.

Migrating from an Internal Directory with LDAP Authentication to a full LDAP directory.

Background

In versions of JIRA prior to 4.3 it was possible to authenticate users against an LDAP directory, but you needed to add the users, groups and memberships manually in JIRA. In JIRA 4.3, support was added that allows you to directly use an LDAP directory for users groups and memberships as well as authentication.

What happens when upgrading to JIRA 4.3

If you were using an LDAP directory for authentication and your osuser.xml configuration file is available to JIRA during the upgrade task then JIRA will create 2 user directory configuration entries:

- An Internal Directory with Delegated LDAP, this is where all your users will be defined.
- An Internal Directory, this will not contain any users, but is required. More about this later.

If the osuser.xml file is not available during the upgrade then JIRA will create 1 user directory configuration entry:

- An Internal Directory, this is where all your users will be defined. Note: The users' passwords in this directory will not be set and no users will not be able to login until you reset the passwords.

Note: If you wish to continue to use LDAP for authentication only, then ensure the osuser.xml is available during the upgrade.

Migrating to full LDAP

Ensure the LDAP directory contains the necessary JIRA Groups and Memberships. All users that are going to use JIRA must belong to a group that has JIRA Use permission, typically "jira-users".

If you cannot add Groups and memberships to your LDAP server, then you can still migrate to LDAP but will need to set the privileges to "READ-ONLY with Local Groups". Once you have completed the migration described below, you will then need to manually add all the users to the required groups. See Managing Groups.

Go to Administration/User Directories and see what directories are configured.

Migrating from *Internal with LDAP authentication* to LDAP

The steps are:

- Add a user to the Internal Directory that has System Administrator privileges.
    - Reorder the directories so that the Internal Directory is first in the list.
    - Add a new user and give that user System Administrator privileges. New users are always added to the first directory that is not READ-ONLY.
- Log out and back in to JIRA as the user you just added.
- Add a new LDAP directory to the list of User Directories.
- Wait for the LDAP directory to be fully synchronised for the first time. This may take some time depending upon the number of users and speed of the LDAP server and your connection to it.
- Re-order the directories so that the Internal Directory with LDAP authentication is at the bottom of the list and the new LDAP directory is at the top of the list.
- Logoff
- Login as a user with System Administrator privileges.
- Go to Administration / User Browser and search for the user you logged in as. The user should show as coming from the LDAP directory you just added.
- Disable the Internal Directory with LDAP authentication.

Once you have completed testing you may delete the Internal Directory with LDAP authentication.

Migrating from *Internal Directory* to LDAP

The steps are:

- Log in to JIRA with as a user with System Administrator privileges.
- Add a new LDAP directory to the list of User Directories.
- Wait for the LDAP directory to be fully synchronised for the first time. This may take some time depending upon the number of users and speed of the LDAP server and your connection to it.
- Re-order the directories so that the new LDAP directory is at the top of the list.

At this stage all the users will be in both the Internal and LDAP directory. There is no simple mechanism to remove the users from the internal directory at this time. If you wish, you can delete these users by SQL. **Manipulating the database with SQL is not supported, and is at your own risk. TAKE A BACKUP FIRST.**

Known LDAP issues when upgrading to JIRA 4.3

No content found for label(s) +ldap,+jira43.

Additional JARs required when running JIRA WAR on Tomcat

Tomcat does not come with some libraries required to run JIRA. These include database libraries that must be in the Tomcat classpath. You will need to add these libraries to your `tomcat/lib` directory

Tomcat 5

- http://www.atlassian.com/software/jira/downloads/binary/jira-jars-tomcat-distribution-5.0-rc2-tomcat-5x.zip
- If you are upgrading a previous JIRA installation, you will need to delete older versions of those libraries that you previously added to `tomcat/lib`.

Tomcat 6

- http://www.atlassian.com/software/jira/downloads/binary/jira-jars-tomcat-distribution-5.0-rc2-tomcat-6x.zip
- In particular, to prevent exceptions related to logging, please ensure that `commons-logging-.jar`, `slf4j.jar` and `log4j-1.2.15.jar` are present in Tomcat's /lib/ directory. Also ensure that these files are not present in Tomcat's webapps/jira/WEB-INF/lib directory.
- If you are upgrading a previous JIRA installation, you will need to delete older versions of those libraries that you previously added to `tomcat/lib`.

Upgrading JIRA connected to a MySQL database

If your JIRA installation is connected to a MySQL database that uses the MyISAM database engine (which is not recommended), some table indexes may not be created successfully upon upgrading to **JIRA 5.0.x**. MyISAM is not recommended for JIRA as its use can lead to serious data corruption — see JRA-24124 for details.

Before upgrading, we recommend switching your MySQL database for JIRA over to the InnoDB database engine.

Upgrading JIRA running on a 64-bit Windows operating system

If you run JIRA on a 64-bit Windows operating system, be aware that the version of Apache Tomcat (6.0.20) bundled with JIRA Standalone cannot run as a Windows service on a 64-bit JDK/JRE (see JRA-12965).

If you need to run JIRA as a Windows service on a 64-bit Windows operating system, we recommend installing the JIRA WAR-EAR distribution on Apache Tomcat version 6.0.26 or greater.

GreenHopper

Please be aware that only GreenHopper 5.5 (and later) is compatible with JIRA 4.3.x.
✅ If your existing version of GreenHopper is not compatible with JIRA 4.3.x, perform your GreenHopper upgrade immediately after performing your JIRA upgrade.

Updated Toolkit Plugin for JIRA 4.3

If you use the Toolkit Plugin with JIRA, you will need to update it to at least version 0.17 for compatibility with JIRA 4.3.

Upgrading from JIRA 4.2 with the Universal Plugin Manager installed

JIRA 4.2 supports the universal plugin manager. If you installed this plugin into your JIRA 4.2 installation, we recommend removing it from your JIRA Home directory while upgrading to JIRA 4.3 or later, as the presence of this plugin may cause problems when JIRA 4.3 is started.

When upgrading JIRA, remove this plugin *before* step 3.5 of the 'in-place database upgrade' or migration procedures.

The universal plugin manager plugin is located at:
`<JIRA-home-directory>/plugins/installed-plugins/atlassian-universal-plugin-manager-plugin-X.Y.Z.jar`

Other Plugins

JIRA 4.3 introduces several changes that may break existing plugins which are not bundled with JIRA.

If you have a developed a plugin, then please read the Plugin Developer Notes for JIRA 4.3 guide. This guide describes changes in JIRA 4.3 which may affect the compatibility of your plugin with JIRA 4.3.

If you are using a plugin developed by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.3.

Older Browsers are no longer supported

As mentioned on our End of Support Announcements for JIRA page, from JIRA 4.3, we will no longer provide support for the following platforms with JIRA:

- Safari 4 and Firefox 3.0.x browsers, as mentioned in Deprecated Web Browsers for JIRA announcement.

Please see the Supported Platforms for a list of supported browsers, databases and application servers.

Unsupported Modes

JIRA does not support running in multitenant mode.

Other Known Issues

Before you begin the upgrade, please check for known issues. Sometimes we find out about a problem with the latest version of JIRA after we have released the software. In such cases we publish information about the known issues in the JIRA Knowledge Base. Please check for known issues and follow the instructions to apply any necessary patches.

If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Upgrading from JIRA 4.1 and Earlier

In addition to the points listed above, please read the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.3 now officially supports 'in-place database upgrades' when upgrading from JIRA 4.0.0 or later. This method requires much less downtime during the JIRA upgrade process, especially if you operate a large JIRA installation. You no longer need to export your existing JIRA data to an XML backup and then restore this data into your new JIRA version. Instead, we now support simply 'pointing' your new version of JIRA at your existing JIRA database!

Plugin Developer Notes for JIRA 4.3

On this page:

- Introduction
- I18nBean no longer in the PICO container
- GadgetRequestContextFactory no longer a host component
- CacheManager deleted
- `ApplicationPropertiesImpl` default constructor has been removed
- Accessing delegators, connections and datasources in OfBiz
  - Delegator and datasource names will change in a future release
  - EntityConfigUtil refactoring
- Embedded Crowd
  - Approach to compatibility
  - Important changes
- auth-refresh required for Gadget modules

    - Extension PICO Containers are no longer supported
- New way to include browser-specific CSS
- WebSudo - Temporary Administrative Access
- JavaScript Reorganisation
    - JavaScript Namespace Changes
    - JavaScript File Changes

Introduction

JIRA 4.3 introduces several changes that may break existing plugins which are not bundled with JIRA. For more details please see the API Change documentation.

If you are using or have been involved in the development of such a plugin, it may need to be updated to work with JIRA 4.3. Please read through the information below to see if any of this content is relevant to your plugin.

If you are using a plugin developed by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.3.

Please Note:

- *This is not the complete list of changes for JIRA 4.3 — it only describes changes in JIRA 4.3 that will impact plugin developers.*
- For details about which versions of Atlassian's Plugin Development Platform and its components (such as the Plugin Framework, Shared Access Layer (SAL), Atlassian User Interface (AUI) and the Atlassian REST Plugin) are included in JIRA 4.3, please refer to Plugin Development Platform Version Matrix.

I18nBean no longer in the PICO container

The I18nBean is no longer available for dependency injection. I doubt many people used it anyway, since it was broken if you tried to use it. (The bean didn't contain any keys defined in plugins.) As we move towards a translation-as-plugin and reloadable-plugin world the problem was just going to get worse. The current, correct way is to have an I18nHelper.BeanFactory injected and call getInstance() on it. This has the additional benefit of caching (only one object per-locale) which minimizes resource-bundle scanning.

GadgetRequestContextFactory no longer a host component

Previously there were two GadgetRequestContextFactory implementations being provided: one by JIRA directly (i.e. a "host component") and one by the gadget-renderer-plugin. The host component has been removed. If you are using a GadgetRequestContextFactory you will need to ensure that your atlassian-plugin.xml has a **component-import** statement so that OSGi gets wired up properly.

CacheManager deleted

The JIRA issue cache has not been in use for quite some time, and the `CachingIssueManager` has not updated the cache for a while. This has now been completely removed out of JIRA. Previously, it may have been necessary to depend on this class to flush it, or call ManagerFactory.getCacheManager() to flush it, after certain operations. This has been a noop for quite some time, and is no longer necessary.

`ApplicationPropertiesImpl` default constructor has been removed

As of JIRA 4.3, the default parameterless constructor of the `ApplicationPropertiesImpl` class has been removed. This class was never intended to be instantiated directly from within plugin code. If you need to get an instance of the `ApplicationProperties` component, please use one of the following approaches:

- dependency injection within plugin components,
- the `ComponentManager.getComponent()` method, or
- the `ComponentLocator` injectable component

All of these approaches are compatible with versions of JIRA from 4.0.

Accessing delegators, connections and datasources in OfBiz

Delegator and datasource names will change in a future release

Calls to the following methods:

- `org.ofbiz.core.entity.GenericDelegator.getGenericDelegator("default");`
- `org.ofbiz.core.entity.ConnectionFactory.getConnection("defaultDS");`

are deprecated, and will break in a future version of JIRA. It should still work in JIRA 4.3, but it is planned that this will break in JIRA 4.4. The correct generic delegator can be injected, or if static access is needed, `com.atlassian.core.CoreFactory.getGenericDelegator()` can be used. To get a connection, JIRA has added a new class called `com.atlassian.jira.ofbiz.DefaultOfBizConnectionFactory`. This can be used using the following code:

```
new DefaultOfBizConnectionFactory().getConnection();
```

EntityConfigUtil refactoring

`org.ofbiz.core.entity.EntityConfigUtil` has been significantly refactored. To access it, call `EntityConfigUtil.getInstance()`. Accessing `DatasourceInfo` should be done using the `getDatasourceInfo()` method on the above mentioned `DefaultOfBizConnectionFactory`.

Embedded Crowd

In JIRA v4.3 we have introduced a new User Management subsystem. Please refer to the JIRA 4.3 Upgrade Guide for general information.

This means that JIRA no longer uses the OSUser framework for user management. Instead it is now using modules taken from the Atlassian Crowd application, and hence this new subsystem is commonly referred to as "Embedded Crowd".

Approach to compatibility

In order to avoid compatibility and upgrade headaches for plugins, we have taken the following measures to transition the change as smoothly as possible:

A binary compatible copy of OSUser has been left in place

This means that existing code using OSUser Objects should continue to work.

The implementation of OSUser has been altered to call through to the underlying "embedded Crowd" user management layer.

The OSUser classes will remain for at least two versions (JIRA 4.3 and 4.4), but will be removed sometime after that.

OSUser's User and Group objects have been made to implement the new User and Group interfaces

```
public class User extends Entity implements com.atlassian.crowd.embedded.api.User

public class Group extends Entity implements java.security.acl.Group,
com.atlassian.crowd.embedded.api.Group
```

This is useful because it means you can use the old User (or Group) object anywhere that is expecting the new User (or Group) object, which allows you to migrate your code piece by piece instead of trying to do it all in one go.

JIRA API methods that accept OSUser classes as input will remain but are deprecated

These methods will be removed when OSUser is removed.

For instance,

```
IssueService.getIssue(com.opensymphony.user.User user, Long issueId)
```

is now deprecated in favour of

```
IssueService.getIssue(com.atlassian.crowd.embedded.api.User user, Long issueId)
```

.

JIRA API methods that return OSUser classes as output will remain but are deprecated.

For instance,

```
GroupManager.getGroup(String groupname);
```

will continue to return `com.opensymphony.user.Group`, but it is deprecated and replaced by

```
GroupManager.getGroupObject(String groupname);
```

which returns `com.atlassian.crowd.embedded.api.Group`.

JIRA Plugin interfaces

Interfaces for plugin points sometimes include a User object in them (eg JqlFunction).

In these cases, we will leave the OSUser object in the short term for the sake of compatibility, and the interface will be altered when OSUser

is removed.

Non-API Classes

Classes that are not considered part of JIRA's public API (Implementation classes) may be converted to the new User object at any time. Plugin developers should be avoiding these anyway.

New methods

New methods in API will only use the new User/Group interfaces.

Converting between old and new User objects.

The "old" User object has been made to implement the new User interface, and so no conversion is required in this direction.

You should not need to convert from the new to the old user Object very often, but if you do a Utility class `OSUserConverter` is available.

Important changes

Don't use `com.opensymphony.user.UserManager`

This is a static factory class that implemented OSUser. It is supported in the short term but will be removed.

Use JIRA's dependency-injected Managers and Services instead:

- com.atlassian.jira.user.util.UserManager
- UserUtil (an extended UserManager)
- UserService
- GroupService
- GroupManager

Changes to the User object.

User-names are now case-insensitive. This is to reflect the way user-names are treated by LDAP.

Previously user-names were case-sensitive but forced to be all lower-case, so this change should not cause problems.

- getEmail() is deprecated to be replaced by getEmailAddress()
- getFullName() is deprecated to be replaced by getDisplayName()

DirectoryID

JIRA is now capable of connecting to multiple "User Directories" at once. A User Directory might be an LDAP server, a Crowd Server, or the "Internal Directory" (that is, users stored in JIRA's DB).

In order to tell which directory a User came from, there is a DirectoryID added to the User object. This means that in theory, you can have two users from two directories with the same username. (Note that this should be considered a rare and unusual situation — documentation will recommend that users try to avoid this).

For this reason, the username and directoryID are both considered in `User.equals()`.

User Properties

OSUser allows you to store custom properties in a `PropertySet` against a User. This will continue to be supported, but the `PropertySet User.getPropertySet()` method is deprecated. Use the `UserPropertyManager` to get user properties.

Changes to Seraph

If you are using a custom authenticator, note that two Seraph methods that were previously implemented in `DefaultAuthenticator` to use OSUser have become abstract methods:

```
protected abstract java.security.Principal getUser(java.lang.String s);

protected abstract boolean authenticate(java.security.Principal principal, java.lang.String s)
throws com.atlassian.seraph.auth.AuthenticatorException;
```

There is a concrete class in JIRA 4.3 called `JiraSeraphAuthenticator` which extends the abstract `DefaultAuthenticator`, implementing the above two abstract methods using Embedded Crowd. This means that, if you have written your own custom authenticator by extending `DefaultAuthenticator`, you will need to either:

- continue to extend `DefaultAuthenticator`, but make sure you implement both abstract methods; or
- extend `JiraSeraphAuthenticator` instead, which has already implemented those methods.

auth-refresh required for Gadget modules

You need to add the `auth-refresh` feature to your Gadget module preferences inside the `gadget.xml` file:

```
<Optional feature='auth-refresh' />
```

This enables your gadget to refresh its authentication token, which allows it to make requests. If you don't include the `auth-refresh` feature, a JavaScript error will occur if your gadget makes a request after 30 minutes of inactivity.

Extension PICO Containers are no longer supported

Please note that from JIRA 4.3 onwards, the use of `jira.extension.container.provider` in jira-application.properties is no longer supported.

New way to include browser-specific CSS

Previously when targeting a specific browser (e.g. IE6/IE7/IE8), you would put all CSS styles in a separate stylesheet and rely on IE-conditional comments to restrict which browsers received the files.

In JIRA 4.3 we have added some JavaScript which adds classes to the HTML tag on page load. This means that you can now put all browser-specific CSS fixes in the main stylesheet near related styles. Having styles in one file increases maintainability, and reduces the number of requests — which helps pages load faster.

Prior to 4.3 you would put IE styles in the IE stylesheet and prefix with * or _ depending on the browser. Now you can use the following:

- .msie (all versions of IE)
- .msie-7 (just IE7)
- .msie-8 (just IE8)
- .msie-gt-7 (IE8 and IE9)
- .msie-lt-8 (IE7, IE6, IE5.5, etc)
- .mozilla (all versions of Firefox. There are no version-specific options like IE)
- .webkit (all versions of Chrome/Safari. There are no version-specific options like IE)
- .opera (note: not a supported browser)

WebSudo - Temporary Administrative Access

We have added an extra layer of authentication to the administration actions in JIRA. If your plugin adds actions to the administration area, that is Actions that should only be for accessible by users with the admin or sys-admin roles, the Action should be annotated with WebSudoRequired on the class (not the action or package). If your administrative pages are not actions you can use the WebSudoManager following the instructions at Adding WebSudo Support to your Plugin.

JavaScript Reorganisation

To improve the consistency of JavaScript resources in JIRA 4.3, the namespaces of various JavaScript objects and functions have been changed and some JavaScript files have been moved or renamed.

JavaScript Namespace Changes

The table below lists all namespace changes to JavaScript objects and functions in JIRA 4.3. If your plugin uses JavaScript which refers to any of these objects and functions by their old namespace, you will need to update these in your plugin to the new namespace.

Old namespace	New namespace
`jira.app.issue`	`JIRA.Issue`
`jira.app.issuenavigator`	`JIRA.IssueNavigator`
`jira.app.issuenavigator.shortcuts`	`JIRA.IssueNavigator.Shortcuts`
`AJS.DropDown`	`AJS.Dropdown`
`jira.widget.dropdown`	`JIRA.Dropdown`
`AJS.containDropdown`	`JIRA.containDropdown`
`AJS.SelectMenu`	`AJS.DropdownSelect`
`AJS.SecurityLevelMenu`	`AJS.SecurityLevelSelect`

AJS.QueryableDropdown	AJS.QueryableDropdownSelect
AJS.IssuePicker	JIRA.IssuePicker
AJS.LabelPicker	JIRA.LabelPicker
jira.widget.favourite.Picker	JIRA.FavouritePicker
AJS.FlexiPopup	JIRA.Dialog
AJS.FormPopup	JIRA.FormDialog
AJS.LabelsPopup	JIRA.LabelsDialog
AJS.IssueActionsPopup	JIRA.IssueActionsDialog
AJS.UserProfilePopup	JIRA.UserProfileDialog
jira.app.attachments.screenshot.ScreenshotWindow	JIRA.ScreenshotDialog
jira.widget.autocomplete	JIRA.AutoComplete
jira.widget.autocomplete.JQL	JIRA.JQLAutoComplete
jira.ajax	JIRA.SmartAjax
jira.app.wikiPreview	JIRA.wikiPreview
jira.app.userhover	JIRA.userhover
jira.app.session.storage	JIRA.SessionStorage
jira.xsrf	JIRA.XSRF

JavaScript File Changes

Along with the JavaScript namespace changes (above), several JavaScript files have been moved or renamed. Any plugins that load JIRA's JavaScript files directly may encounter "Resource not found" warnings due to this change.

Instead of loading these files directly, we recommend using the Web Resource framework to include JIRA's JavaScript files with plugins, as this method is backwards-compatible with JIRA 4.2.

```
<web-resource key="my-resource">
  <dependency>jira.webresources:jira-global</dependency>
</web-resource>
```

JIRA 4.3.4 Release Notes

27 May 2011

The Atlassian JIRA team announces the release of **JIRA 4.3.4**. This point release contains several updates and fixes, plus version 2.6 of the JIRA Importers Plugin, providing:

- support for Trac import; and
- improvements to CSV import, including the ability to import multiple columns of the same name (for fields that are multi-valued in JIRA) and the ability to import sub-tasks.

JIRA 4.3.4 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.3 yet?

Take a look at all the new features in the JIRA 4.3 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.3.4 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.3.4 includes the following updates and bug fixes:

JIRA Issues (25 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-24713	Source download for 4.3.4 not available		Resolved
	JRA-23691	Bundle in JIRA the correct driver for Microsoft SQL Server		Resolved
	JRA-24644	javax.mail system properties do not get passed along to the javax.mail session for POP and IMAP servers		Resolved
	JRA-24516	When navigating to issue navigator via return to search link, the window is blurred behind other windows in all versions of IE		Resolved
	JRA-24488	Change of Tomcat binary package for Windows breaks HTTPS		Resolved
	JRA-24466	Upgrade Task Build 602 leaves the Membership sequence in an inconsistent state		Resolved
	JRA-24406	Add CrowdEmbedded to JIRA source distribution		Resolved
	JRA-23969	"Internal with LDAP Authentication" directory needs to be able to follow referrals		Resolved
	JRA-23843	Dot dialog dropdown is missing scrollbars		Resolved
	JRA-25509	Javadocs are not including all the Atlassian classes		Resolved
	JRA-24517	Database connection leak in JRA15731Check		Resolved
	JRA-24479	Performance issues in web work configuration		Resolved
	JRA-24468	Update to v4.3.3 is unable to install as service when v4.3.2 service is still installed		Resolved
	JRA-24442	JIRA's CustomField class implementation does not respect JiraRendererPlugin.transformFromEdit()		Resolved
	JRA-24269	SAL: I18nResolver.getAllTranslationsForPrefix(key, locale) does not return raw format string		Resolved
	JRA-24176	JQL Statement using combination of WAS and BEFORE and AFTER keywords results in NullPointerException		Resolved
	JRA-24059	History status search for renamed statuses does not work on Oracle/MSSQL DB		Resolved
	JRA-24403	Stack trace in logs when user select "Printable" view from issue Navigator		Resolved
	JRA-24380	Using keyboard to navigate calendar popup moves 2 days at a time		Resolved
	JRA-24376	Link issue dialog looks rubbish in Chrome		Resolved

	JRA-24270	header.js needs to have a dependency declared on jira.webresources:jira-global		Resolved
	JRA-24208	JIRA presents a message "You cannot view this page as a guest" even when logged in		Resolved
	JRA-24158	The Jira Plugin Manager logs an un-necessary warning and stack-trace when disablePluginWithoutPersisting is called		Resolved
	JRA-23956	"Reindexing required" notification has null name		Resolved
	JRA-23350	Attachment Link in the comment pointing to the older attachment instead of the latest		Resolved

JIRA 4.3.4 Upgrade Guide

Upgrading from JIRA 4.3.x to 4.3.4

Please follow the instructions in the general upgrading JIRA documentation.

Upgrading from JIRA 4.2.x and earlier

In addition to the above, please read the JIRA 4.3 Upgrade Guide and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.3.3 Release Notes

29 April 2011

The Atlassian JIRA team announces the release of **JIRA 4.3.3**. This point release contains several updates and fixes, plus version 2.3.1 of the JIRA Importers Plugin, providing support for Pivotal Tracker and improvements in the way attachments are imported. The FishEye plugin version 3.1.5 is also included.

JIRA 4.3.3 also delivers some improvements with LDAP and Crowd directories, including:

- Synchronisation performance gains of up to 400% in some cases.
- Changed behaviour for automatic group membership in LDAP directories: In previous releases, users were added to the given groups each time they logged in. In JIRA 4.3.3 and later, automatic group assignment will happen only the first time the user logs in. This allows administrators to remove group memberships after they have been granted. For more information, see Connecting to an LDAP Directory.
- Automatic user creation for internal directories with LDAP authentication: You can now configure the directory to automatically copy the user from LDAP to the internal directory on first login, and to add the user to given groups. See Connecting to an Internal Directory with LDAP Authentication.
- More directory types available when connecting to an internal directory with LDAP authentication: You can now select from a number of LDAP directory servers. See Connecting to an Internal Directory with LDAP Authentication.

JIRA 4.3.3 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.3 yet?

Take a look at all the new features in the JIRA 4.3 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.3.3 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.3.3 includes the following updates and bug fixes:

JIRA Issues (31 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-24221	Internal with LDAP Authentication does not have copy user on first login option		Resolved
	JRA-24188	Use FilteredCrowdServiceImpl as the implementing class of the CrowdService interface		Resolved

	JRA-24166	Cannot Attach a Screenshot After Upgrade to JIRA 4.3		Resolved
	JRA-24164	JiraServiceContextImpl breaks backwards compatibility		Resolved
	JRA-23871	Need a way to force a re-synch on the FE config.		Resolved
	JRA-22714	The preview button does not toggle.		Resolved
	JRA-24321	'Enable profiling' causes the Attach Files function to fail		Resolved
	JRA-24306	{noformat} block horizontally scrolling is inconsistent across browsers		Resolved
	JRA-24300	Printable View in Issue Navigator is broken, logs show missing printable.css and global-static		Resolved
	JRA-24206	Performance issues when configuring issue navigator columns with lots of custom fields		Resolved
	JRA-24156	When Confluence is connected to JIRA, users in directories defined after the JIRA directory will fail to authenticate		Resolved
	JRA-24126	Add "Help" link on the "System Info" page		Resolved
	JRA-23984	Display bug on Simple Issuenav Search for radio/checkbox custom field descriptions		Resolved
	JRA-23945	activity streams broken in jira-plugin-test-resources 4.3		Resolved
	JRA-23933	'Create New User' doesn't allow you to choose which user directory to add the user into.		Resolved
	JRA-23755	FileService doesn't allow to specify a sub-directory inside the import/mail directory		Resolved
	JRA-23667	"Raphael is null or not an object" Javascript error when any dropdown is clicked		Resolved
	JRA-22903	padlock menu - move bullet selection down 2 pixels		Resolved
	JRA-20001	"Columns to Display" cannot be selected in a Filter Results gadget when a field with a name longer than the width of the multi-select box exists		Resolved
	JRA-19404	Browse Project UI elements do not use colours from look and feel configuration		Resolved
	JRA-24627	Horizontal lines separating issues are missing in printable view for issue navigator		Resolved
	JRA-24255	HTML Emails link to global-static.css, which is not available		Resolved
	JRA-24248	Upgrade Crowd dependencies to Crowd 2.2.5 and EmbCwd 1.2.4		Resolved
	JRA-24211	i18n Date Picker Issue: wk --> KW in German		Resolved
	JRA-24075	JIRA announcement banner		Resolved
	JRA-23943	WebSudo action throwing NPE when user not logged in		Resolved
	JRA-23804	Error message in BrowseProjects page frormated incorrectly on IE		Resolved
	JRA-23650	Announcement banner text font is blurry in Issue Page using IE 8		Resolved
	JRA-23563	error in #controlFooter() templates/plugins/jira/macros.vm		

				Resolved
	JRA-22900	OTS: dialog tweak - keyboard shortcuts		Resolved
	JRA-22849	remove vertical separator when there's nothing to the right		Resolved

JIRA 4.3.3 Upgrade Guide

Upgrading from JIRA 4.3.x to 4.3.3

Please follow the instructions in the general upgrading JIRA documentation.

Upgrading from JIRA 4.2.x and earlier

In addition to the above, please read the JIRA 4.3 Upgrade Guide and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.3.2 Release Notes

11 April 2011

The Atlassian JIRA team announces the release of **JIRA 4.3.2**. This point release fixes two issues (JRA-24251 & JRA-24291) associated with the Windows Installer.

JIRA 4.3.2 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.3 yet?

Take a look at all the new features in the JIRA 4.3 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.3.2 Upgrade Guide.

JIRA 4.3.2 Upgrade Guide

Upgrading from JIRA 4.3.x to 4.3.2

Please follow the instructions in the general upgrading JIRA documentation.

Upgrading from JIRA 4.2.x and earlier

In addition to the above, please read the JIRA 4.3 Upgrade Guide and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.3.1 Release Notes

1 April 2011

The Atlassian JIRA team announces the release of **JIRA 4.3.1**. This point release contains several updates and fixes, plus version 2.1 of the JIRA Importers Plugin, providing improved support for Bugzilla. Please see the documentation: Importing Data from Bugzilla.

JIRA 4.3.1 now supports anonymous binding when connecting to LDAP directories. Hence, when configuring an LDAP directory or an internal directory with LDAP authentication for user management in JIRA, you do not need to specify a **Username** and **Password** for JIRA to access your LDAP directory.

Notes for plugin developers:

- JIRA 4.3.1 resolves a Jersey-related issue that was preventing the building of plugins against 4.3.
- JIRA 4.3.1 provides backwards compatibility for the **JavaScript Namespace Changes** listed in the Plugin Developer Notes for JIRA 4.3.

The version of Tomcat included in JIRA 4.3.1 Standalone has been updated to Tomcat 6.0.32.

JIRA 4.3.1 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.3 yet?
Take a look at all the new features in the JIRA 4.3 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.3.1 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.3.1 includes the following updates and bug fixes:

JIRA Issues (38 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-11174	auto create user on login - with LDAP authentication		Resolved
	JRA-24138	JIRA startup can fail non-deterministically with NullPointerException		Resolved
	JRA-23886	Web sudo setting is cached		Resolved
	JRA-23876	Plugin System upgrade tasks should not be run before JIRA's own upgrade tasks are run		Resolved
	JRA-23846	Jersey 1.0.3 in JIRA 4.3-RC1		Resolved
	JRA-23756	resource leakage and misleading log message when a bundled plugin gets upgraded		Resolved
	JRA-23595	Upgrade Task for labels can initially fail but then pass on second run with out converting a lot of the data		Resolved
	JRA-22633	JIRA starts up in unworkable stage if core plugin is not started		Resolved
	JRA-24143	When upgrading plugins that provide plugin points, the clients don't come up properly		Resolved
	JRA-23946	Failed jira.home startup check leaves JIRA broken		Resolved
	JRA-23926	XSRF token broken when you edit an Issue Type Scheme		Resolved
	JRA-23887	User Directory page shows 500 page after session timeout		Resolved
	JRA-23858	Upgrade path for delegated LDAP locks out internal JIRA users		Resolved
	JRA-23824	two dimensional filter gadget: link to subset contains wrong filter		Resolved
	JRA-23820	Error message displayed in errors page need to link to the upgride guide		Resolved
	JRA-23819	LDAP anonymous bind is not allowed in JIRA v4.3		Resolved
	JRA-23763	UAL Trusted apps dialog misleads a user trying to log into another system if they are not a sys admin		Resolved
	JRA-23551	fault JQL searches created for Label Custom Fields within 2-dimensional statistic gadget and others		Resolved
	JRA-23522	edit-searcher/search-radio.vm [line 10,column 16] : ${checked} is not a valid reference.		Resolved
	JRA-22431	Decimal estimate time prevents saving of edited issues		Resolved
	JRA-22397	DefaultOSWorkflowConfigurator.JiraTypeResolverDelegator should fall back to JiraUtils.loadComponent(objClass) to load the class if type resolver is not found		Resolved

	JRA-20286	Date resolved is updated whenever resolution is included in transition screen		Resolved
	JRA-17976	project Avatar: Trying to create a project with custom avatar and custom permission scheme give java.lang.IllegalStateException error		Resolved
	JRA-24145	Upgrade to JIM 2.1		Resolved
	JRA-24116	README.txt incorrect script names		Resolved
	JRA-24114	UserBrowser returns an Iterable instead of an Iterator		Resolved
	JRA-23927	Users should be able to have correct docs for /transitions sub-resource		Resolved
	JRA-23802	Remove menu link to 'Support Request' from the JIRA admin		Resolved
	JRA-23787	change portlets to gadgets in i18 file		Resolved
	JRA-23679	Duplicate language displayed when default language is also in a language pack		Resolved
	JRA-23580	Projects gadget on dashboard not displaying projects in correct categories		Resolved
	JRA-23339	Incorrect JQL Created from Pie Chart Gadget for Label Custom Fields		Resolved
	JRA-22638	New line removed in log work description using Default Text Renderer		Resolved
	JRA-22198	typo in "Date Range Selector" screen: first line of help text should end with a full-stop instead of comma.		Resolved
	JRA-22179	Upgrade tomcat app server for standalone to tomcat 6.0.32		Resolved
	JRA-21880	NullPointerException when there are no cookies and AccessLogRequestInfo is enabled		Resolved
	JRA-20975	Time Tracking Field Accepts more than one entry unqualified by a time unit		Resolved
	JRA-23952	Typo in usage message for start-jira.sh		Resolved

JIRA 4.3.1 Upgrade Guide

Upgrading from JIRA 4.3 to 4.3.1

Please follow the instructions in the general upgrading JIRA documentation.

Upgrading from JIRA 4.2.x and earlier

In addition to the above, please read the JIRA 4.3 Upgrade Guide and as well as the Important Version-Specific Upgrade Notes for the versions of JIRA you are skipping.

JIRA 4.2 Release Notes

21 October 2010

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

JIRA 4.2 gives you a few extra minutes of precious time every day, by providing the ability to triage issues directly from the Issue Navigator without you having to open each issue. For the mouse-averse, the new 'Operations Dialog' box provides access to all menu options via the keyboard.

To help speed up your ability to work on issues, we've also made common actions and workflow operations accessible from convenient dialog boxes.

Time-tracking has become much more flexible: you can now edit the Original Estimate, and set the Remaining Estimate to zero when resolving an issue. You can also log work via workflow 'transition' (or any other) screens.

The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users and administrators — and everyone else who likes to label their JIRA issues.

Note to developers: JIRA 4.2 includes the Atlassian Plugin Framework version 2.5, and an alpha release of the JIRA REST API. We are very keen to hear your feedback on the REST API — please try it out and add your comments to JRA-22139.

Upgrading to JIRA 4.2 is free for all customers with active JIRA software maintenance as of October 21, 2010.

Highlights of JIRA 4.2:

- Dialogs for Common Actions and Workflow Operations
- Keyboard Shortcuts and 'Operations Dialog'
- Issue Labelling
- Improvements to the 'View Issue' Screen
- Auto-complete for Versions and Components
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- 'Filter' Gadget creation via the Issue Navigator
- User Avatars and Hover Profile
- New JQL Functions
- Other Enhancements and Fixes
- Plus over 120 other fixes and improvements

Thank you for your feedback:

More than 40 new feature requests implemented

More than 850 votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

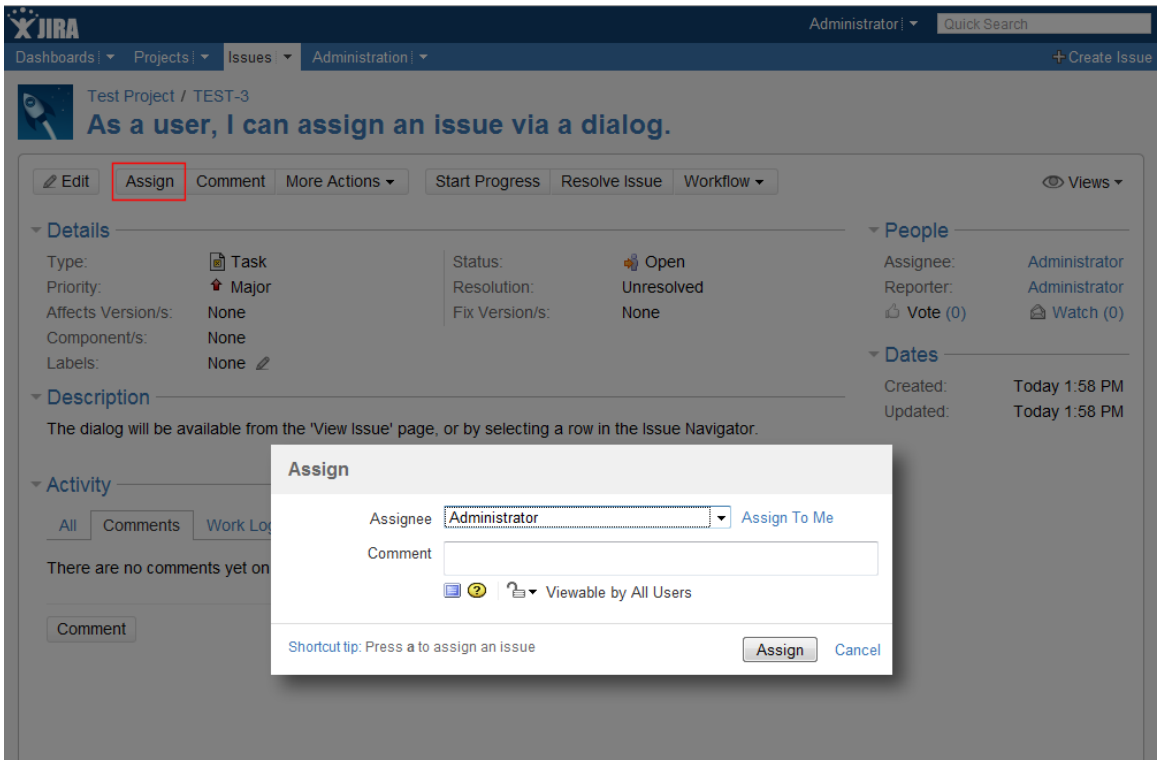
Upgrading to JIRA 4.2

JIRA 4.2 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 4.2 Upgrade Guide.

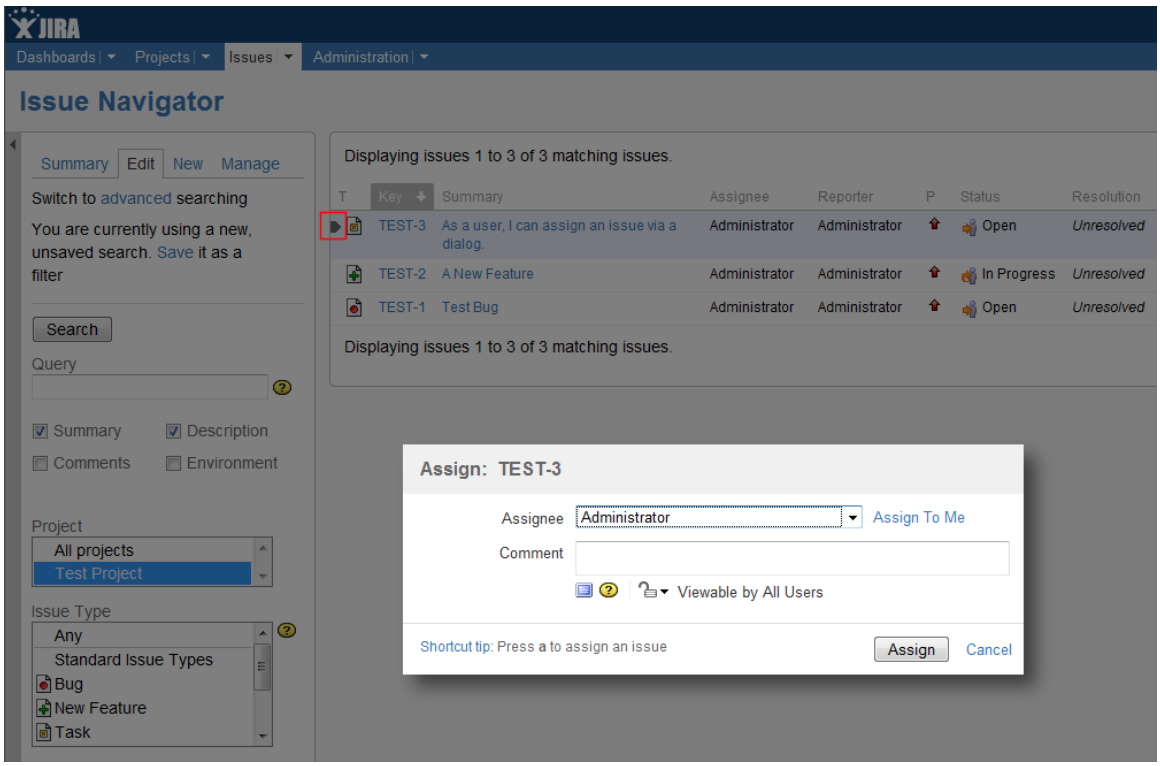
Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

For faster edits to an issue, pop-up dialogs have replaced screen changes for common actions and workflow operations.



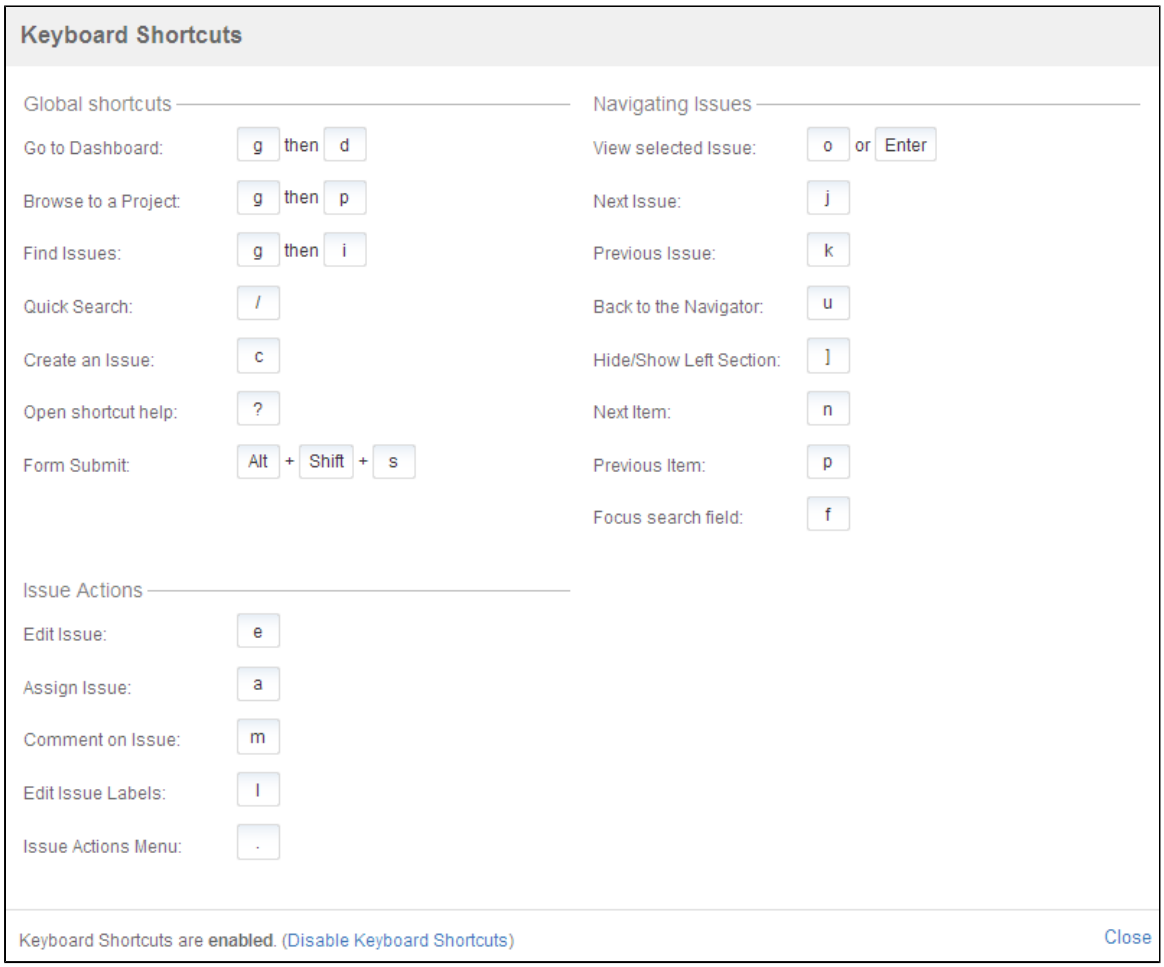
You can now perform an action on an issue (via a dialog) directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



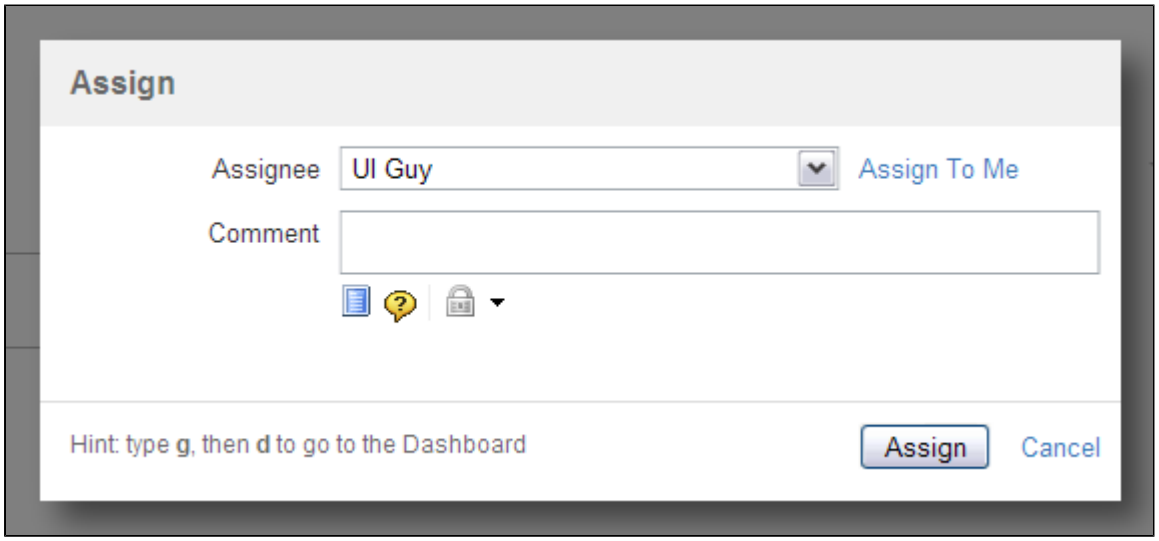
^Top

Keyboard Shortcuts and 'Operations Dialog'

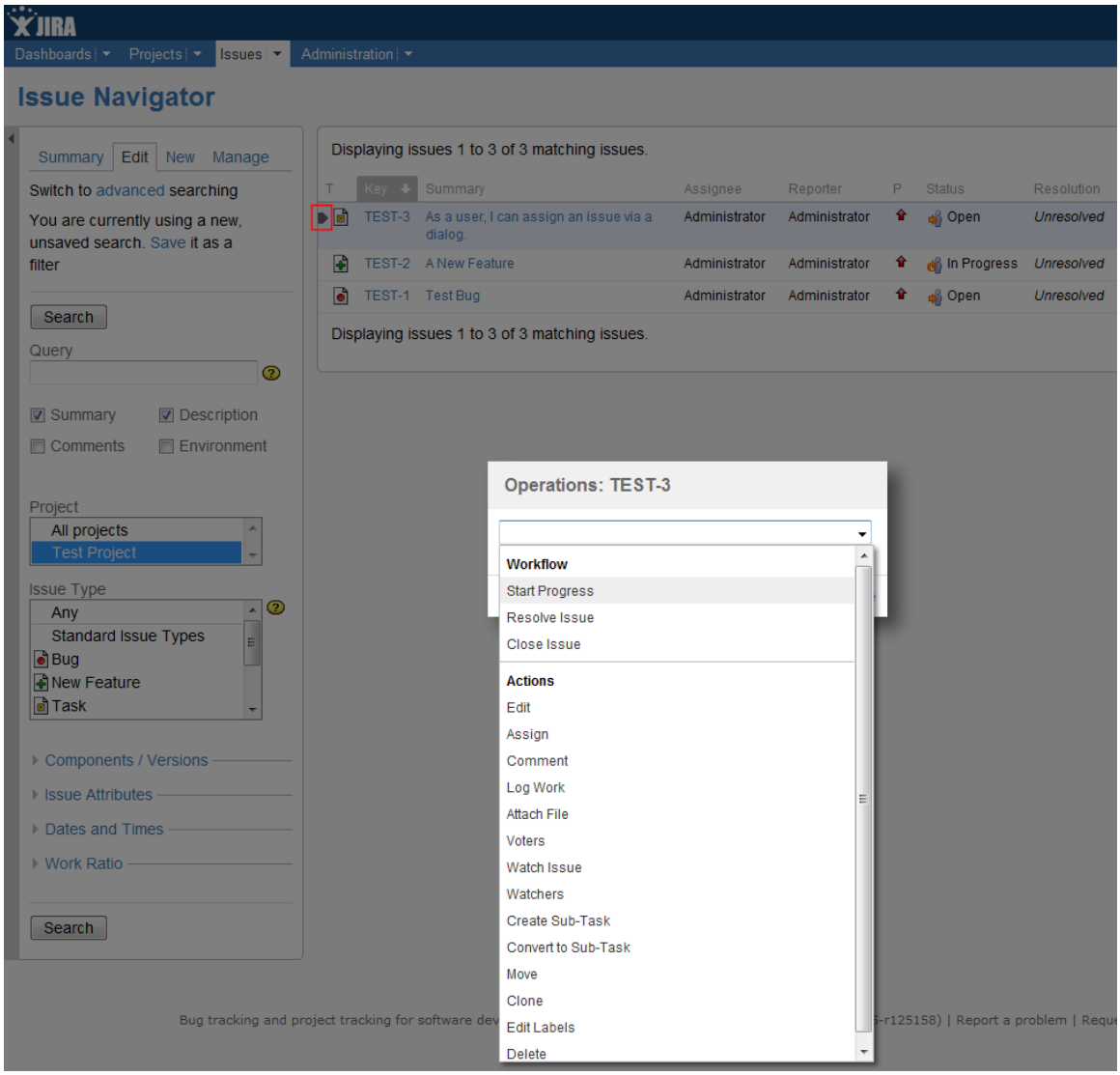
JIRA users can now do more without a mouse — perfect for power users:



Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly:



The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.

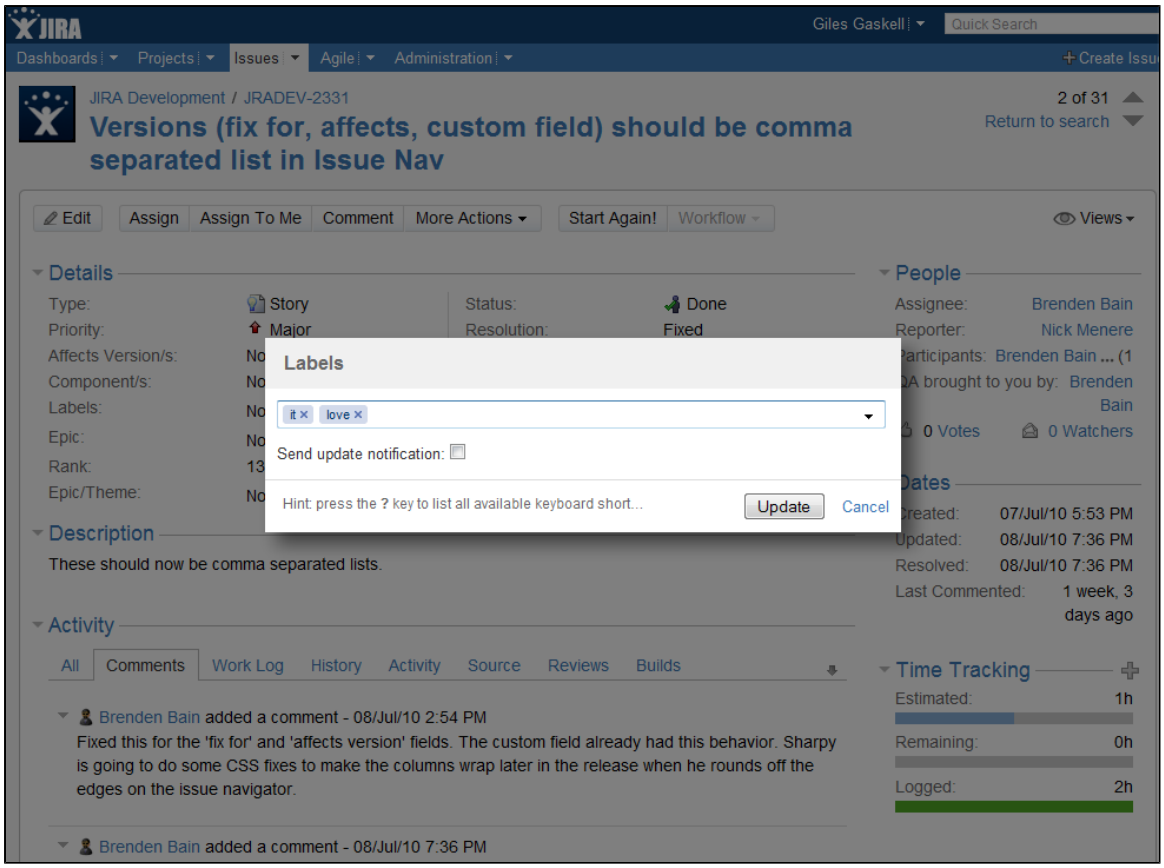


See Using Keyboard Shortcuts for more information.

^Top

Issue Labelling

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and email notifications are now optional when updating labels for an issue.



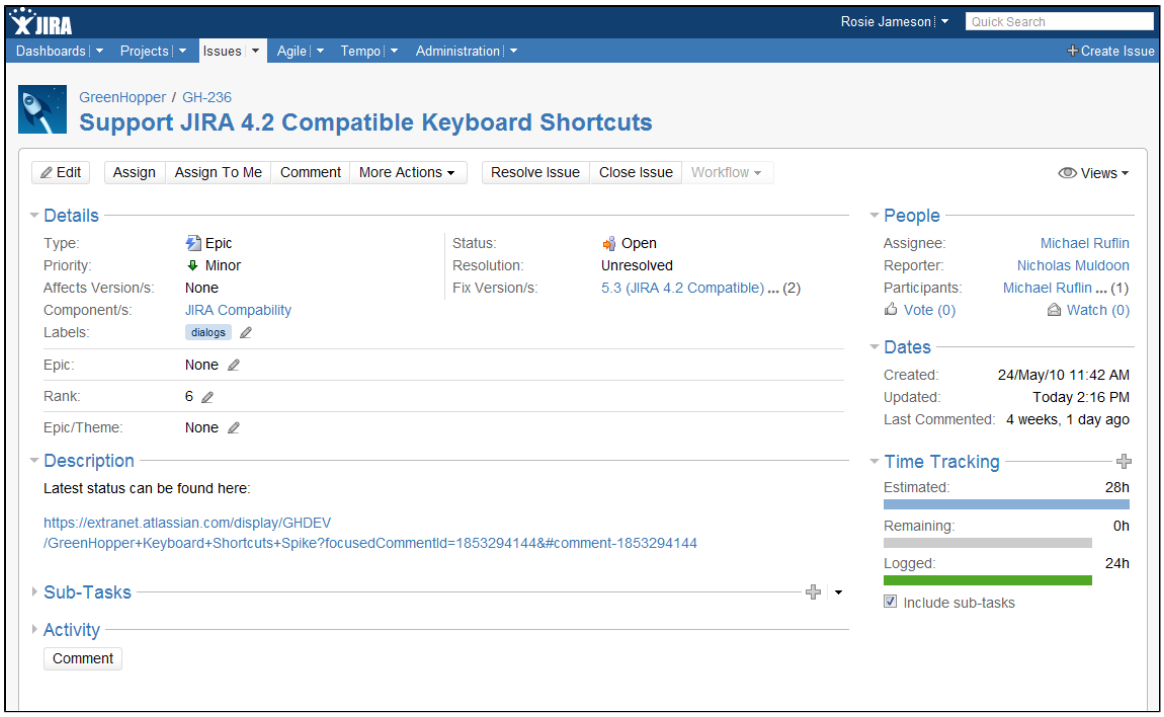
See Labelling an Issue for more information.

^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the 'View Issue' screen even more. Some improvements include:

- Tighter spacing between lines to reduce space between elements
- Stronger section headings
- The 'Description' field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



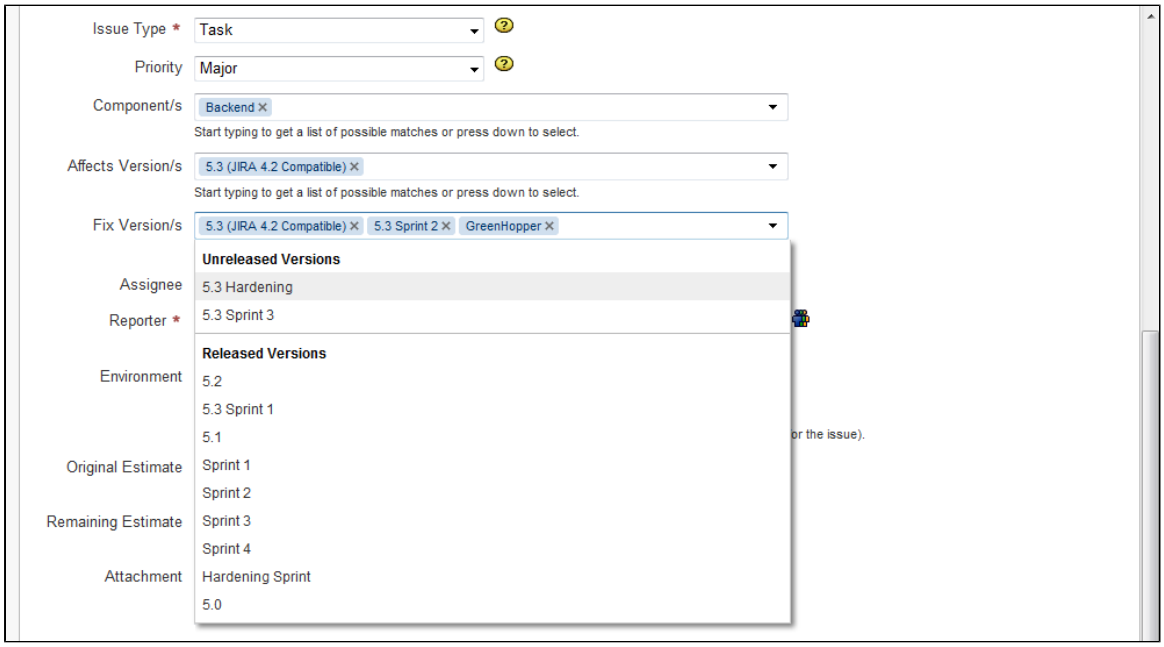
Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. You can also add a comment at the bottom of the '**Comments**' section.

^Top

Auto-complete for Versions and Components

Editing the **Component**, **Affects Version(s)** and **Fix Version(s)** is now quicker and easier. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

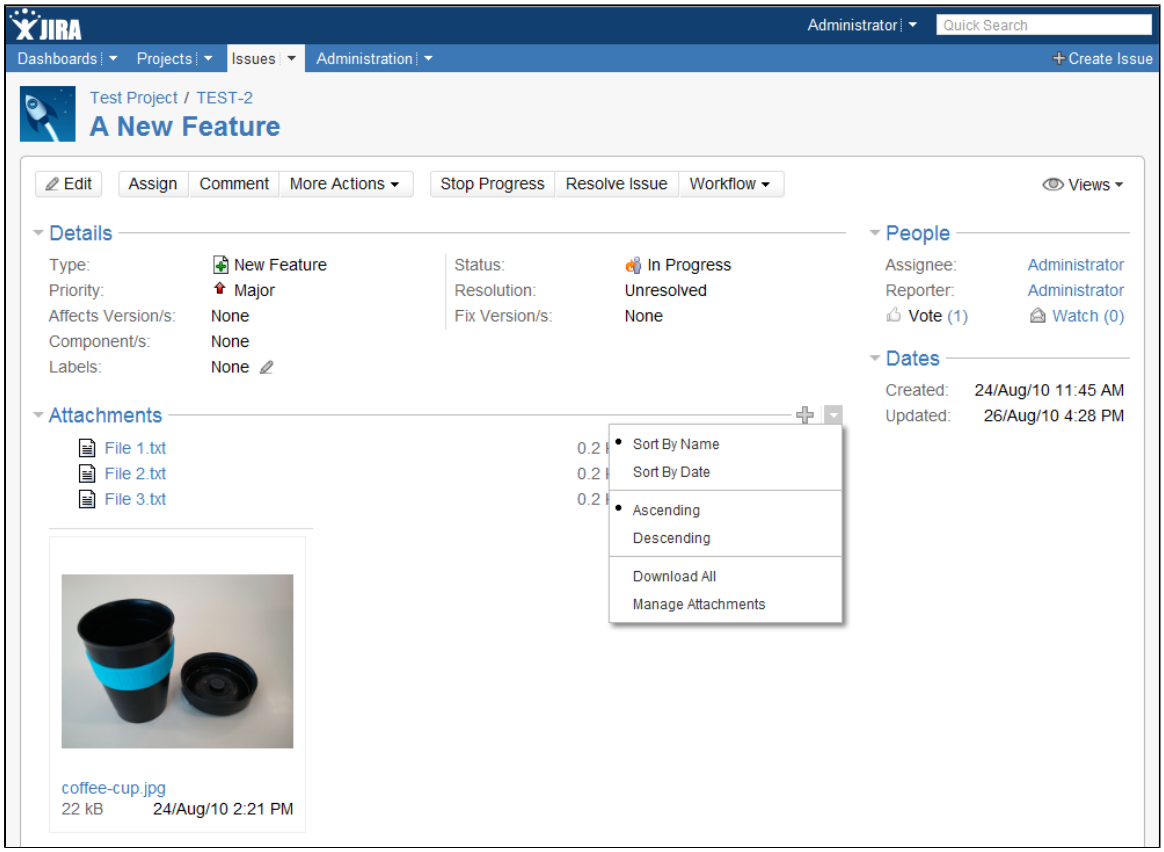
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

Attachments can now be sorted by date or by name, in ascending or descending order.

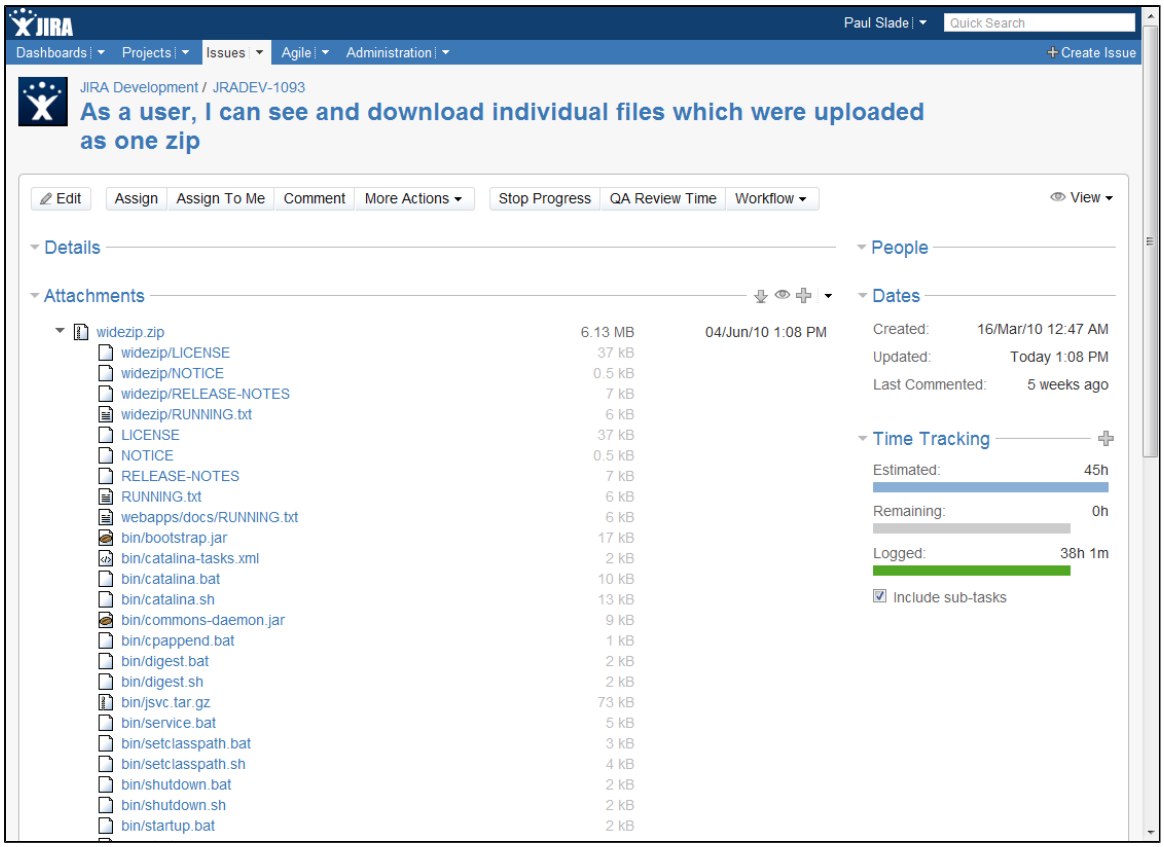


See Attaching a File for more information.

^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



See Accessing ZIP-format File Contents for more information.

^Top

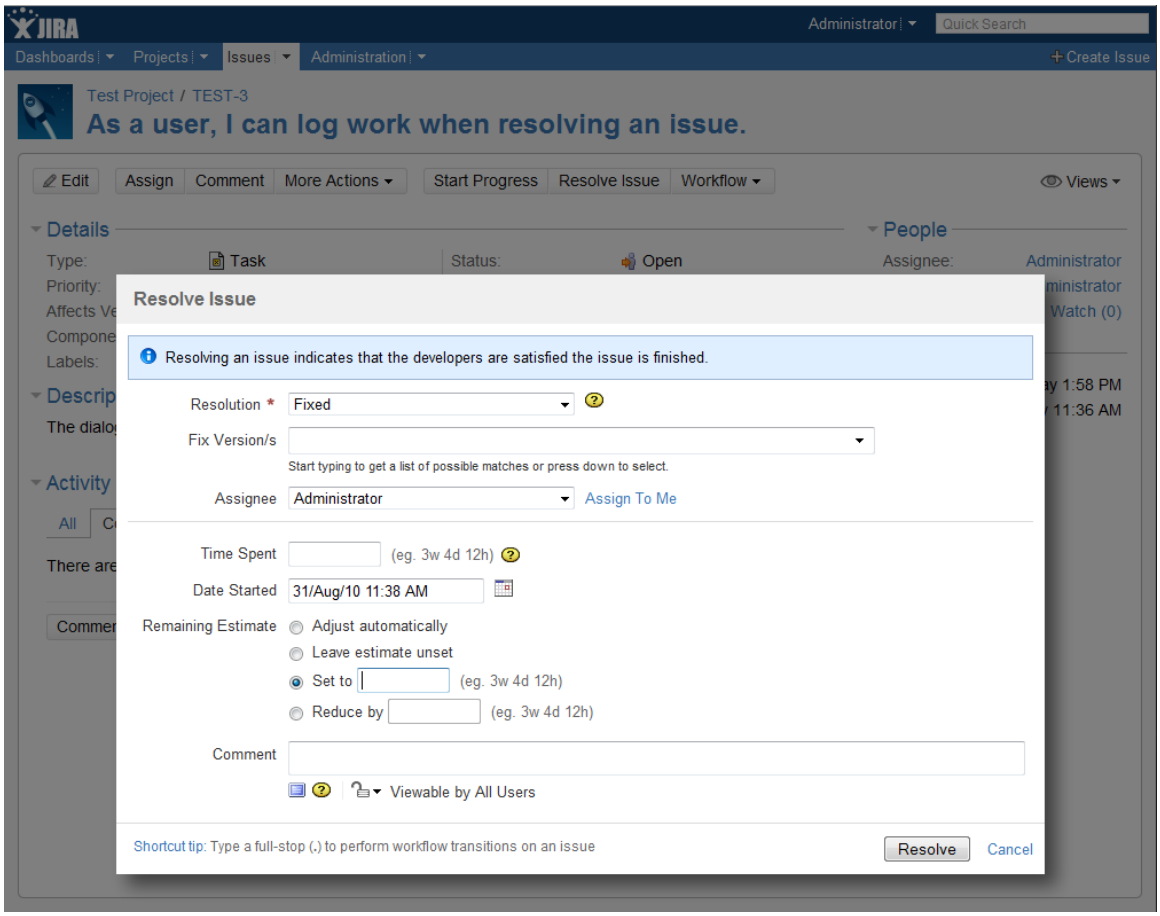
'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).

Lastly, you can add the 'Log Work' fields to any screen in JIRA (not shown here).



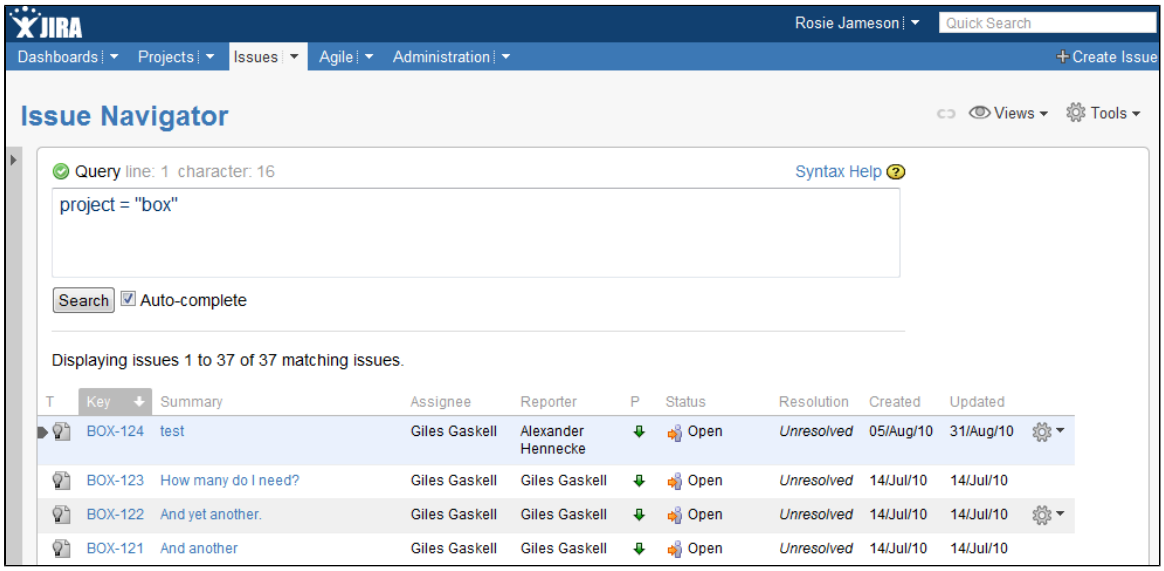
See Logging Work on an Issue for more information.

^Top

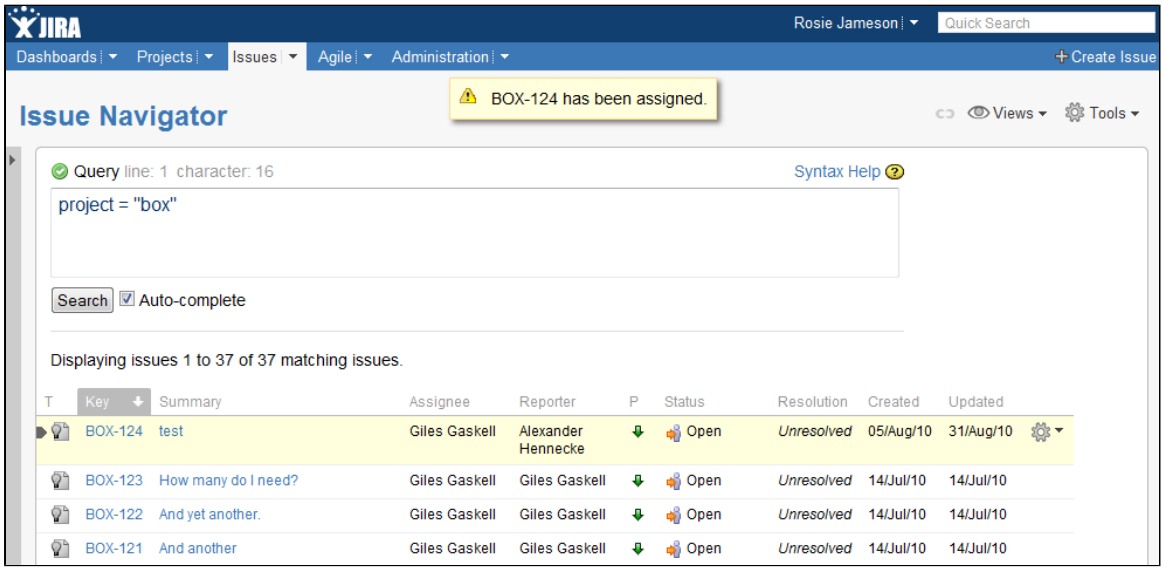
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



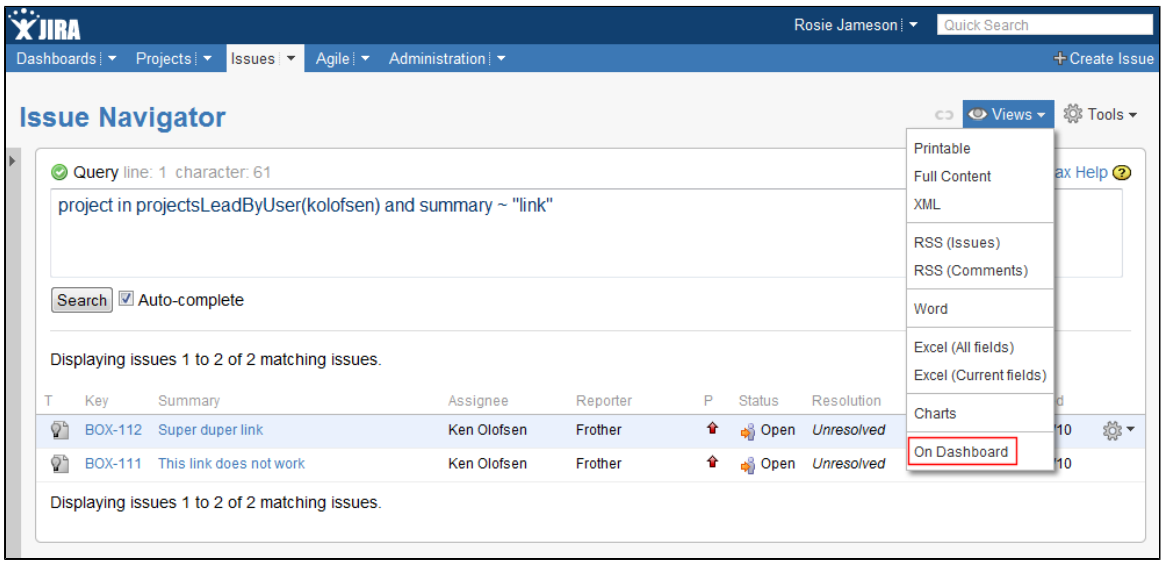
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

'Filter' Gadget creation via the Issue Navigator

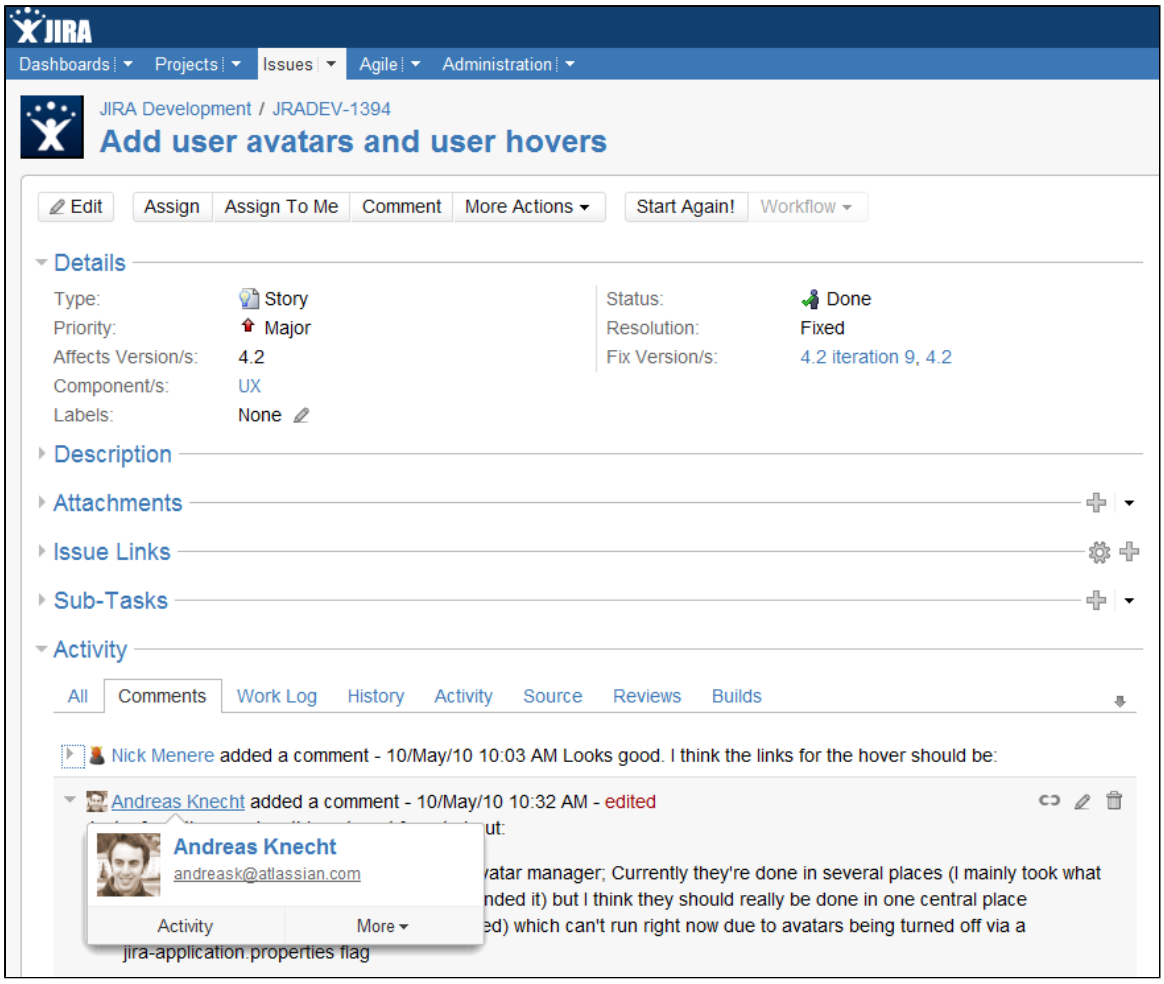
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



^Top

User Avatars and Hover Profile

JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.

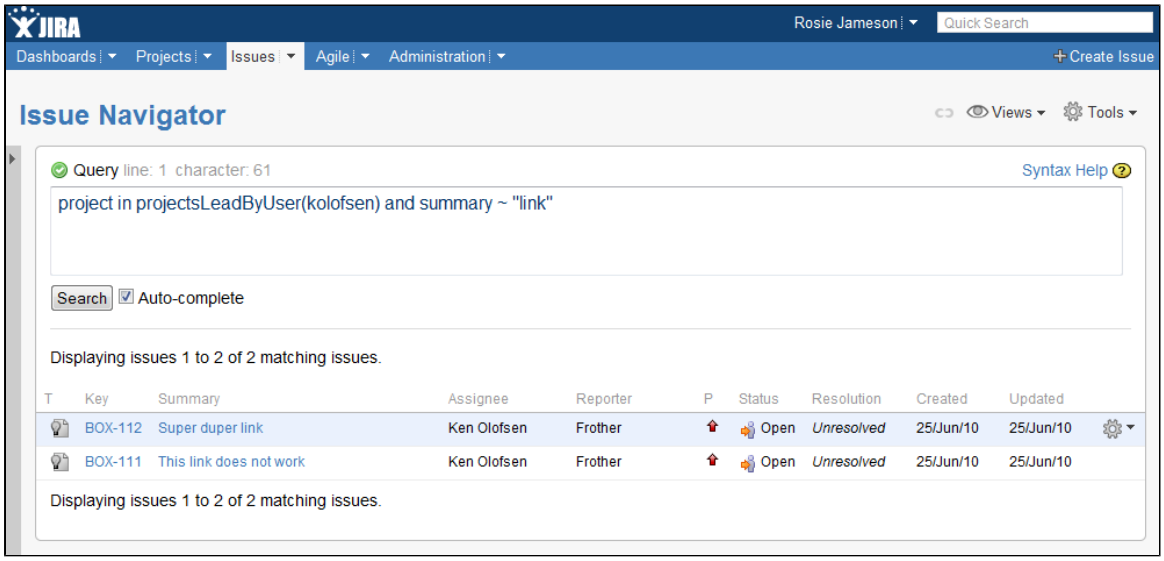


See Adding a User Avatar and Using Hover Profile for more information.

^Top

New JQL Functions

Additional JQL functions for projectsLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



See Advanced Searching for more information.

^Top

Other Enhancements and Fixes

- If you use Firefox or IE, you can now search JIRA issues from the convenience of your browser's search box. Just add your JIRA site as a search engine/provider, via the dropdown menu next to the browser's search box. This is because JIRA now supports the autodiscovery part of the OpenSearch standard.

^Top

Plus over 120 other fixes and improvements

The top 50 most popular issues resolved in JIRA 4.2 are listed below. Click here for the full list.

JIRA Issues (50 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-868		Resolve & Time spent		Resolved
JRA-1744		"Log work done" as a MANDATORY field when resolving issues		Resolved
JRA-4888		Can't edit original estimate		Resolved
JRA-7624		Add time tracking fields to the RemoteIssue object		Resolved
JRA-13512		Setting remaining time to 0 in a post function of the workflow causes the original estimate to be set to 0 as well (only when no time was logged by the assignee)		Resolved
JRA-1993		Resolving an issue should automatically put the estimated time remaining to 0		Resolved
JRA-9170		Enable Wiki Renderering for the Work Log description field		Resolved
JRA-13733		I should be able to change the remaining estimate without logging work done		Resolved
JRA-1423		Option to set Estimated Time Remaining to 0 when issue is closed		Resolved
JRA-12498		New functions for SOAP API		Resolved
JRA-11276		SOAP: Retrieving Parent- and Sub- Issues information		Resolved
JRA-5761		Session timeout should be handled more gracefully.		Resolved
JRA-19796		REST API for JIRA		Resolved
JRA-14076		DoubleConverter / NumberCFType is not as I18N as it could / should be - decimal separator does not respect user's locale		Resolved
JRA-8557		Add a addWatcher(RemoteUser) and / or addWatcher(RemoteUser[]) RPC / SOAP method		Resolved
JRA-22368		Linking Issues - Select multiple issues problem		Resolved

JRA-21189		View Issue screen custom tabs show fields from all custom tabs, on page load and when Activity tabs are clicked		Resolved
JRA-21439		Support Transparent Image for Project Avatar		Resolved
JRA-21105		IE6 over HTTPS displays popup about secure & unsecure content on all view issue pages		Resolved
JRA-20956		Issue View shows fields of second tab when first opened		Resolved
JRA-19557		Thumbnail of certain image attachments fail and cause ERROR in log		Resolved
JRA-15862		Thumbnail in JPEG breaks transparency used in PNG/GIF		Resolved
JRA-20995		Privilege escalation vulnerability when administrator access is compromised		Resolved
JRA-20562		JQL breaks issue security levels based on custom fields		Resolved
JRA-17759		CAPTCHA image broken when running in OpenJDK		Resolved
JRA-15978		Add functionality to JIRA XML-RPC and soap for downloading attachments		Resolved
JRA-21605		New UI makes issue key very hard to select for copy and paste		Resolved
JRA-7287		Soap: GetSubTaskParent(IssueKey)		Resolved
JRA-21244		Enable keyboard navigation of users in Assignee drop down		Resolved
JRA-21166		Can't select issue summary in issue view screen		Resolved
JRA-11682		Put an 'Add comment' link at the bottom of the page.		Resolved
JRA-15776		Get Standalone JIRA to detect the DBCP settings and RDBMS and add warnings if it is using MySQL without a validation query.		Resolved
JRA-20994		XSS Vulnerabilities in JIRA		Resolved
JRA-21841		XsrfVulnerabilityDetectionSQLInterceptor causes NPE on GreenHopper		Resolved
JRA-17760		JIRA does not compile under OpenJDK		Resolved
JRA-21876		LDAP delegating CROWD installations are broken by JIRA brute password protection code		Resolved
JRA-17395		Some gif images are not being sent with browser caching headers		Resolved
JRA-19947		List of attachments is not kept when updating an issue fails due to validators		Resolved
JRA-21159		PermGen memory not set, if JAVA_HOME is set incorrectly or missing		Resolved
JRA-21308		Comment form buttons aren't visible by scrolling when box fills up the browser window.		Resolved
JRA-22643		Bullets are different from preview		Resolved
JRA-22064		Attachment Sort order		Resolved

JRA-20810		Attachment sorting needed like issue sorting		Resolved
JRA-15079		Sent information is lost when data is submitted after session has expired		Resolved
JRA-21806		Feedback for combo boxes		Resolved
JRA-8082		SOAP should be able to find out what groups a user is a member of		Resolved
JRA-19691		Attach file should have replace or rename option and default comment		Resolved
JRA-24548		JIRA Create Issue hot-key interferes with Copy hot-key on the Mac		Resolved
JRA-22172		using JSESSIONID from /auth REST resource doesn't work		Resolved
JRA-21243		Unable to run from start menu after installation due to Service name on JIRA 4.0 and JIRA 4.1.1 has different SERVICE_NAME description		Resolved

^Top

JIRA 4.2 Upgrade Guide

On this page:

- Upgrading from JIRA 4.1 to 4.2
  - General Upgrade Instructions
  - Changes in `jira-application.properties`
  - Changes in `seraph-config.xml` and Crowd Integration
  - Non-bundled Plugins
  - Updated Toolkit Plugin for JIRA 4.2
  - Labels Plugin is Now in Core JIRA
  - Time Tracking Changes
  - Changes to Sub-Tasks
  - Changes to Attachment Creation
  - System Path Changes for Attachments, Indexes, Automated Backups and Services
  - New Security Option Available in `seraph-config.xml`
  - 'Contact Administrators' Link has been Removed
  - GreenHopper Versions 5.2 and Older are Not Compatible
  - Crowd Versions 2.0.6 and Older are Not Supported
    - Users of Fisheye/Crucible
  - Various Platforms are No Longer Supported
  - Internet Exporer 7 and 8 users must have "native XMLHTTP support" enabled
  - Other Known Issues
- Upgrading from JIRA 4.0 and Earlier

Upgrading from JIRA 4.1 to 4.2

General Upgrade Instructions

Please follow the instructions in the general JIRA upgrade guide (non-version specific), as well as the JIRA 4.2-specific instructions in the sections below. The general upgrade guide contains important tasks that are essential for getting your upgraded JIRA instance to work correctly (e.g. merging `jira-application.properties` customisations from the old instance to the upgraded instance).

Changes in `jira-application.properties`

If you are merging your old and new configuration files, as described in the Upgrade Guide, the following tables list the changes which have been made to the `jira-application.properties` file in JIRA 4.2.

The purpose of each *new* property is documented in the `jira-application.properties` file itself.

New properties in `jira-application.properties`
`jira.date.time.picker.use.iso8061 = false`
`jira.timetracking.estimates.legacy.behaviour=true`

`jira.timetracking.copy.comment.to.work.desc.on.transition=true`
`user.keyboard.shortcuts.disabled=false`
`jira.attachment.number.of.zip.entries=30`
`jira.attachment.do.not.expand.as.zip.extensions.list=docx, docm, dotx, ...`
`jira.ajax.autocomplete.labelsuggestion.limit = 20`
`jira.user.avatar.enabled=true`
`jira.browser.unsupported.warnings.disabled=false`

Properties removed from `jira-application.properties`
`jira.attachment.number` (See Changes to Attachments below for more information.)
`jira.paths.set.allowed` (See System Path Changes below for more information.)
`jira.paths.safe.backup.path` (See System Path Changes below for more information.)

Properties changed in `jira-application.properties`
`jira.avatar.megapixels=5` was changed to `jira.avatar.megapixels=10`

Changes in `seraph-config.xml` and Crowd Integration

When merging your old and new configuration files, as described in the Upgrade Guide, please take extra care with the `seraph-config.xml` file, since this file contains several new entries in JIRA 4.2.

If you simply copy your old `seraph-config.xml` to your new 4.2 installation, then:

- Your users will find that the Remember my login... functionality will not work.
- If you had Crowd integrated with JIRA prior to upgrading, your Crowd integration with JIRA will no longer work after the upgrade.

The following table lists the changes to the `seraph-config.xml` file in JIRA 4.2:

Elements in `seraph-config.xml` prior to JIRA 4.2	**Change in JIRA 4.2**
`<param-name>login.cookie.key</param-name>` `<param-value>seraph.os.cookie</param-value>`	`<param-value>seraph.os.cookie</param-` `<param-value>seraph.rememberme.cookie`
`<param-name>autologin.cookie.age</param-name>` `<param-value>31536000</param-value>`	`<param-value>31536000</param-value>` ha `<param-value>1209600</param-value>`

`<init-param>`
`<param-name>cookie.encoding</param-name>`
`....`
`<init-param>`
	Removed as this `<init-param>` entry is no long

`<authenticator class=`
`"com.atlassian.crowd.integration.seraph.JIRAAuthenticator"/>`
	Changed to
	`<authenticator class=`
	`"com.atlassian.crowd.integration.`

`<param-name>config.file</param-name>` `<param-value>seraph-paths.xml</param-value>`	`<param-value>/seraph-paths.xml</param` `<param-value>seraph-paths.xml</param-`

Non-bundled Plugins

If you use or develop a plugin that is not bundled with JIRA, then please read the Updating JIRA Plugins for JIRA 4.2 guide. This guide describes changes in JIRA 4.2 which may affect the compatibility of your plugin with JIRA 4.2.

Updated Toolkit Plugin for JIRA 4.2

If you use the Toolkit Plugin with JIRA, you will need to update it to at least version 0.15 for compatibility with JIRA 4.2.

Labels Plugin is Now in Core JIRA

The Labels plugin functionality is now part of JIRA core, so the Labels plugin should no longer be installed. If an earlier version of the Labels plugin is installed when upgrading to JIRA 4.2, JIRA will not start up. An appropriate error message wil be shown in the logs and UI.

All existing labels will continue to exist. As part of the 4.2 upgrade process, any data in a pre-existing "Labels" custom field (e.g. if you were using the Labels plugin) will be migrated to the new "Labels" system field.

Only labels custom-fields called "Labels" will be converted to the system field. All other label custom fields will remain as label custom-fields. If a particular label custom-field should be moved into the system field, it should be named "Labels" before the upgrade!

Time Tracking Changes

Prior to JIRA 4.2, the **Original Estimate** and **Remaining Estimate** fields could not be edited independently and their values would be synchronised after logging work.

From JIRA 4.2, time tracking is more flexible. The values in these fields are not tied to each other and they can be edited independently. Additionally, it is now possible to change the Original Estimate value *after* work has been logged.

Clean installations of JIRA 4.2 will automatically have access to this more flexible time tracking feature. However, if you are a JIRA customer *upgrading* to version 4.2, JIRA's time tracking feature will be set to **Legacy Mode** so that users can continue to operate JIRA's original work logging features as usual.

If you have upgraded to JIRA 4.2 and wish to use the newer and more flexible time tracking features, you will need to disable **Legacy Mode** in JIRA's Time Tracking settings. (You will need to deactivate Time Tracking before you can disable **Legacy Mode**.)

If you have upgraded to JIRA 4.2 and have disabled **Legacy Mode**, the **Log Work** fields *will not automatically be available* on the 'Resolve' and 'Close' transitions of JIRA's default workflow, nor will they be available on any other (custom) workflow transition.

To add these fields to a workflow transition, add the **Log Work** field to the appropriate screen used by that workflow transition. Refer to Adding Time Tracking capabilities to a screen on the Defining a Screen page for more information.

(The **Log Work** field is actually a group of time-tracking fields.)

Tip: If you use JIRA's default workflow, add the **Log Work** field to the **Resolve Issue Screen**.

Changes to Sub-Tasks

On new installations of JIRA 4.2 or later, Sub-Tasks are enabled by default. However, upon upgrading to JIRA 4.2, your Sub-Task configuration will remain unchanged. Therefore, if Sub-Tasks are disabled on your JIRA site before upgrading to JIRA 4.2, Sub-Tasks will still be disabled after the upgrade is completed.

Changes to Attachment Creation

Attachments are uploaded 'inline', that is when a user selects a file it is immediately uploaded to the server. A temporary attachment will be created on the server for this file. Once the form the file was uploaded in is submitted, the temporary attachment will be converted to a real attachment for the issue in question. Due to this improvement, limiting the number of file upload boxes via the `jira.attachment.number` property is no longer necessary and can safely be removed from your jira-application.properties file.

System Path Changes for Attachments, Indexes, Automated Backups and Services

For security reasons, from JIRA 4.2, you are no longer able to customise directories for storing the following types of JIRA content:

- attachments,
- search indexes,
- automated XML backups and

- other JIRA services

JIRA now stores these types of content in '**Default**' directories within the JIRA home directory.

- For new installations of JIRA 4.2, these types of content will only be stored within their default directories.
- If you upgrade to JIRA 4.2 from an earlier JIRA version that used custom directories to store these types of content, JIRA 4.2 will respect these custom directories. However, once you change to using the default directory for storing any of these types of content, you can no longer specify nor use a custom directory for that content type.

Note that you will need to copy your attachments to your JIRA Home Directory if you set your Attachments path to "default".

If you upgrade to JIRA 4.2 on a different machine (or different operating system) and previously:

- used custom directories to store attachments or search indexes, JIRA will warn you that it cannot create these custom directories if they do not exist on the new system.
  Hence, you can either:
  - Choose to use JIRA's default directories to store these types of content, or
  - Shutdown JIRA, recreate those custom directories (with permission for JIRA to write to them) and start the JIRA upgrade process again with the same XML backup.
    To identify these custom directories, JIRA will indicate them during the XML backup restore process.
- used custom storage directories for any JIRA service (such as an automated XML backup), JIRA will respect these custom directories and create them for you on the new system.

For more information about these changes, please refer to JRA-21232.

New Security Option Available in `seraph-config.xml`

Seraph can now be configured to invalidate your session upon login, which is a more secure configuration than before.

What this means is that the session you establish with JIRA **before** logging in is effectively destroyed and recreated with a new identity. This means that the session cookie value will be different after logging in. The implications of this are that you can be sure that even if you as a user have been unknowingly tricked into following a poison link to JIRA that forces you to use a session id that an attacker also already has access to, the act of logging in will free you from that session and you will not enable the attacker to gain access to your account.

In addition to the new identity, the new session will maintain, as best it can, the state of your interactions with JIRA. Your current project, current filter, etc, will remain after login. This is nice because sometimes you use JIRA without realising you are not authenticated, and logging in should disrupt your work as little as possible.

By default, seraph is configured to invalidate sessions in JIRA 4.2. This can be turned off in the seraph configuration file, typically called `seraph-config.xml`.

Here's a sample of the part of the config file that enables session invalidation:

```
<init-param>
    <param-name>invalidate.session.on.login</param-name>
    <param-value>true</param-value>
</init-param>
<init-param>
    <param-name>invalidate.session.exclude.list</param-name>
    <param-value>ASSESSIONID,dashboardPage</param-value>
</init-param>
```

In the example above you can see that in addition to the `invalidate.session.on.login` parameter which activates the feature, there is a second (optional) parameter which can hold a list of session keys which are to be excluded from the new session after login. By default, all session attributes are copied to the new session. If there are any to be excluded from this, they should be defined in a comma separated list for the `invalidate.session.exclude.list` parameter.

'Contact Administrators' Link has been Removed

Please note that the '**Contact Administrators**' link has been removed from the JIRA footer. If you have users who rely on this link, please publish a list of administrators elsewhere before upgrading to JIRA 4.2.

GreenHopper Versions 5.2 and Older are Not Compatible

If you use the GreenHopper plugin, please note that only version 5.3 is compatible with JIRA 4.2.

Crowd Versions 2.0.6 and Older are Not Supported

If you have integrated your JIRA instance with Atlassian Crowd, please upgrade to Crowd 2.0.7. Crowd versions up to and including 2.0.6 will not work with JIRA 4.2. See the Crowd 2.0.7 release notes.

Users of Fisheye/Crucible

Please note that there is a compatibility issue with the `crowd-integration-client-2.0.7.jar` and Fisheye/Crucible that is described

in this JIRA issue.

Various Platforms are No Longer Supported

As mentioned in our End of Support Announcements for JIRA page, from JIRA 4.2, we will no longer provide support for the following platforms with JIRA:

- Oracle WebLogic and IBM WebSphere applications servers, as mentioned in Deprecated Application Servers for JIRA announcement,
- Java Platform 5, as mentioned in the Deprecated Java Platforms for JIRA announcement and
- Internet Explorer 6 web browser, as mentioned in Deprecated Web Browsers for JIRA announcement.

Internet Exporer 7 and 8 users must have "native XMLHTTP support" enabled

There is a known bug that will make transitioning issues impossible from Internet Explorer 7 and 8 when "native XMLHTTP support" is disabled. See http://jira.atlassian.com/browse/JRA-22609 or the JIRA Knowledge Base for details.

Other Known Issues

Before you begin the upgrade, please check for known issues. Sometimes we find out about a problem with the latest version of JIRA after we have released the software. In such cases we publish information about the known issues in the JIRA Knowledge Base. Please check for known issues and follow the instructions to apply any necessary patches.

If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Upgrading from JIRA 4.0 and Earlier

In addition to the points listed above, please read the *Upgrade Guide* for every version you are skipping during the upgrade, particularly the JIRA 4.0 Upgrade Guide as JIRA 4.0 introduced significant licensing and technical changes. The complete list of Upgrade Guides is available here: Production Releases.

Updating JIRA Plugins for JIRA 4.2

On this page:

- Plugin Developer Notes
  - Custom fields that require JavaScript
  - Custom fields that directly implement `GroupSelectorField`
  - Custom Issue Operations need to work without an issue being present

Plugin Developer Notes

JIRA 4.2 introduces several changes that may break existing plugins which are not bundled with JIRA.

If you are using or have been involved in the development of such a plugin, it may need to be updated to work with JIRA 4.2. Please read through the information below to see if any of this content is relevant to your plugin.

If you are using a plugin developed by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.2.

- Custom fields that require JavaScript
- Custom fields that directly implement `GroupSelectorField`
- Custom Issue Operations need to work without an issue being present

This is not the complete list of changes for JIRA 4.2 — It only describes changes in JIRA 4.2 that will impact plugin developers.

Custom fields that require JavaScript

Use the following pattern to execute JavaScript when dialogs are loaded. The `dialogContentReady` event is fired after the dialog boxes are loaded. This pattern is mainly useful when adding behaviour to, or interacting with, custom fields/forms.

```
AJS.$(function () {

    function initMyCustomField(dialog) {
        AJS.$(dialog || document.body).each(function () {
            var $dialog = AJS.$(this);
            ...
        });
    }

    // Init the control on DOM ready
    initDatePicker();

    // Bind the init function so it runs when the dialog loads
    AJS.$(document).bind("dialogContentReady", function (e, dialog) {
        initDatePicker(dialog.get$popupContent());
    });

});
```

Custom fields that directly implement `GroupSelectorField`

To address JRA-20562, the `GroupSelectorField` marker interface (typically used for User and Group Picker-based custom fields), has been changed to an interface with the following method:

```
Query getQueryForGroup(final String fieldID, String groupName);
```

The intention of this method is to return a query that takes into account any case folding that has been done by the underlying custom field type. For instance, JIRA appends `_raw` to the field id before indexing `GroupSelectorField` objects. Hence, if you want similar behaviour in your custom field, you would define the following (as used within JIRA):

```
public Query getQueryForGroup(final String fieldID, String groupName)
    {
        return new TermQuery(new Term(fieldID+"_raw",groupName));
    }
```

Custom Issue Operations need to work without an issue being present

JIRA 4.2 permits the use of keyboard shortcuts for custom issue operations on the Issue Navigator. To do this, we render on an Issue Navigator page (but hide from the user) all issue operation links defined in the system, without a specific issue being available in the context for the issue operation. Custom issue operations must be able to gracefully handle this scenario, or they will be ignored from this list and plugin developers will not be able to attach keyboard shortcuts to their issue operations.

Specifically, issue operation web-items need to handle the case where the `JiraHelper` does not provide an issue in its context!

Furthermore, issue operation URLs need to follow a specific format in order for keyboard shortcut actions to be attached to them via JavaScript. Here is an example of a valid issue operation URL format:

```
<a id="assign-issue" class="issueaction-assign-issue"
href="/jira/secure/AssignIssue!default.jspa?id={0}&returnUrl=/secure/IssueNavigator.jspa"></a>
```

The corresponding issue operation web-item definition looks as follows:

```
<web-item key="assign-issue" name="Assign Issue Link" section="operations-top-level"
i18n-name-key="webfragments.view.issue.opsbar.operations.top.assign" weight="5">
    <label key="common.words.assign"/>
    <tooltip key="admin.issue.operations.plugin.assign.issue.name"/>
    <styleClass>issueaction-assign-issue</styleClass>
    <link linkId="assign-issue">
        /secure/AssignIssue!default.jspa?id=$issueId
    </link>
    <condition
class="com.atlassian.jira.plugin.webfragment.conditions.HasIssuePermissionCondition">
        <param name="permission">assign</param>
    </condition>
    <condition
class="com.atlassian.jira.plugin.webfragment.conditions.IsIssueEditableCondition"/>
    <condition class="com.atlassian.jira.plugin.webfragment.conditions.IsFieldHiddenCondition"
invert="true">
        <param name="field">assignee</param>
    </condition>
</web-item>
```

The important part of the Issue Navigator URL is the parameter `id={0}`. The Issue Navigator will automatically substitute the `$issueId` variable with {0}. This will then be substituted by the JavaScript triggered by a keyboard shortcut with the currently selected issueId on the Issue Navigator.

On the issue view page, which only deals with a single issue, the `$issueId` will simply be substituted with the current issue id. The webwork action or servlet (in this case `AssignIssue`) needs to use the id URL parameter to resolve the issue.

JIRA 4.2.4 Release Notes

8 February 2011

The Atlassian JIRA team announces the release of **JIRA 4.2.4**. This point release fixes an issue with the 'Filter Results' gadget. It also contains version 1.7.1 of the JIRA Importers Plugin.

JIRA 4.2.4 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.2 yet?
Take a look at all the new features in the JIRA 4.2 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.2.4 Upgrade Guide.

JIRA 4.2.4 Upgrade Guide

Upgrading from JIRA 4.2.x to 4.2.4

Please follow the JIRA general upgrade instructions, plus note the following:

Changes to filenames

When you download JIRA, please note that the filename now includes the build number, e.g. "b588" in the filename "atlassian-jira-enterprise-4.2.1-b588-windows-installer.exe". This has no effect on the JIRA directory structure or on the upgrade process.

Please note that the inclusion of the build number has also been applied to the name of JIRA artifacts deployed to Maven and to the versions specified in the JIRA POMs.

Changes to the "JIRA Users" Global Permission

Please note that it is no longer possible for groups that have the "JIRA System Administrators" global permission to be also granted the "JIRA Users" global permission (see Managing Global Permissions). This is because newly-created users are automatically added to groups that have the "JIRA Users" permission, and it is not good security practice to automatically grant the "JIRA System Administrators" global permission to new users.

Upgrading from JIRA 4.1.x and earlier

In addition to the above, please read the JIRA 4.2 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.2.3 Release Notes

31 January 2011

The Atlassian JIRA team announces the release of **JIRA 4.2.3**. This point release contains several updates and fixes, plus a new release of the JIRA Importers Plugin, providing improved support for Mantis and CSV imports. Please see the documentation: Importing Data from Mantis and Importing Data from CSV.

JIRA 4.2.3 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.2 yet?
Take a look at all the new features in the JIRA 4.2 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.2.3 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.2.3 includes the following updates and bug fixes:

JIRA Issues (28 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-23327	Labels upgrade task assumes screen has a tab; may not be true		Resolved
	JRA-23390	Order of E-mail notifications (sent from queue)		Resolved
	JRA-23357	Javascript error in IE7 when Dashboard have Projects gadgets more than 420+ projects		Resolved
	JRA-23052	Sidebar clips tab names in Internet Explorer 6 and up on using other languages		Resolved
	JRA-21648	Issue filters do not follow issue security scheme correctly in some cases		Resolved
	JRA-23489	Calendar javascript widget should warn the user if there is no localised resources for the user's current locale		Resolved
	JRA-23257	Content-Length header on DELETE is incorrect		Resolved
	JRA-23183	Create Issue Components dropdown with many components is outside of viewable area and page scrolling by browser's (IE) vertical scrollbar causes dropdown to disappear.		Resolved
	JRA-23086	DefaultComponentClassManager does not find the plugin that can autowire a given class name correctly		Resolved
	JRA-23058	User is unable to scroll field values in "column to display" while editing gadget configuration in FF 3.6		Resolved
	JRA-22720	Attachments JSP error on JAC		Resolved
	JRA-22109	Issue Navigator thinks my Simple search is "too complex" for the simple page when I include a version		Resolved
	JRA-21238	Activity Stream broke, faultString: java.io.IOException: java.lang.IllegalArgumentException: The char '0x1' after		Resolved
	JRA-23387	Camel cased filters sometimes don't show when using filter/project autocomplete in Gadgets		Resolved
	JRA-23371	Javascript error shows 'Calendar' is undefined		Resolved
	JRA-23369	Comment security dropdown text incorrectly shortened		Resolved

	JRA-23321	Scrolling in the transition window seems to scroll in the background window in some occasions		Resolved
	JRA-23289	JIRA HTML email template refers to Enterprise Edition and the JIRA link is outdated		Resolved
	JRA-23229	NPE on upload of corupted avatar image		Resolved
	JRA-23152	The Comment security level dropdown does not align correctly under firefox when it is opened over an image.		Resolved
	JRA-23136	The browse project page's javascript does not have error handler for ajax calls which leaves the page blank on error.		Resolved
	JRA-23135	The browse project tabs can fail when their content is large enough.		Resolved
	JRA-23120	Label names not resolved on Single Level Group By Report		Closed
	JRA-23108	Attachment description is displayed twice		Closed
	JRA-22707	ID clash between ops bar Link Issue button and the Link Issue form in the dialog		Resolved
	JRA-21143	Null project descriptions result in an NPE when loading the project gadget		Resolved
	JRA-19236	User Picker auto-complete suggestions can be incorrect		Resolved
	JRA-16102	Accesskey + S for submitting sub tasks do not work		Resolved

JIRA 4.2.3 Upgrade Guide

Upgrading from JIRA 4.2.x to 4.2.3

Please follow the JIRA general upgrade instructions, plus note the following:

Changes to filenames

When you download JIRA, please note that the filename now includes the build number, e.g. "b588" in the filename "atlassian-jira-enterprise-4.2.1-b588-windows-installer.exe". This has no effect on the JIRA directory structure or on the upgrade process.

Please note that the inclusion of the build number has also been applied to the name of JIRA artifacts deployed to Maven and to the versions specified in the JIRA POMs.

Changes to the "JIRA Users" Global Permission

Please note that it is no longer possible for groups that have the "JIRA System Administrators" global permission to be also granted the "JIRA Users" global permission (see Managing Global Permissions). This is because newly-created users are automatically added to groups that have the "JIRA Users" permission, and it is not good security practice to automatically grant the "JIRA System Administrators" global permission to new users.

Upgrading from JIRA 4.1.x and earlier

In addition to the above, please read the JIRA 4.2 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.2.2 Release Notes

4 January 2011

The Atlassian JIRA team announces the release of **JIRA 4.2.2**. This point release contains several updates and fixes, plus a Beta release of the JIRA Importers Plugin, providing improved support for Bugzilla. Please see the documentation: Importing Data from Bugzilla.

This point release is a highly recommended upgrade as it contains important fixes to security vulnerabilities — please refer to the JIRA Security Advisory 2011-02-21 for details.

JIRA 4.2.2 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.2 yet?
Take a look at all the new features in the JIRA 4.2 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.2.2 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.2.2 includes the following updates and bug fixes:

JIRA Issues (24 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-22448	Comment Query Performance: Please Change CommentClauseQueryFactory.generateIssueIdFromCommentQuery from for loop to HitCollector		Resolved
	JRA-23196	Renaming Priority, Resolution, Status, Issue Type does not get reflected until a system restart		Resolved
	JRA-24146	After workflow scheme migration, issue moved to a newly added status are not readable		Resolved
	JRA-23265	AJP port is a bad default in JIRA 4.2		Resolved
	JRA-22618	Excel Export adds the issue key to the Project column & an extra blank row between records		Resolved
	JRA-23188	Basic auth authentication does not allow files to be attached in 4.2		Resolved
	JRA-23124	Bug in seraph breaks custom authenticators		Resolved
	JRA-23024	JIRA data anonymiser doesn't anonymise non-English characters		Resolved
	JRA-22673	Non-secure content warning in IE8 on the Dashboards screen caused by the wiki renderer		Resolved
	JRA-22623	Watched Issues gadget gives error when profiling is enabled.		Resolved
	JRA-22207	Add warning to Shared Filter explaining consequence of 'everyone'		Resolved
	JRA-21670	Upgrading to Jira 4.1.2 breaks PrepareAction functionality in plugins		Resolved
	JRA-21582	SimpleSearch turns Complex when searching on Fix Version options		Resolved
	JRA-19918	Cannot select some filters from filters/project picker in gadgets		Resolved
	JRA-23405	Issue keys are not hyperlinked when exporting to Excel		Resolved
	JRA-23268	Wrong link after attaching screenshot		Resolved
	JRA-23167	Date picker format incorrect when using Dutch(Belgium) language		Resolved
	JRA-23154	Forget Password link appear after enable"External password management"		Resolved
	JRA-23140	JiRA Calander - years listed alternative		Resolved
	JRA-23132	Required field asterisks and error messages not aligned correctly for some fields		Resolved
	JRA-23085	Increase the web session timeout from 60 minutes to 300 minutes		Resolved

	JRA-22885	The 'Date Started' field is compulsory on the Log Work field but there is no compulsory '*'		Resolved
	JRA-22823	Permalink icon on issue nav has a busted "outline focus" thing		Resolved
	JRA-23126	Typo in DEBUG log output		Resolved

JIRA 4.2.2 Upgrade Guide

Upgrading from JIRA 4.2 or 4.2.1 to 4.2.2

Please follow the JIRA general upgrade instructions, plus note the following:

Changes to filenames

When you download JIRA, please note that the filename now includes the build number, e.g. "b588" in the filename "atlassian-jira-enterprise-4.2.1-b588-windows-installer.exe". This has no effect on the JIRA directory structure or on the upgrade process.

Please note that the inclusion of the build number has also been applied to the name of JIRA artifacts deployed to Maven and to the versions specified in the JIRA POMs.

Changes to the "JIRA Users" Global Permission

Please note that it is no longer possible for groups that have the "JIRA System Administrators" global permission to be also granted the "JIRA Users" global permission (see Managing Global Permissions). This is because newly-created users are automatically added to groups that have the "JIRA Users" permission, and it is not good security practice to automatically grant the "JIRA System Administrators" global permission to new users.

Upgrading from JIRA 4.1.x and earlier

In addition to the above, please read the JIRA 4.2 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.2.1 Release Notes

6 December 2010

The Atlassian JIRA team announces the release of **JIRA 4.2.1**. This point release contains several updates and fixes, including the issue of remembering which sections of the 'View Issue' screen are collapsed. This point release is a **highly recommended upgrade** as it contains important fixes to security vulnerabilities — please refer to the JIRA Security Advisory 2010-12-06 for details.

JIRA 4.2.1 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.2 yet?
Take a look at all the new features in the JIRA 4.2 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.2.1 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.2.1 includes the following updates and bug fixes:

JIRA Issues (21 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-22554	Sending mails on a separate thread which can be timed out to avoid the mail queue from getting clogged		Resolved
	JRA-19998	Upgrade/Backup documentation doesn't mention copying avatars		Closed
	JRA-22980	Jasper exception when trying to add a zip containing a file with an ' (apostrophe) character in its name		Resolved
	JRA-22947	Hitting enter in autocomplete field doesn't work after typing component name with a space in it		Resolved

	JRA-22927	Searching and selection on decimal value is broken for fields like affected version/fix version in JQL search		Resolved
	JRA-22917	Several dialogs do not have correct focus when errors occur under IE		Resolved
	JRA-22750	"Viewable by" dropdown position in dialog is not retaining its position and also srolling up and down moves the view issue page in background		Resolved
	JRA-22674	Duplicate ids in Issue comments		Resolved
	JRA-22966	labels dialog has second redunant text field above the labels field		Resolved
	JRA-22829	Increase throbber timeout for frother control		Resolved
	JRA-22609	JIRA 4.2 hangs on Workflow transitions when Ajax is disabled in IE7/8		Resolved
	JRA-22572	The loading of the field layout cache is not properly synchronized, leading to major performance issues for large customers with many custom fields.		Resolved
	JRA-22505	Sequenced keyboard shortcuts don't 'mute' other shortcuts assigned to the same key		Resolved
	JRA-16851	Mail queue hangs after trackback problems		Resolved
	JRA-23217	JIRA 4.1.2 "Created vs Resolved Chart" gadget failed to render when adding it as external gadget to another instance		Resolved
	JRA-23216	Adding "Heat Map" gadget to another instance not working for JIRA 4.1.2		Resolved
	JRA-23047	Using the word "Footer" in "Components" makes UI break		Resolved
	JRA-22859	Screenshot attach always takes you to manage attachments.		Resolved
	JRA-22845	FC: Slow connections -> throbbers?		Resolved
	JRA-22320	Favicon's not included in frontpage.jsp		Resolved
	JRA-22628	Missing full stop on "Login Required" page		Resolved

JIRA 4.2.1 Upgrade Guide

Upgrading from JIRA 4.2 to 4.2.1

Please follow the JIRA general upgrade instructions.

When you download JIRA, please note that the filename now includes the build number, e.g. "b588" in the filename "atlassian-jira-enterprise-4.2.1-b588-windows-installer.exe". This has no effect on the JIRA directory structure or on the upgrade process.

Please note that the inclusion of the build number has also been applied to the name of JIRA artifacts deployed to Maven and to the versions specified in the JIRA POMs.

Upgrading from JIRA 4.1.x and earlier

In addition to the above, please read the JIRA 4.2 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.1 Release Notes

April 7, 2010

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

This release makes your JIRA experience easier and more convenient than ever. The issue UI has been redesigned for a simpler, friendlier experience, and keyboard shortcuts have been streamlined. Issues can now be actioned directly from your dashboard via a handy dropdown

in the gadgets. Each issue's attachments are now displayed in an image gallery, and can all be downloaded to a single ZIP file with just one click.

The installation process has been improved by the inclusion of industry standard database drivers and a GUI database configuration tool, and JIRA Standalone is now being shipped with Tomcat 6.0.20.

JIRA 4.1 also addresses some other enhancements and fixes, notably a fix to the 'Malformed security token' error appearing in gadgets on the JIRA 4.0 dashboard.

> **JIRA Plugin Developers:**
>
> If you develop JIRA plugins, please refer to our guide on updating JIRA plugins for JIRA 4.1, for details on keeping your plugins compatible with JIRA 4.1.

Upgrading to JIRA 4.1 is free for all customers with active JIRA software maintenance as of April 7, 2010.

Highlights of JIRA 4.1:

- New-Look Issues
- Improved Global Keyboard Shortcuts
- 'Actions' Dropdown in Gadgets
- 'Bulk Move' Components and Versions
- New-look User Profile, 'Manage Dashboards' and 'Manage Filters'
- New JQL Functions 'lastLogin' and 'currentLogin', and Fields 'Voter' and 'Watcher'
- List of Logged-In Users
- List of Upgrade History
- Time Tracking now accepts Fractions
- Image Gallery
- Download Attachments as a ZIP
- Customisable Email Subject
- 'Heat Map' Gadget
- Database Configuration Tool
- XSRF protection
- Database Drivers Included
- Other Enhancements and Fixes
- Plus over 80 other fixes and improvements

Thank you for your feedback:

More than 33 new feature requests implemented

More than 658 votes fulfilled

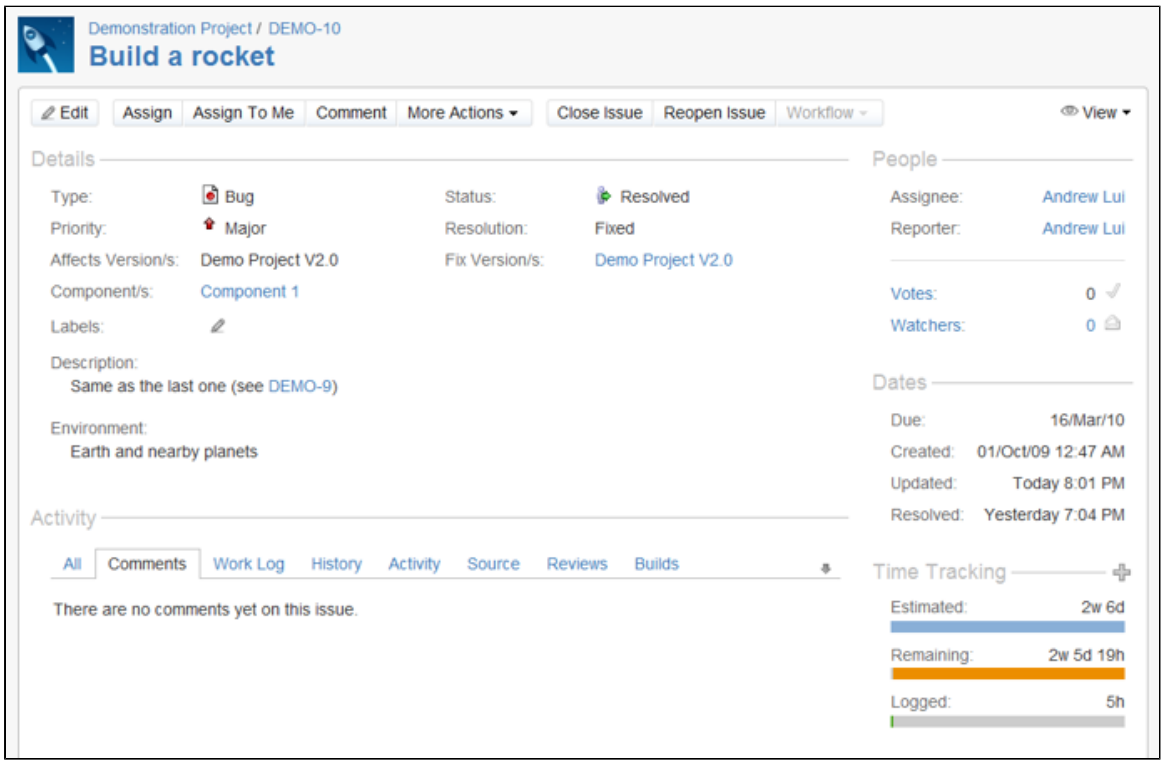
Your votes and issues help us keep improving our products, and are much appreciated.

> **Upgrading to JIRA 4.1**
>
> JIRA 4.1 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 4.1 Upgrade Guide.

Highlights of JIRA 4.1

New-Look Issues

The 'View Issue' and 'Edit Issue' screens have been simplified and modernised:



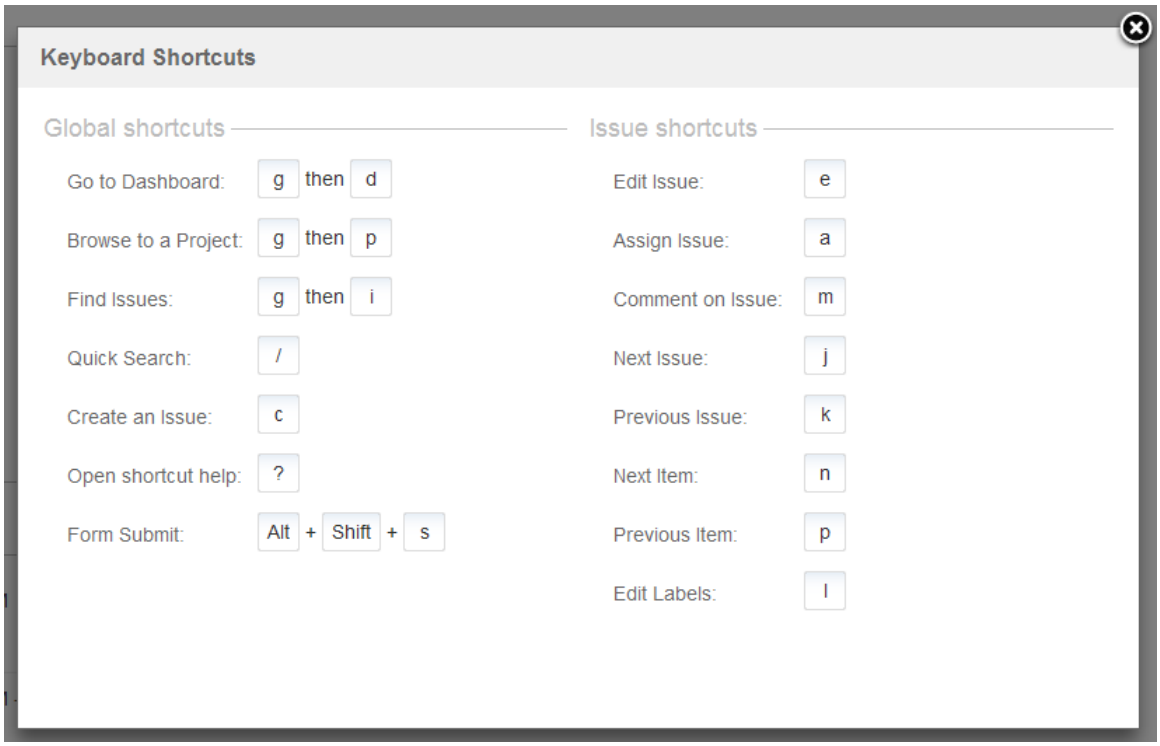
See the documentation for details.

^Top

Improved Global Keyboard Shortcuts

JIRA now has improved global shortcuts to provide quick navigation around JIRA. You can now go directly to the dashboard from anywhere else in JIRA, simply by typing 'g' then 'd'. Type 'g' then 'p' to jump to the browse project page, or type 'c' to start creating an issue. You can find what other keyboard shortcuts are available by typing '?', to bring up the Keyboard Shortcuts dialog box.

Screenshot: Keyboard Shortcuts Dialog Box



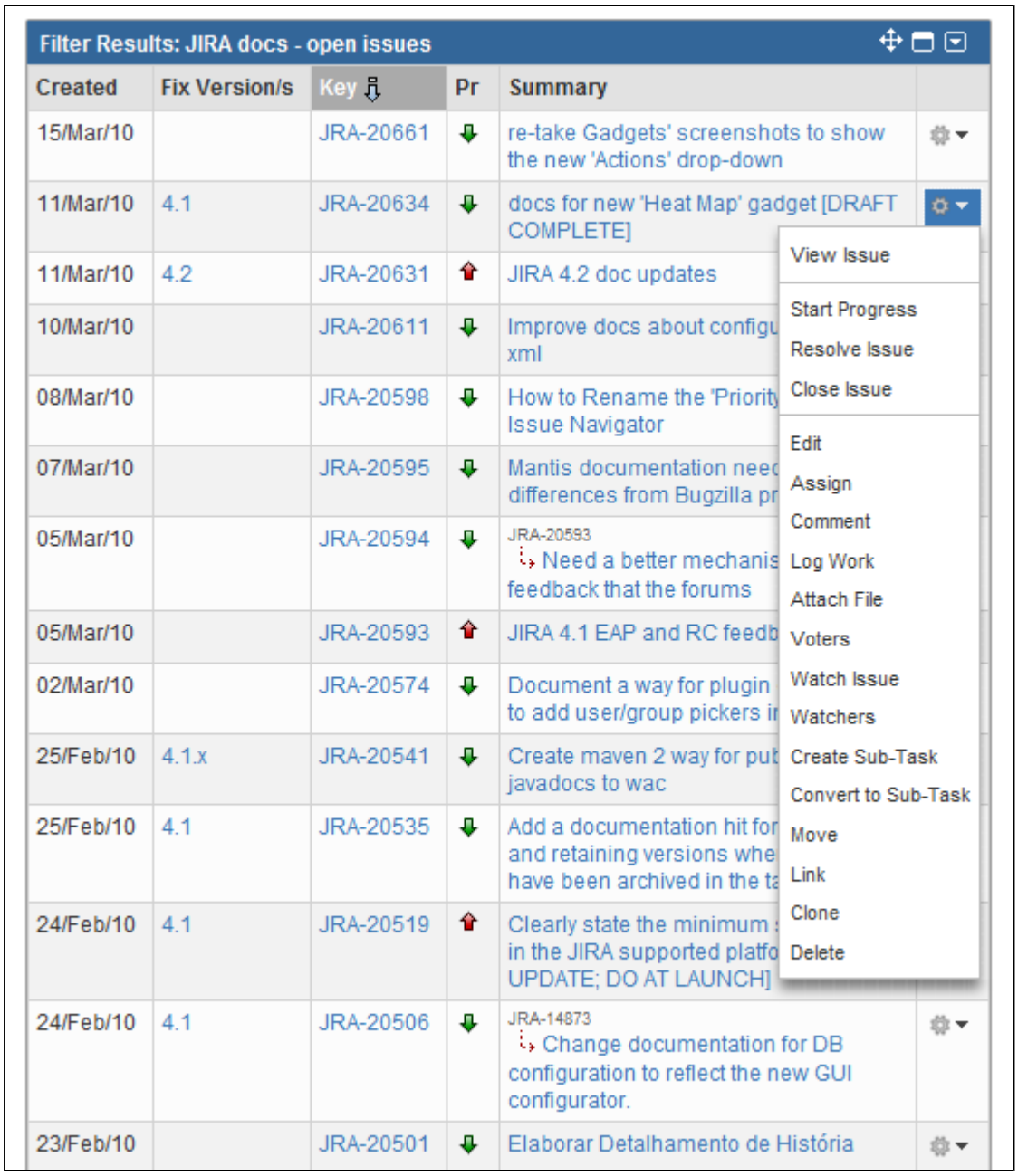
See the documentation for full details.

> **JIRA Plugin Developers:**
>
> This feature is implemented as a plugin, which will facilitate the addition of more keyboard shortcuts in future JIRA versions and iterations.

^Top

'Actions' Dropdown in Gadgets

You can now action issues directly from your JIRA dashboard:



^Top

'Bulk Move' Components and Versions

When performing a 'Bulk Move' of issues from one project to another, JIRA now allows mapping of old project values (for Components, Versions) to new values, using multiple edit controls and a simple name matching strategy to preselect the smart choice.

See the documentation for details.

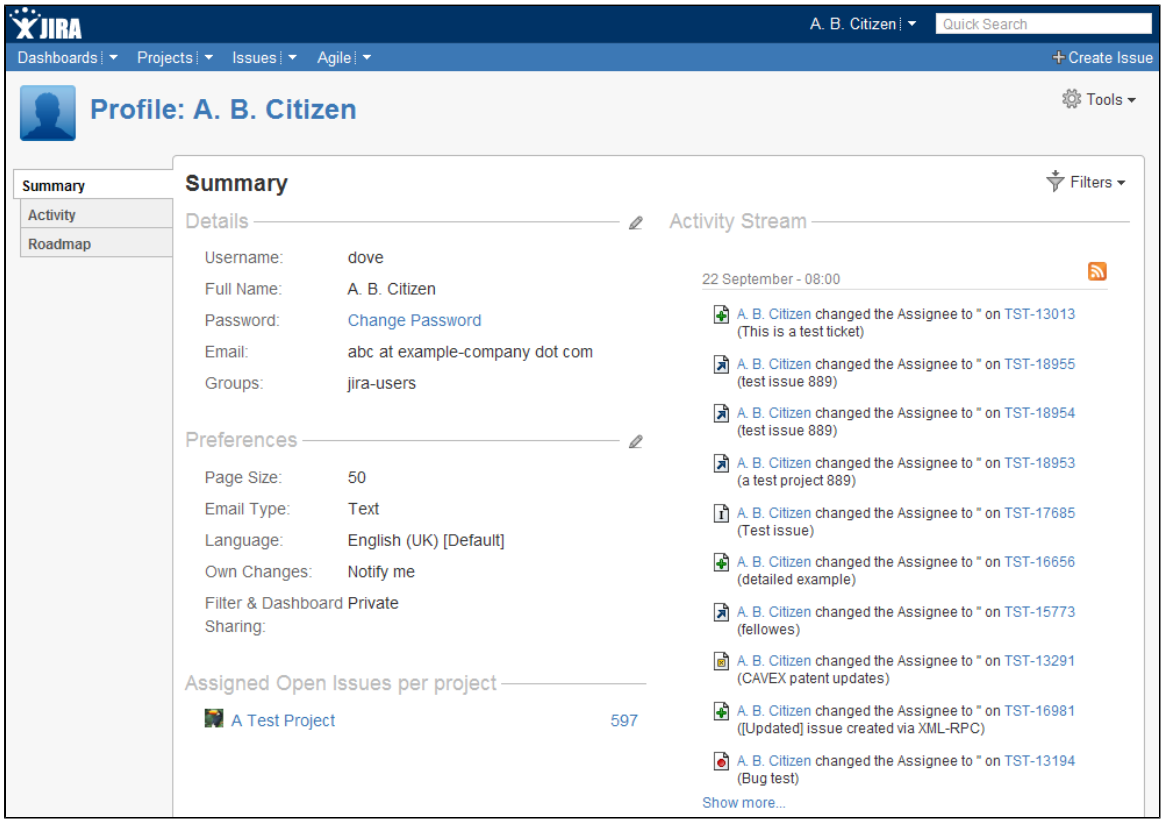
^Top

New-look User Profile, 'Manage Dashboards' and 'Manage Filters'

User Profile

JIRA User Profile screens have been redesigned to show you more details about your personal information and preferences, and to make it easier to access other useful features associated with your account.

Screenshot: The New-Look User Profile Summary Screen

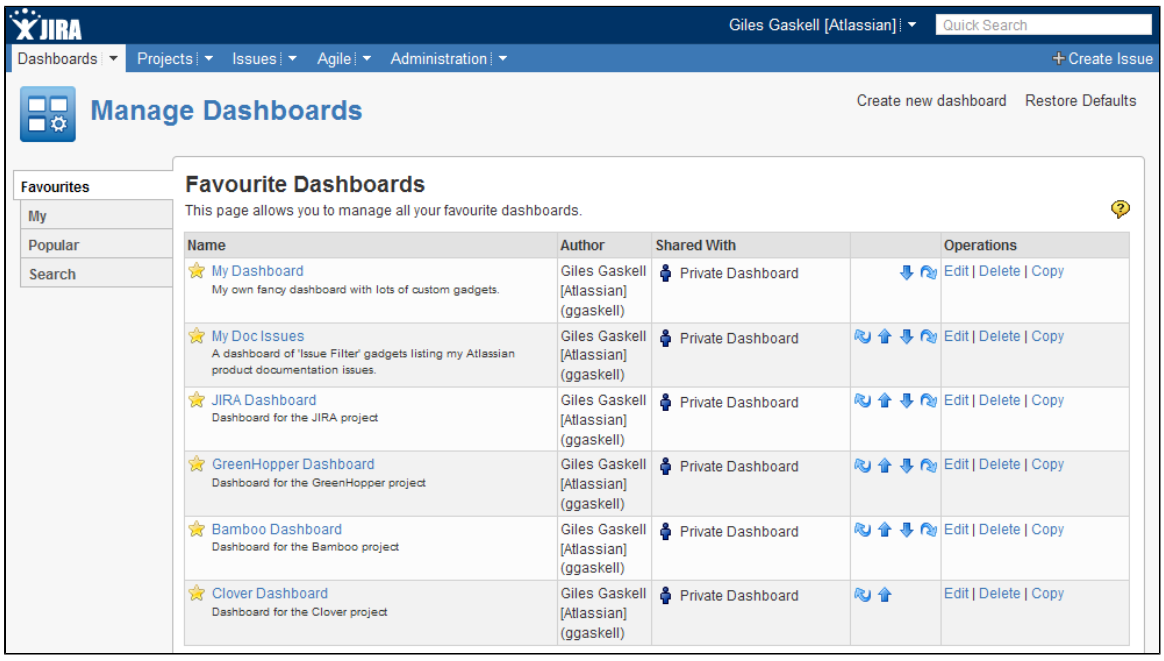


Please see the documentation for more details.

Manage Dashboards

The interface for the Manage Dashboards screens have been streamlined to blend in with the rest of the JIRA 4.1.

Screenshot: The New-Look Manage Dashboards Screen

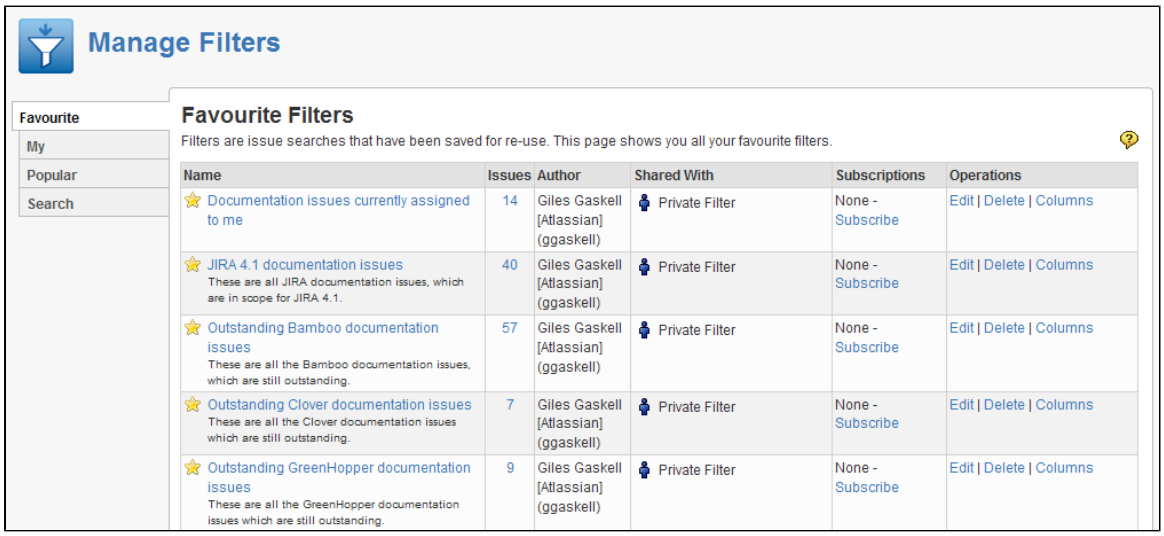


Please see the documentation for more details.

Manage Filters

Like the Managed Dashboards screens, the interface of the Manage Filters screens have also been streamlined to blend in with the rest of the product.

Screenshot: The New-Look Manage Filters Screen



Please see the documentation for more details.

^Top

New JQL Functions 'lastLogin' and 'currentLogin', and Fields 'Voter' and 'Watcher'

Now you can do JQL searches based on your current session, e.g. find all issues that have been created since you last logged in to JIRA:

```
created > lastLogin()
```

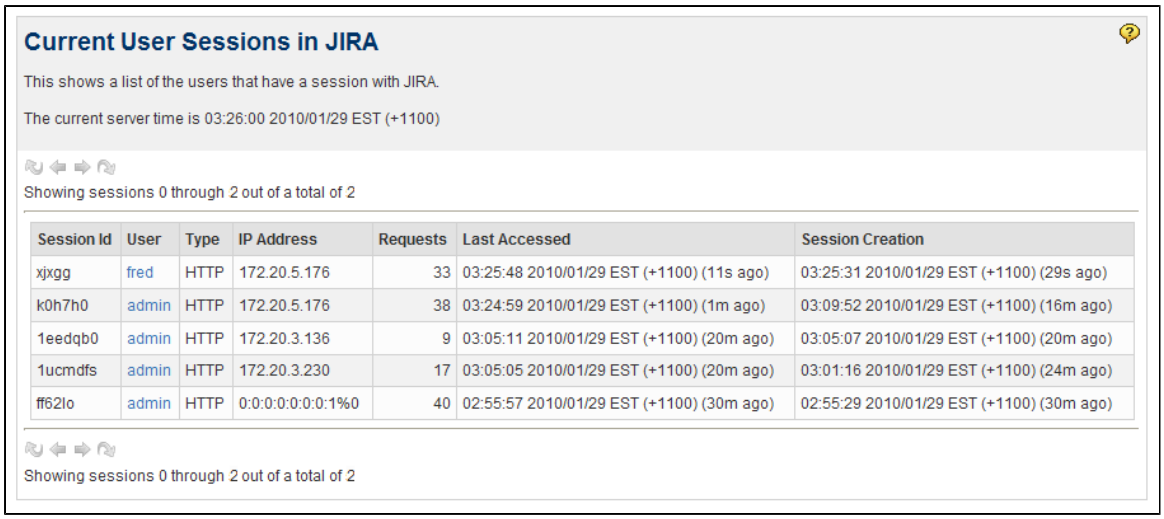
You can also search for issues that you (or anyone else) are watching or have voted for:

```
voter in membersOf("jira-developers")
```

See the documentation for details.

List of Logged-In Users

Need to see who's currently using JIRA? Now you can take a look:



Current User Sessions in JIRA

This shows a list of the users that have a session with JIRA.

The current server time is 03:26:00 2010/01/29 EST (+1100)

Showing sessions 0 through 2 out of a total of 2

Session Id	User	Type	IP Address	Requests	Last Accessed	Session Creation
xjxgg	fred	HTTP	172.20.5.176	33	03:25:48 2010/01/29 EST (+1100) (11s ago)	03:25:31 2010/01/29 EST (+1100) (29s ago)
k0h7h0	admin	HTTP	172.20.5.176	38	03:24:59 2010/01/29 EST (+1100) (1m ago)	03:09:52 2010/01/29 EST (+1100) (16m ago)
1eedqb0	admin	HTTP	172.20.3.136	9	03:05:11 2010/01/29 EST (+1100) (20m ago)	03:05:07 2010/01/29 EST (+1100) (20m ago)
1ucmdfs	admin	HTTP	172.20.3.230	17	03:05:05 2010/01/29 EST (+1100) (20m ago)	03:01:16 2010/01/29 EST (+1100) (24m ago)
ff62lo	admin	HTTP	0:0:0:0:0:0:1%0	40	02:55:57 2010/01/29 EST (+1100) (30m ago)	02:55:29 2010/01/29 EST (+1100) (30m ago)

Showing sessions 0 through 2 out of a total of 2

^Top

List of Upgrade History

Just for the record (and to assist with troubleshooting, etc), the System Info page now shows you a list of the upgrades that have been performed on your JIRA system.

^Top

Time Tracking now accepts Fractions

Ever wanted to enter `1.5 hours`, instead of `90 minutes`? Now you can.

^Top

Image Gallery

Each issue's attached images are now displayed for your convenience:

See the documentation for details.

^Top

Download Attachments as a ZIP

Another handy time-saver — simply click to download all of an issue's attachments as a single ZIP file:

See the documentation for details.

^Top

Customisable Email Subject

We are very pleased to announce that this much-requested feature is now a reality: you can now customise the subject of your JIRA-generated emails. See the documentation on Customising Email Content for details.

^Top

'Heat Map' Gadget

Brand new gadget to show a heatmap of a key statistic in a bunch of issues:

For more details, please see Adding the Heat Map Gadget.

^Top

Database Configuration Tool

JIRA 4.1 ships with a new GUI application that that will help you set up and test your database connection.

For more details, please see the documentation on Connecting JIRA to a Database.

^Top

XSRF protection

JIRA now implements a 'form token checking' mechanism. This provides JIRA with the ability to validate the origin and intent of key browser requests, thus adding an additional level of security against cross-site request forgery (XSRF).

For details, please see the documentation on Form Token Handling.

^Top

Database Drivers Included

To save you time when installing or upgrading JIRA, we are now shipping database drivers for MySQL, Postgres and MSSQL. Oracle drivers will be included soon.

^Top

Other Enhancements and Fixes

- The 'Malformed security token' error in JIRA dashboard gadgets was fixed – If a user logged in to JIRA and left the JIRA dashboard open for a period of time, gadgets on the Dashboard would generate a 'Malformed security token' error.

^Top

Plus over 80 other fixes and improvements

The top 50 most popular issues resolved in JIRA 4.1 are listed below. Click here for the full list.

JIRA Issues (50 issues)

Key	Type	Summary	Priority	Status
JRA-3609		Customize mail subject line		Resolved
JRA-19248		Dashboard gadgets fail with HTTP 401 malformed security token, when left unattended for some time		Resolved
JRA-1655		Ability to see users currently logged in		Resolved
JRA-8248		Retain values checkbox for Bulk Move should preserve versions or components based on version and components names		Resolved
JRA-2176		Time Tracking Config - Working hours with fractions (ie 7h 30m)		Resolved
JRA-7711		Allow code and noformat sections, and long text in general, in Wiki or Plain textfields to scroll horizontally		Resolved
JRA-5789		Transition order should be sortable for each Step Name		Resolved
JRA-6175		Passwords sent as clear text in email		Resolved
JRA-6514		Search by user name in admin section		Resolved
JRA-865		Display new items since I last logged in		Resolved
JRA-10611		"My Watches" portlet shows resolved issues.		Resolved
JRA-20050		Activity stream in Norwegian fails to load javascript resources		Resolved
JRA-12321		Ability to save multiple attachments (or all of them) at one time.		Resolved
JRA-15277		Soap API Should Support Trusted Applications		Resolved
JRA-13142		Cannot add a custom text renderer plugin		Resolved
JRA-19617		IE 6.0 compatibility: Activity Stream Gadgets come up with JS errors		Resolved
JRA-12380		Implement user lockout mechanism to stop bruteforce login attacks		Resolved

JRA-19853	Investigate and fix duplicate version numbers found during Package Scanning.	↓	Resolved
JRA-12943	Log work interface should accept fractional durations		Resolved
JRA-19995	jiraform is logging harmless but incessant log messages	↓	Resolved
JRA-15189	noformat tag breaks	↓	Resolved
JRA-18436	votedIssues() & watchedIssues() - ability to take params to support users		Resolved
JRA-6715	Allow filtering by number of votes		Resolved
JRA-19530	Activity Stream throws NFE when parsing a changeitem for a custom field with the name "status"	↓	Resolved
JRA-20144	Customization of notification email subject		Resolved
JRA-19822	2d filter Stats gadget takes along time to scroll when there is a width overflow	↓	Resolved
JRA-12662	It's a permAlink, dummy	↓	Resolved
JRA-13569	CAPTCHA should be possible to enable on user login page		Resolved
JRA-11144	Don't send passwords in emails when External password/user management enabled		Resolved
JRA-14306	Sorting Select List For Permission List		Resolved
JRA-13819	Add "Watch it" and "Vote For it" toggles next to totals		Resolved
JRA-16918	Lucene SegmentReader synchronization issues		Resolved
JRA-20429	There is a noticible delay when Left or Right Clicking on a Dashboard with ~30 gadgets.	↑	Resolved
JRA-20428	Left or Right Clicking on a Dashboard with ~30 gadgets causes Firefox to issue slow script warning.	↑	Resolved
JRA-20621	Quick Links not showing anything in JIRA 4.0.2 for Firefox and Safari but IE 6, 7, & 8 are working fine	↓	Resolved
JRA-20592	Add ability to log time worked using decimals		Closed
JRA-3627	Display changed items since I last logged in		Resolved
JRA-14540	setenv.sh in standalone should be tweaked to have more pre-configured options		Resolved
JRA-15605	The use of existing CAPTCHA after a certain number of unsuccessful login attempts		Resolved
JRA-5169	Avoid generating useless SQL rollback and commit queries		Resolved
JRA-14173	Display upgrade history in System Info		Resolved
JRA-20705	Provide javadoc in the maven repository please		Closed
JRA-17239	Improve the help text on the Announcment Banner screen		Resolved
JRA-19882	Strike through behaviour to be explain in the documentation		Closed

JRA-20678	Searching for a user in JQL by Fullname or Email address is very inefficient, and can lead to excessive memory usage.		Resolved
JRA-13873	Convert issue to subtask operation isn't sorted correctly on operations menu		Resolved
JRA-19515	watchedIssues() JQL function is very slow		Resolved
JRA-19814	Watchers can be added to a project without having rights in that project		Resolved
JRA-14494	NullPointerException thrown when clicking the "Start" watching link in the "Manage Watch List" panel and not logged in		Resolved
JRA-13755	Translation of time tracking estimates not working		Resolved

^Top

JIRA 4.1 Upgrade Guide

On this page:

- Upgrading from JIRA 4.0 to 4.1
  - General Upgrade Instructions
  - New Location of JIRA Log Files
  - Changes to Plugins
  - Form Token Handling
  - Customers with Crowd Integration
- Upgrading from JIRA 3.13 and earlier

Upgrading from JIRA 4.0 to 4.1

General Upgrade Instructions

Please ensure that you follow the instructions in the general JIRA upgrade guide (non-version specific), as well as the JIRA 4.1 specific instructions in the sections below. The general upgrade guide contains important tasks that are essential for getting your upgraded JIRA instance to work correctly (e.g. merging `jira-application.properties` customisations from the old instance to the upgraded instance).

New Location of JIRA Log Files

As of 4.1, JIRA no longer writes logs in your working directory. All logs are now written to the `$JIRA_HOME/log` directory.

Changes to Plugins

Please read Updating JIRA Plugins for JIRA 4.1.

Form Token Handling

JIRA 4.1 employs a new token authentication mechanism, which is used whenever JIRA actions are performed either through link request or form submission. This provides JIRA with the means to validate the origin and intent of the request, thus adding an additional level of security against cross-site request forgery. While the core JIRA product and its bundled plugins use this token handling mechanism by default, non-bundled plugins or those developed by third parties may not.

Therefore, if you are a JIRA plugin developer, please refer to the Form Token Handling documentation for details on how to incorporate this token handling mechanism into your JIRA plugin.

If you choose to implement form token handling into your JIRA plugin, please be aware of the following points:

- Any functions that use screen scraping, such as the 'create sub-task' function in FishEye, will be broken.
- REST API end points will not be affected *unless* they use form encoding.

Form token checking is enabled by default in JIRA 4.1. However, JIRA administrators can disable it on their site by following the instructions in Disabling Form Token Checking.

Customers with Crowd Integration

If you use Crowd with JIRA, please be aware that when you upgrade to JIRA 4.1, the `seraph-config.xml` file located in `JIRA/atlassian-jira/WEB-INF/classes/` will be overwritten to include a new element:

```
<elevatedsecurityguard class="com.atlassian.jira.security.login.JiraElevatedSecurityGuard"/>
```

This element is required for security features in JIRA 4.1.

After upgrading to JIRA 4.1, however, you may notice problems with the login gadget. For more information, please refer to JRA-21205.

If you encounter these problems, please run through step 2 (Configuring JIRA to talk to Crowd) of Integrating Crowd with Atlassian JIRA procedure to ensure that Crowd is successfully reconfigured with JIRA. If you had made any customisations to any of the files throughout this procedure prior to upgrading to JIRA 4.1 (for example, the `seraph-config.xml` file), then you will need to re-integrate these customisations after the upgrade process.

Upgrading from JIRA 3.13 and earlier

In addition to the points listed above, please read the *Upgrade Guide* for every version you are skipping during the upgrade, particularly the JIRA 4.0 Upgrade Guide as JIRA 4.0 introduced significant licensing and technical changes. The complete list of Upgrade Guides is available here: Production Releases.

Updating JIRA Plugins for JIRA 4.1

On this page:

- Plugin Developer Notes
  - Dashboard API changes
  - Issue Operation module type is no longer available
  - IssueService should be used for performing issue operations
  - User & Date Customfields
  - Form Token Handling
- JIRA 4.1 Early Access Program (EAP)

Plugin Developer Notes

JIRA 4.1 introduces several changes that may break existing plugins. If you are using a plugin that is not shipped with JIRA, the plugin may need to be updated to work with JIRA 4.1. If the plugin was written by you, please read through the information below and see if any of it is relevant to your plugin. If you are using a plugin written by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.1.

- Dashboard API changes
- Issue Operation module type is no longer available
- IssueService should be used for performing issue operations
- User & Date Customfields
- Form Token Handling

Please note that this is not the complete list of changes for JIRA 4.1; it is just the changes that plugin developers are likely to encounter most often.

Dashboard API changes

The `PortletConfigurationStore` has had the following method renamed in JIRA 4.1:

- **old name** (introduced in JIRA 4.0): addLegacyGadget
- **new name** (changed in JIRA 4.1): addLegacyPortlet

In JIRA 4.1 the `PortalPage` interface has also been updated to become a final concrete class. The `PortalPageImpl` has been removed. Instances of the `PortalPage` final class can now be constructed using the `PortalPage.Builder` class. If your plugin was using the `PortalPage` interface it will need to be re-compiled against JIRA 4.1 to use the new `PortalPage` class.

Issue Operation module type is no longer available

The `IssueOperation` module type, which allowed plugin developers to add their own links to the "Issue Operations" list on the "View Issue" screen, is no longer available. In JIRA 4.1+ any plugin that needs to add a new issue operation to the "View Issue" page will need to be updated to use a `Web-Item` module instead.

So for example to convert the existing "Edit Issue" operation you would have to change the following plugin defintion:

```
   <issue-operation key="edit-issue"
i18n-name-key="admin.issue.operations.plugin.edit.issue.name" name="Edit this issue"
class="com.atlassian.jira.issue.operations.EditIssueOperation" state='enabled'>
        <resource type="velocity" name="view"
location="templates/plugins/operations/editissue.vm" />
        <order>80</order>
    </issue-operation>
```

...to be a Web-Item:

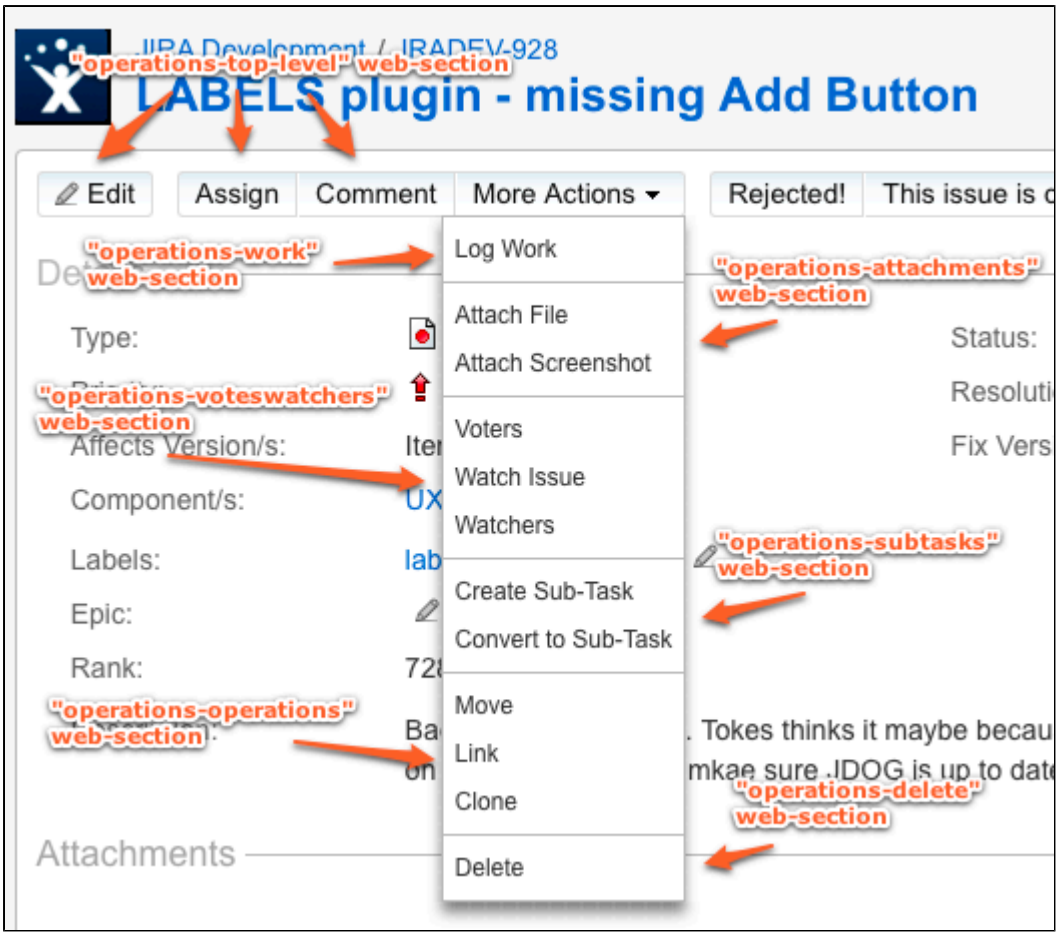
```
           <web-item key="edit-issue"
i18n-name-key="webfragments.view.issue.opsbar.operations.top.edit" name="Edit this issue"
section="operations-top-level" weight="1">
                <label key="common.words.edit"/>
                <tooltip key="admin.issue.operations.plugin.edit.issue.name"/>
                <link linkId="edit-issue">
                    /secure/EditIssue!default.jspa?id=${issue.id}
                </link>
                <condition
class="com.atlassian.jira.plugin.webfragment.conditions.IsIssueEditableCondition"/>
                <condition
class="com.atlassian.jira.plugin.webfragment.conditions.HasIssuePermissionCondition">
                    <param name="permission">edit</param>
                </condition>
                <condition
class="com.atlassian.jira.plugin.webfragment.conditions.ContextContainsCondition"
invert="true">
                    <param name="context-key">display-context</param>
                    <param name="context-value">view-issue</param>
                </condition>
            </web-item>
```

The backing `EditIssueOperation` class required previously by the Issue Operation Module is no longer required.

Issue Operation Web-Items need to be added to an appropriate Web Section. The default structure for the issue operation sections is as follows:

- web-section key="opsbar-operations" name="Ops Bar Operations Section" location="view.issue.opsbar"
  - web-section key="operations-top-level" name="Ops Bar Operations Top level Section" location="opsbar-operations"
  - web-section key="operations-work" name="Ops Bar Operations Work Section" location="opsbar-operations"
  - web-section key="operations-attachments" name="Ops Bar Operations Attachments Section" location="opsbar-operations"
  - web-section key="operations-voteswatchers" name="Ops Bar Operations Votes & Watchers Section" location="opsbar-operations"
  - web-section key="operations-subtasks" name="Ops Bar Operations Subtask Section Section" location="opsbar-operations"
  - web-section key="operations-operations" name="Ops Bar Operations Operations Section Section" location="opsbar-operations"
  - web-section key="operations-delete" name="Ops Bar Operations Delete Section Section" location="opsbar-operations"

This structure gives the following default menu:



Plugin developers can add an issue operation `web-item` to any of the default menu sections, or define their own menu web-section and add it there.

IssueService should be used for performing issue operations

JIRA 4.1 introduces a new `IssueService` for performing operations (e.g. create/read/update/delete) on issues, which makes it much easier to perform issue operations from within a plugin. Plugin developers are strongly encouraged to change their plugins to use the new `IssueService`. The existing "back-end Actions" (e.g. `ISSUE_UPDATE`, `ISSUE_DELETE`) have been deprecated in favour of the new `IssueService` and may be removed in future releases of JIRA. The `IssueManager` class should also no longer be used directly to create or retrieve issues, as the new `IssueService` provides more robust validation and error handling.

For more information please see the detailed documentation on the `IssueService`.

User & Date Customfields

With the new View Issue page in JIRA 4.1, Date and User fields are grouped together in their own sections.

In order for Customfields to be placed in either section, their implementation of CustomFieldType must also implement either DateField or UserField.

DateField.java

```
/**
 * A marker interface to mark all date fields available in the system.  Please note that for
custom fields, the
 * custom field type needs to be marked by this interface.
 *
 * @since v4.0
 */
public interface DateField
{
}
```

UserField.java

```
/**
 * A marker interface to mark all user fields available in the system.  Please note that for
custom fields, the
 * custom field type needs to be marked by this interface.
 *
 * @since v4.0
 */
public interface UserField
{
}
```

Form Token Handling

JIRA 4.1 employs a new token authentication mechanism, which is used whenever JIRA actions are performed either through link request or form submission. This provides JIRA with the means to validate the origin and intent of the request, thus adding an additional level of security against cross-site request forgery. While the core JIRA product and its bundled plugins use this token handling mechanism by default, non-bundled plugins or those developed by third parties may not.

Therefore, if you are a JIRA plugin developer, please refer to the Form Token Handling documentation for details on how to incorporate this token handling mechanism into your JIRA plugin.

If you choose to implement form token handling into your JIRA plugin, please be aware of the following points:

- Any functions that use screen scraping, such as the 'create sub-task' function in FishEye, will be broken.
- REST API end points will not be affected *unless* they use form encoding.

Form token checking is enabled by default in JIRA 4.1. However, JIRA administrators can disable it on their site by following the instructions in Disabling Form Token Checking.

JIRA 4.1 Early Access Program (EAP)

Pre-release versions of JIRA 4.1 can be downloaded from our main Atlassian website or from one of the links below.

JIRA plugin developers and other interested parties can download and install these pre-release versions to:

- Help update plugins for JIRA 4.1 compatibility and
- Check out JIRA 4.1's new features.

Do not use pre-release JIRA builds in production!

Beta releases should not be used in production environments as they may still contain bugs and are not officially supported. Please use these builds at your own risk.

You can download one of the following pre-release JIRA beta distributions that best suits your needs:

- JIRA 4.1.0-beta Enterprise - WAR/EAR (TAR.GZ Archive)
- JIRA 4.1.0-beta Enterprise - WAR/EAR (ZIP Archive)
- JIRA 4.1.0-beta Enterprise - Standalone (ZIP Archive)

JIRA 4.1.2 Release Notes

18 June 2010

The Atlassian JIRA team is proud to announce the release of **JIRA 4.1.2**. This point release is a **highly recommended upgrade** as it contains important fixes to security vulnerabilities — please refer to the JIRA Security Advisory 2010-06-18 for details.

This point release also contains over 40 fixes and improvements, notably including:

- **New 'Comment' button** — You can now add comments via a 'Comment' button at the end of the 'View Issue' page. Clicking this button opens a large text box immediately below the last comment, into which you can add your new comment. This function is useful when you want to reply to recently added comments. For more information, please refer to Commenting on an Issue.

- **'HttpOnly' Session Cookies** — To improve the security of JIRA, JIRA standalone distributions (version 4.1.2 and later) now implement 'HttpOnly' session ID cookies, as indicated in our Security Advisory. Please refer to our Preventing Security Attacks guide for details on implementing 'HttpOnly' session ID cookes for JIRA EAR-WAR distributions.

Please also note that the bundled FishEye plugin and Bamboo plugin have been updated with security improvements, which are also mentioned in our Security Advisory.

JIRA 4.1.2 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.1 yet?
Take a look at all the new features in the JIRA 4.1 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.1.2 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.1.2 includes the following updates and bug fixes:

JIRA Issues (52 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-21531	Support Safari 5		Resolved
	JRA-21333	JIRA Calendar Plugin page should state it is a Plugin 2 plugin		Closed
	JRA-21157	Current Jira create issues button and popdown unusable in Firefox with projects with long names		Closed
	JRA-21149	Put a new security log into JIRA so that important events can be specifically logged		Resolved
	JRA-21033	Support HttpOnly cookies		Resolved
	JRA-20673	Include a list of all known JIRA application properties in the System Info page		Resolved
	JRA-15940	Priority interface should expose a method to get the status colour/color		Resolved
	JRA-11682	Put an 'Add comment' link at the bottom of the page.		Resolved
	JRA-21624	Set of critical security vulnerabilities		Resolved
	JRA-21322	Attach Screenshot applet no longer works in 4.1.2		Resolved
	JRA-21208	Tomcat supportability on Supporte Platform page needs to be fixed		Closed
	JRA-20964	Upgrade Link in 4.1 Installer is Broken		Resolved
	JRA-20963	SSL connector in server.xml incorrect for JIRA 4.1 Standalone		Resolved
	JRA-20940	JIRA Activity Stream gadget is broken when accessed externally using IE		Resolved
	JRA-20528	Workflow plugin modules defined in V2 plugins will sometimes not work		Resolved
	JRA-19851	Cannot authorise "Created vs Resolved Chart" gadget in Confluence		Resolved
	JRA-21397	View issue layout broken by wiki markup in comments - people and dates box shown in footer		Resolved
	JRA-21363	Voting for issue is not working on FF 3.5		Resolved
	JRA-21192	JIRA config tool does not like "escaped colons" in the jira-application.properties file		Resolved

	JRA-21190	Broken links in the Windows Installer		Resolved
	JRA-20917	Sperious Warnings Displayed at Start-up		Resolved
	JRA-20776	New comment UI is difficult to use when replying		Resolved
	JRA-20675	Can't vote for issues in JIRA 4.1-rc1		Resolved
	JRA-20620	The javascript shortenting participants, components and versions performs badly if there's a lot of options (100+)		Resolved
	JRA-20578	Shared filter queries break when users don't have browse permissions for projects in the search query		Resolved
	JRA-20554	Some operations fail with "ERROR: operator does not exist" when JIRA is connected to Postgres 8.3+		Resolved
	JRA-20417	JIRA Activity Stream gadget displays error in Confluence due to en_GB / en_UK mismatch		Resolved
	JRA-20235	Attach screenshot link doesn't work if group name includes ''		Resolved
	JRA-20018	Non-standard HTML is generated when embedding flash movies		Resolved
	JRA-19733	Recently Created Issues Chart is no longer clickable and doesn't show hover information		Resolved
	JRA-19225	"How to Upgrade JIRA" link in JIRA installer is not pointing to right link		Resolved
	JRA-21848	NO automatically generated password for JIRA 4.0.x when password is not filled in		Resolved
	JRA-21402	AttachScreenshot applet - when you cancel the dialog, you end up with empty screenshot-redirecter.jsp popup		Resolved
	JRA-21175	Watch Issue doesn't work with Opera		Resolved
	JRA-21145	When adding comment, if window is narrow, the comment entry box is wider than the window and cannot be scrolled horizontally		Resolved
	JRA-20968	'Current User Sessions' page should say '1 through x' (not '0 through x')		Resolved
	JRA-20905	the "next issue arrow" on view issue sits in the spot of the prev issue arrow		Resolved
	JRA-20837	"ISSUE DOES NOT EXIST" heading should be sentence case		Resolved
	JRA-20712	Stalkerbar overlaps comment header when permalinks are accessed through browser address bar		Resolved
	JRA-20657	Multi User Picker autocomplete creates extra spaces in between values when selecting multiple values		Resolved
	JRA-20638	Voted Issues gadget - clicking "Include resolved issues" toggles "Show the total..."		Resolved
	JRA-20618	List of available locales is rendered in the JVM locale's language instead of the user's locale		Resolved
	JRA-20589	It's possible to create users with leading and trailing spaces when signing up for an account		Resolved
	JRA-20573	User autocomplete displays "Oops! You have already entered the value" error for a non-duplicate value		Resolved
	JRA-20291	Mac OSX install Movie wonderful - but not current		Closed
	JRA-20241	JIRA currently cannot search for colons (:) in text fields or this feature appears to be broken.		

				Resolved
	JRA-19910	UserAgentUtilImpl throws NPE when user-agent HTTP header is null		Resolved
	JRA-15256	Could not find atlassian-jira.log		Resolved
	JRA-21766	Documentation mistake, 'Configure' instead of 'Screen'		Closed
	JRA-20198	Superfluous text in Issue Navigator when a filter is modified in Advanced mode		Resolved
	JRA-19714	Misleading error for a deactivated user trying to log into Jira		Closed
	JRA-3436	Image thumbnails not displayed when image content-type is "image/pjpeg"		Resolved

JIRA 4.1.2 Upgrade Guide

Upgrading from JIRA 4.1 or 4.1.1 to 4.1.2

Please follow the JIRA general upgrade instructions, plus note the following:

- A loophole has been closed whereby users who have the 'JIRA Administrators' permission but not the 'JIRA Users' permission were previously able to log in to JIRA. After the upgrade to JIRA 4.1.2, if you find that JIRA administrators can no longer log in, please check that they have been granted the 'JIRA Users' permission (see Managing Global Permissions).
- For JIRA instances connected to Atlassian Crowd, a loophole has been closed whereby users who do not have the 'JIRA Users' permission were previously able to log in to JIRA via Crowd Single Sign On (SSO). After the upgrade to JIRA 4.1.2, if you find that users can no longer login, please check that they have been granted the 'JIRA Users' permission (see Managing Global Permissions).

Upgrading from JIRA 4.0.x and earlier

In addition to the above, please read the JIRA 4.1 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.1.1 Release Notes

21 April 2010

The Atlassian JIRA team announces the release of **JIRA 4.1.1**. This point release is a **highly recommended** upgrade as it contains important fixes to security vulnerabilities in JIRA (listed below). For more information about these security vulnerabilities and patches to fix these vulnerabilities in earlier versions of JIRA, please refer to the JIRA Security Advisory 2010-04-16.

Please also refer to the JIRA 4.1.1 Upgrade Guide for important changes in JIRA, which are designed to minimise the risk of security attacks.

JIRA 4.1.1 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4.1 yet?
Take a look at all the new features in the JIRA 4.1 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.1.1 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.1.1 includes the following updates and bug fixes:

JIRA Issues (16 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-21003	The AJP connector should always have the URIEncoding="UTF-8" set		Closed
	JRA-20782	Save gadget setting (maximized, minimized, normal) between logon sessions		Closed
	JRA-21004	XSS and Privilege Escalation Vulnerabilities in JIRA		Resolved

	JRA-20995	Privilege escalation vulnerability when administrator access is compromised		Resolved
	JRA-20994	XSS Vulnerabilities in JIRA		Resolved
	JRA-21038	brute force password attack protection by default		Resolved
	JRA-21024	500page.jsp contains HTTP Header XSS vulnerability		Resolved
	JRA-21023	screenshot-redirecter.jsp XSS attach via the afterURL parameter		Resolved
	JRA-21022	issuelinkssmall.jsp has an XSS hole via the URL used to access it		Resolved
	JRA-21019	runportleterror.jsp contains XSS hole		Resolved
	JRA-21018	Miscellaneous support-related JSPs contain XSS holes		Resolved
	JRA-21017	Announcement Preview banner is a vector for an XSS attack		Resolved
	JRA-20665	xss vulnerability in issuelinksmall.jsp		Resolved
	JRA-21037	Group picker popup JSP has XSS hole if group names are XSS shaped		Resolved
	JRA-21150	Soap deleteProject call may try to delete an issue more than one time causing it to fail with an Exception		Resolved
	JRA-20446	Right clicking on dashboard links when there are a lot of gadgets is super slow.		Resolved

JIRA 4.1.1 Upgrade Guide

On this page:

- Upgrade Notes
  - Setting File Paths via the Administration User Interface
    - Changing JIRA's File Path Settings
    - 'Contact Administrators' Link
  - Backing Up Data to XML
  - Announcement Banner
  - Data for Support Requests
  - Differences between JIRA 4.1.1 and the Security Patches for Earlier JIRA Versions
  - Developers Note
- Upgrading from JIRA 4.1 to 4.1.1
- Upgrading from JIRA 4.0.x and Earlier

Upgrade Notes

JIRA 4.1.1 fixes several security vulnerabilities in JIRA. Patches that fix these vulnerabilities in earlier versions of JIRA are also available. Please refer to the JIRA Security Advisory 2010-04-16 or JIRA issue JRA-21004 for more information about these vulnerabilities and links to these patches.

Please be aware that these fixes have resulted in the following changes to JIRA's behaviour.

Setting File Paths via the Administration User Interface

JIRA now recognises a new variable called **`jira.paths.set.allowed`** in the `jira-application.properties` file (located in your JIRA Installation Directory).

By default, the value of this variable is set to `false`, such that it appears as `jira.paths.set.allowed=false` in the `jira-application.properties` file.

JIRA's file path settings are secure when any of the following is true:

- the `jira.paths.set.allowed` variable in `jira-atlassian.properties` is set to `false`
- the `jira.paths.set.allowed` variable in `jira-atlassian.properties` is set to anything other than `true` or its value is left blank
- the `jira.paths.set.allowed` property does not exist in `jira-atlassian.properties` or it is 'commented-out'

and the following JIRA screens:

.../secure/admin/ViewAttachmentSettings.jspa (see Configuring File Attachments)
.../secure/admin/IndexActivate.jspa (see Search Indexing)
.../secure/admin/jira/ViewServices!default.jspa (see Automating JIRA Backups)
.../secure/admin/XmlRestore!default.jspa (see Restoring Data)

will display this message:

> *Changing the attachment, index, backup or restore settings is not allowed for security reasons. You must edit jira-application.properties and explicitly set 'jira.paths.set.allowed=true'. Restart JIRA and then the path settings will be able to be changed.*

Changing JIRA's File Path Settings

If you want to change the locations for storing file attachments, backups, etc, you will need to do the following:

1. Shutdown JIRA.
2. Ensure `jira.paths.set.allowed=true` has been set in the `jira-application.properties` file and restart JIRA.
3. Perform your location changes and shutdown JIRA.
4. Secure JIRA's file path settings again by disabling the `jira.paths.set.allowed` property in `jira-application.properties` using one of the methods above.
   Although this step is optional, it is *strongly recommended* as it will minimise the risk of attack to your JIRA instance.
5. Restart JIRA.

Upon setting the value of the `jira.paths.set.allowed` variable to `true` in `jira-atlassian.properties`, this message is displayed in the screens above:

> *You have enabled the ability to change attachment, index, backup or restore path settings from within JIRA. Having this setting on can cause a known security risk. See http://jira.atlassian.com/browse/JRA-21004 for more details*
> *To re-enable stronger security, edit jira-application.properties and explicitly set 'jira.paths.set.allowed=false'. Restart JIRA and then the path settings will be NOT able to be changed.*

'Contact Administrators' Link

For security reasons, the list of JIRA administrators, which can be accessed via the 'Contact Administrators' link in the JIRA footer, will be blank unless `jira.paths.set.allowed` is set to `true` (which is not recommended — see above).

Backing Up Data to XML

JIRA now recognises another new variable called **`jira.paths.safe.backup.path`** in the `jira-application.properties` file (located in your JIRA Installation Directory).

By default, this variable is present in the `jira-application.properties` file, but it is disabled ('commented-out') and its value is an example directory path value only. If you enable the `jira.paths.safe.backup.path` variable and set its value to a valid directory, the following screen in JIRA:

.../secure/admin/XmlBackup!default.jspa (see Backing Up Data for more information)

will display this message:

> *You have named a safe backup directory. Any arbitrary backups will be written to this directory.*

Otherwise, this message is displayed:

> *You have not named a safe backup directory and hence you are not allowed to make backups for security reasons. You must edit jira-application.properties and explicitly set 'jira.paths.safe.backup.path=/to/some/safe/path'. Restart JIRA and then you will be able to make arbitrary backups. NOTE : If you are using Windows, you will need to use double \ characters, for example*
>
> ```
> d:\\some\\safe\\path
> ```

Examples of valid directory paths used with this variable:

- **UNIX-based systems (e.g. Linux or Mac OS X)**

  ```
  jira.paths.safe.backup.path=/some/safe/path
  ```

- **Windows systems**

  ```
  jira.paths.safe.backup.path=d:\\some\\safe\\path
  ```

JIRA's manual 'Backup Data to XML' feature will not be available unless the value of the `jira.paths.safe.backup.path` variable in `jira-application.properties` has been set to a valid path.

Announcement Banner

For security reasons, the ability to preview the Announcement Banner has been disabled.

Data for Support Requests

For security reasons, we no longer attach XML backups and logs to the emails generated by the Support Request page.

Differences between JIRA 4.1.1 and the Security Patches for Earlier JIRA Versions

The main purpose of the JIRA 4.1.1 point release was to fix several security vulnerabilities in JIRA. (Patches to fix these vulnerabilities in earlier versions of JIRA can be obtained via the JIRA Security Advisory 2010-04-16 or JIRA issue JRA-21004.)

However, that there are some differences in behaviour between JIRA 4.1.1 and the patches applied to earlier JIRA versions:

- Upon upgrading to or initially installing JIRA 4.1.1, Captcha will automatically be activated after five failed login attempts.

  If you had set the Maximum Authentication Attempts Allowed option (via the '**Administration**' -> '**Global Settings**' -> '**General Configuration**' page) to another value prior to upgrading, it will be overridden and set to 5 upon upgrading to JIRA 4.1.1. Hence, to revert this option back to your previous setting, you will need to do this manually via the 'Global Settings' -> 'General Configuration' page.

- From JIRA 4.1.1, the following additional JSP pages have been disabled and are no longer available:
  - .../secure/admin/cacheViewer.jsp
  - .../secure/admin/editworklog.jsp
  - .../secure/admin/manageyourkitprofiling.jsp
  - .../secure/admin/plugin-bundles.jsp
  - .../secure/admin/workflow-debug.jsp

Developers Note

When using the Atlassian SDK, the correct JIRA version to reference is 4.1.1.1 (not 4.1.1). See the Atlassian Plugin SDK 3.1.2 Release Notes for details.

Upgrading from JIRA 4.1 to 4.1.1

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 4.0.x and Earlier

In addition to the above, please read the JIRA 4.1 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.0 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

October 6, 2009

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

We have improved the UI to provide contextual awareness, improving the navigation and usability with features like activity streams and issue history. We have also added the most powerful searching capabilities ever seen in a bug tracker, called JIRA Query Language (JQL). The simple auto-complete entry system makes it incredibly easy for any user to create sophisticated queries.

We have completely overhauled the JIRA dashboards to make it quick and easy for anyone to create and add gadgets, move them around using drag & drop, and share dashboards with other team members. Using the new Plugins 2.0 architecture, we built new dashboards using OpenSocial. This means each JIRA dashboard is an OpenSocial container allowing you to consume any OpenSocial compliant gadgets from either Atlassian tools or other external sources. Conversly, JIRA gadgets can be exposed in any OpenSocial container (like iGoogle, and soon, Confluence).

Upgrading to JIRA 4.0 is free for all customers with active JIRA software maintenance as of October 6, 2009.

Highlights of JIRA 4.0:

- Advanced Searching

- Dashboard Gadgets
- Activity Streams
- New-look "Browse Project"
- Charting Now Comes Standard
- New-look Header
- Issue Actions in the Issue Navigator
- Project Icons
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- Plus over 900 other fixes and improvements

Thank you for your feedback:

More than 50 new feature requests implemented

More than 2600 votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

DOWNLOAD latest version

Upgrading to JIRA 4.0
JIRA 4.0 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide. Also note that you will need to obtain a **new license key** before you can upgrade. Obtaining a new JIRA 4 license key is **free** and the key is valid for the remainder of your existing maintenance period. For details, please see the JIRA licensing changes FAQ.

Highlights of JIRA 4.0

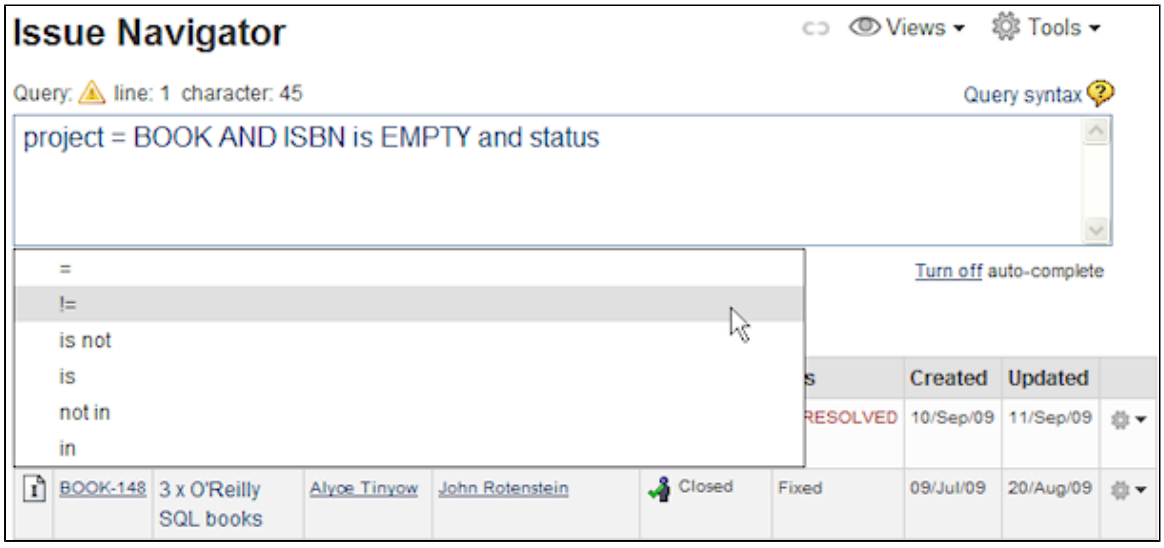
1

Advanced Searching

The power of search can never be understated, especially in a system like JIRA that sits at the centre of your development team.

JIRA Query Language (or JQL) brings search to whole new level!

JQL is a structured query language that provides support for logical operations, including AND, OR, NOT, NULL, EMPTY — even on custom fields:



Using JQL is simple even for those who don't know what "DBA" means. Just start typing and the auto-complete feature starts to suggest fields, operators and values for you to define your query.

You can now create more advanced filters such that you can stay up to date using RSS feeds & e-mail subscriptions, as well as see detailed statistics and charts, on issues that you are actually interested in.
^Top

Dashboard Gadgets

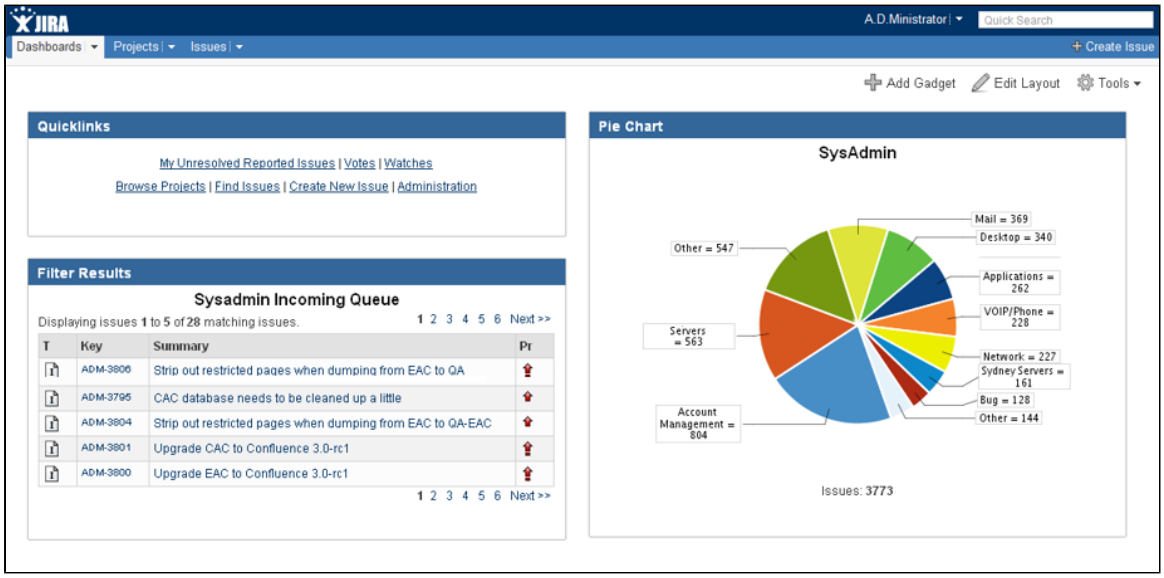
Whether you are tracking bugs or managing your entire development process, JIRA dashboards let you stay up to date on what matters most.

The new-look JIRA dashboard not only looks awesome, it now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

You can easily customise your dashboard by choosing a different layout, adding more gadgets, dragging the gadgets into different positions and changing the look of individual gadgets.

What's happened to your favourite JIRA portlets? Don't worry, every portlet that previously shipped with JIRA has been converted to a gadget.

If you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.



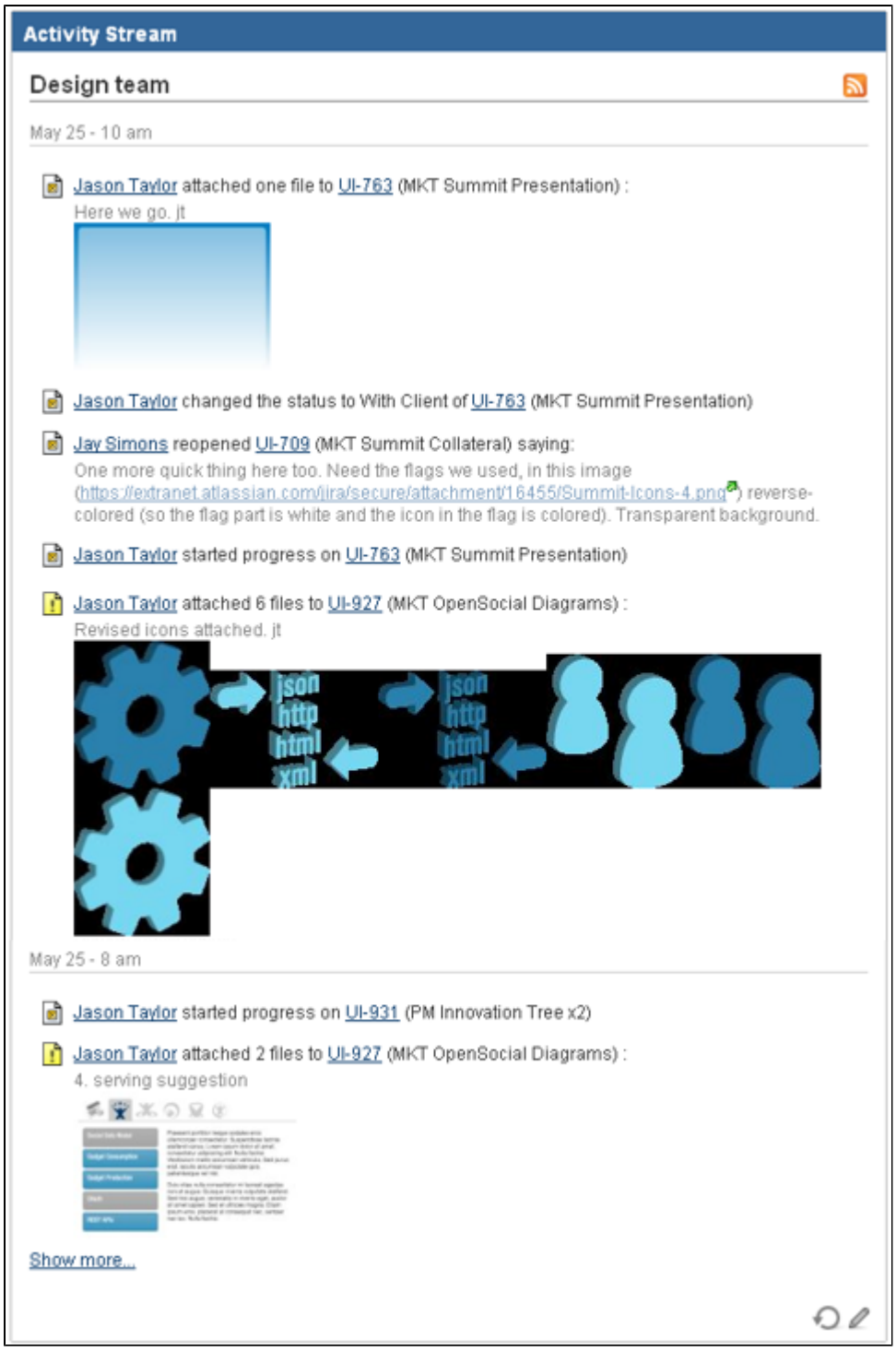
^Top

Activity Streams

The new activity stream allows you to stay up to date with exactly what is going on right this moment, what happened in that last hour or last few days.

Activity streams appear where you need them most — your user profile, project summary and view issue screens. You can even add an activity stream as a gadget on your dashboard.

The activity stream also provides an RSS feed, allowing you to subscribe to very specific RSS feeds of only the information that is most relevant to you.



See the documentation for more details.

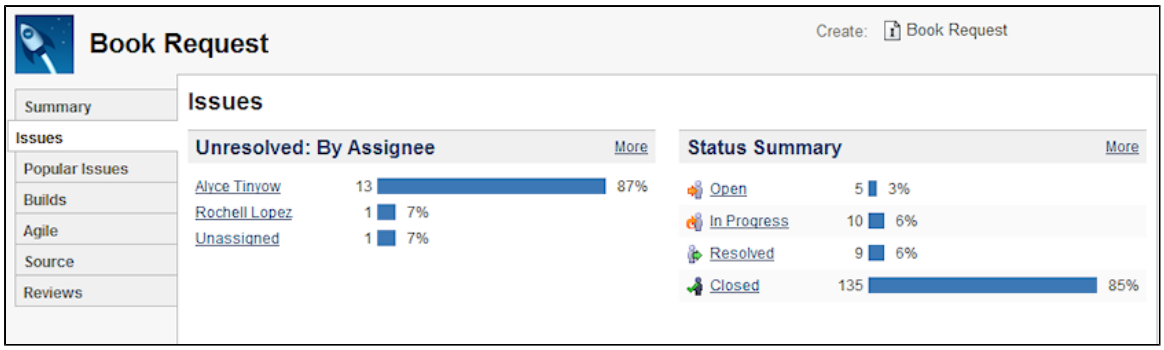
^Top

New-look "Browse Project"

Understanding the status of your projects just got a lot easier with the new browse project UI.

Quickly see what work is complete as well as outstanding. You can then drill down to specific issues you want to see.

Your Bamboo builds, FishEye source information and Crucible code reviews are only a click away, as well.

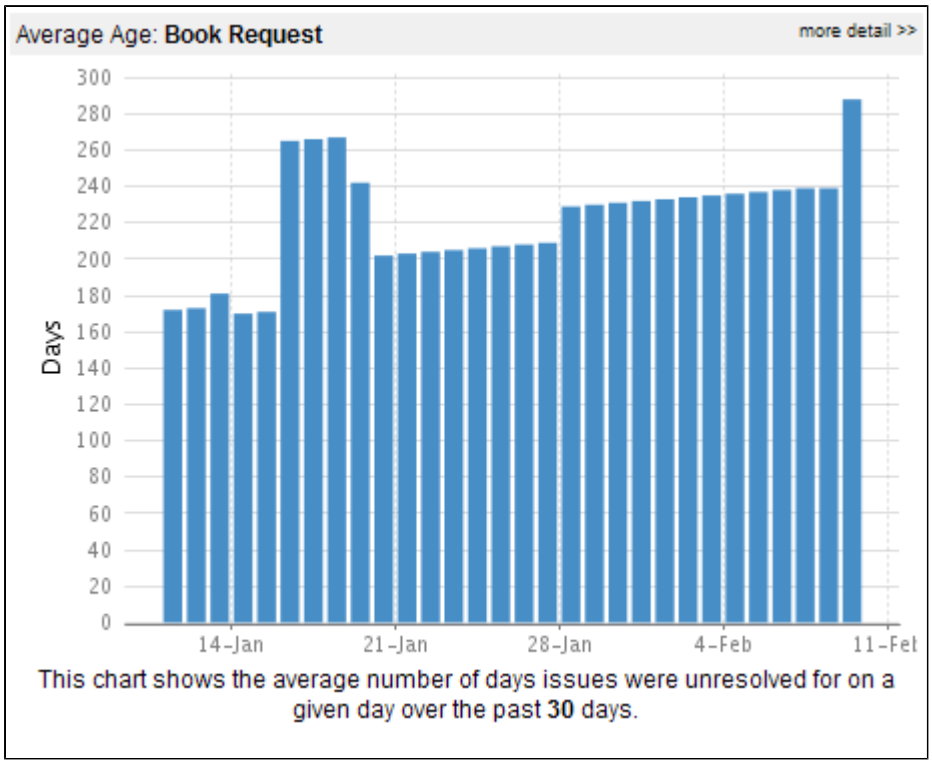


See the documentation for more about browsing projects, versions and components.
^Top

Charting Now Comes Standard

We've built charts into JIRA and given them a visual redesign as well.

- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over a given period of time.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.
- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



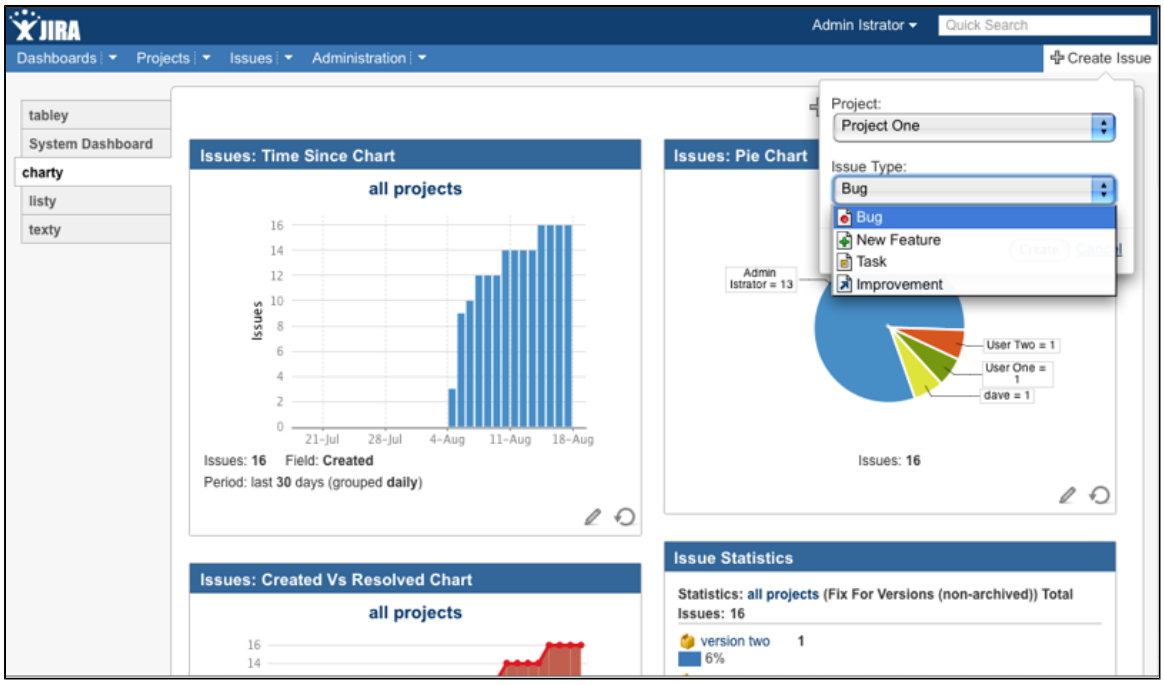
Also, the "Resolution Time" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution time recorded.

^Top

6

New-look Header

The new-look JIRA header gives you quick access to all of the most commonly-used functions. Creating an issue just got even faster!



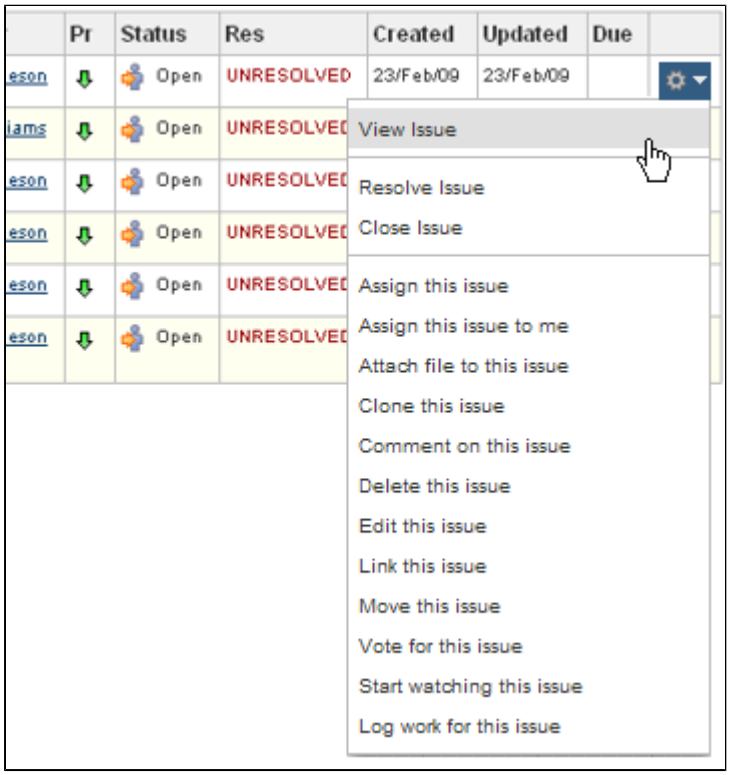
Click to zoom in:

If you prefer keystrokes rather than mouse-clicks, you'll be pleased to know that you can use your keyboard to navigate the new header menus.
^Top

7

Issue Actions in the Issue Navigator

By popular request, issues are now actionable directly from the Issue Navigator:

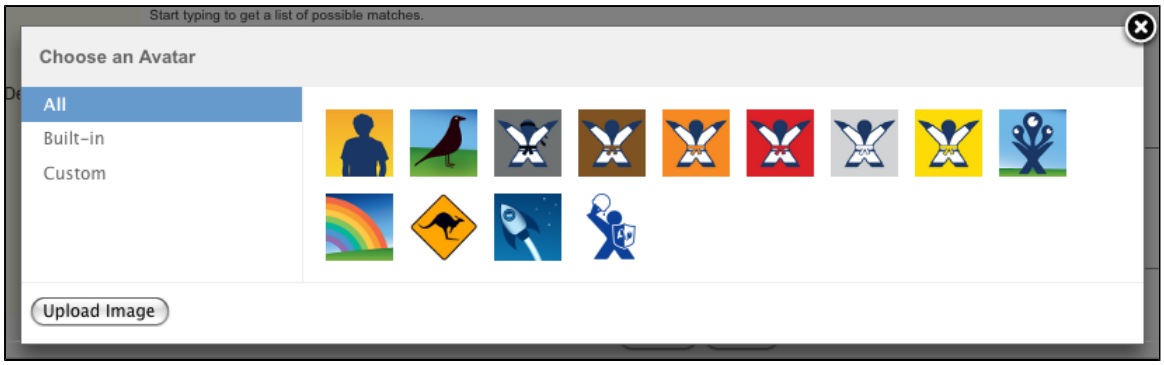


The "Actions" menu is also available for the list of sub-tasks within an issue.
^Top

8

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):



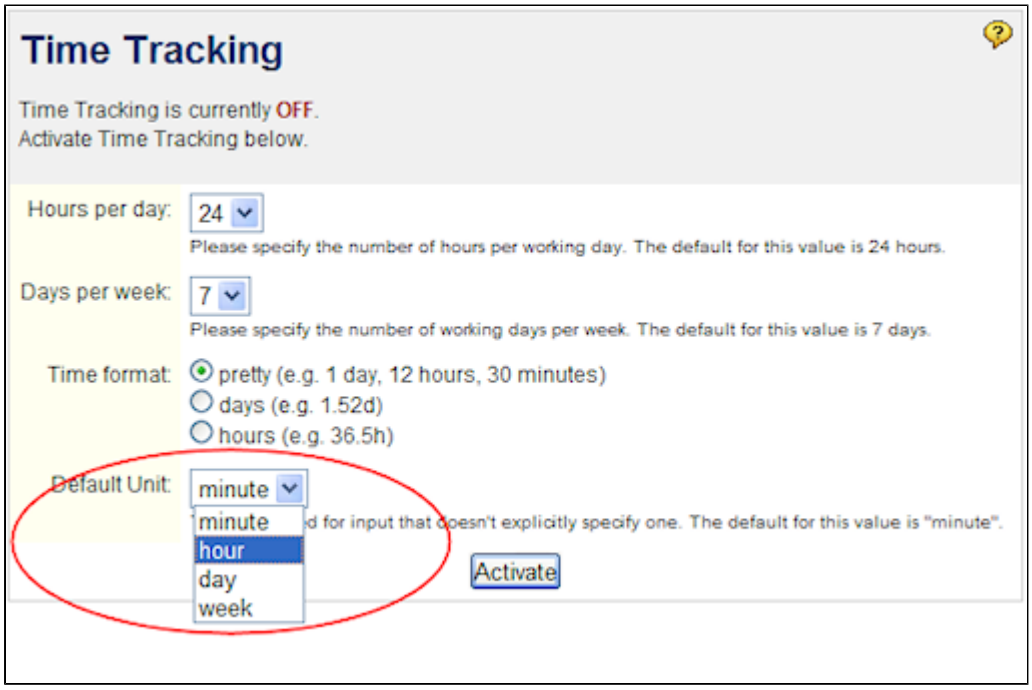
^Top

9

Default Unit for Time Tracking

You can now specify your preferred **Default Unit** (minutes/hours/days/weeks) for your JIRA system. This will be applied whenever users log

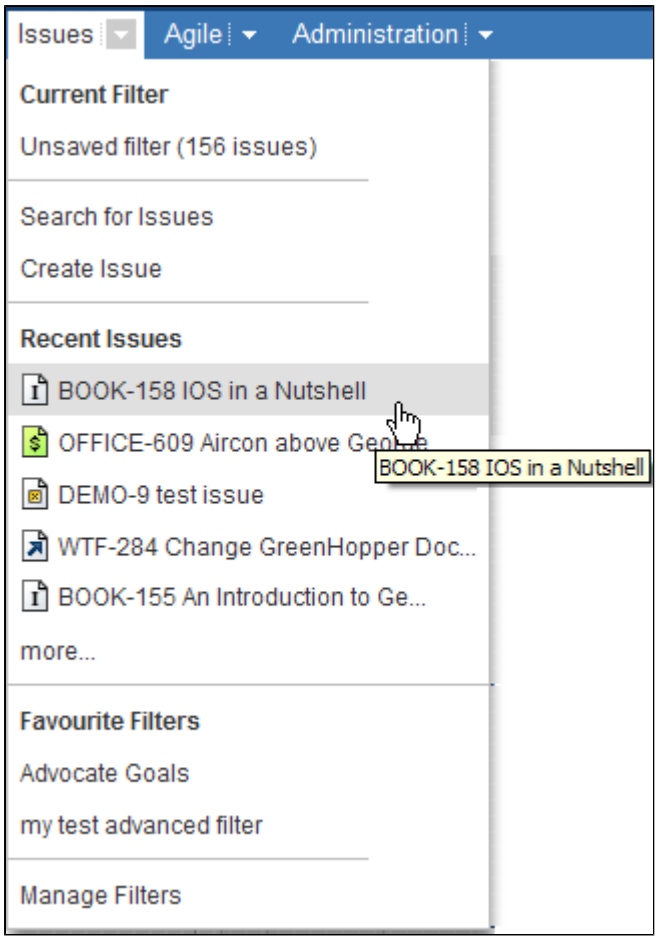
work on an issue without specifying a unit.



^Top

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.



^Top

Engine Room

Beyond the 'Back' Button

When navigating away from a page where you have modified data, you will be prompted to see if you would like to save the data or discard your changes (see JRA-14911).

Index Queue

Index updates are now put in a queue. So even if the update takes longer than 30 seconds, the operation remains on the queue and is not lost. (See JRA-14220.)

^Top

Plus over 900 other fixes and improvements

The top 50 most popular issues resolved in JIRA 4.0 are listed below. Click here for the full list.

JIRA Issues (50 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-1560		Better support for logical operation (and/or/not) type of filters.		Resolved

JRA-7909	Search/ filter for "empty" fields	Resolved
JRA-1538	Filter on Versions and Components across Projects	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-1844	Display attachment comments associated with their attachments	Resolved
JRA-5383	My Votes and My Watches as filters	Resolved
JRA-5844	allow multiple users when creating filters	Resolved
JRA-6180	Search for a custom field that is empty	Resolved
JRA-5121	Filter Portlet with configurable columns	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-6527	Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-9551	Search for all Sub-Tasks of one given issue	Resolved
JRA-7772	Ability to create advanced queries to search across all data	Resolved
JRA-7626	Build search queries remotely	Resolved
JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-7068	Allow for list of issues to be saved as a filter	Resolved
JRA-5965	Allow configure units of time tracking	Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.	Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"	Resolved
JRA-5560	Improved query functionality	Resolved
JRA-1635	"not" qualifier on fields for searching	Resolved
JRA-6344	Send to both previous and current assignees for all notifications	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-5201	Enable filter to specify more than 1 user	Resolved
JRA-10405	Attachment ordering	Resolved

JRA-5152	Show issue linked to another issue.		Resolved
JRA-10492	Search for several users as Assignee or Reporter		Resolved
JRA-3451	Enable filtering by Date Resolved		Resolved
JRA-16744	Improve the performance of checking if a user belongs to a particular group.		Resolved
JRA-8758	Cannot create filter for multiple projects all issues in version "Released Versions"		Resolved
JRA-10427	Changing field descriptions in "Field Configurations" for custom fields does not work	↑	Resolved
JRA-8606	Need a way to find watched issues		Resolved
JRA-8159	Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type		Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria		Resolved
JRA-14031	Form data lost when using back and forward web browser buttons	↑	Resolved
JRA-9115	Ability to search for issues with no due date associated		Resolved
JRA-9048	Calendar week begins with sunday independently from locale	↑	Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype		Resolved
JRA-14983	Fetch only updated or changed issues		Resolved
JRA-10658	More columns on Dashboards		Resolved
JRA-8973	RSS of Project Changes		Resolved
JRA-14613	Each project can have its own logo		Resolved
JRA-2852	search for issues on version lower or equal to a given version		Resolved
JRA-1800	Improve the UI for browse project		Resolved
JRA-923	Allow filter by "No Fix For" across projects		Resolved
JRA-15546	Versions no longer display descriptions when browsing project	↑	Resolved
JRA-13801	Call method addWorklogAndAutoAdjustRemainingEstimate, the soap server reponse with this type IssueServiceImpl$RemoteWorklogImpl	↓	Resolved

^Top

JIRA 4.0 Upgrade Guide

On this page:

- Upgrading from JIRA 3.13.x to 4.0
  - Licensing Changes
  - General Upgrade Instructions
  - Scheduling the Upgrade
  - Java Version
  - JIRA Home
  - Recommended Browsers
  - Users May Encounter 'Advanced Search'

- 'mail.mime.decodeparameters' System Property
- 'Resolution Date' System Field
- Database Schema Changes
- Charting Plugin
- JIRA Toolkit
- GreenHopper Plugin
- Issue 'Status' Field Problem
- Plugin Notes
- Memory

- Upgrading from JIRA 3.12 and earlier
  - Upgrading from JIRA 2.x Data and earlier

Before You Upgrade

Please note the following points before starting your upgrade:

- **Problems running JIRA 4.0 with WebLogic** — We are currently aware of an issue that is preventing JIRA 4.0 from running on WebLogic. If you are currently using JIRA with WebLogic, we strongly recommend that you **do not upgrade JIRA** until a fix is available. You can track the progress of this issue here: JRA-19367
- **Problems running JIRA 4.0 with IBM JVM and JRocket JVM** — We recommend that you **use the Sun JVM** with JIRA 4.0. We are currently aware of issues preventing JIRA 4.0 from working with the IBM JVM and JRocket JVM. You can track the progress of these issues here:
  - IBM JVM issue (JRA-19379)
  - JRocket JVM issue (JRA-19384)
- **Problems using the JIRA Portlet Macro in Confluence 3.0.x with JIRA 4.0**, please read JIRA Portlet Macro page in the Confluence documentation for further information.
- If you are upgrading from **JIRA 3.12 or earlier**, please read the 'Upgrading from JIRA 3.12 and earlier' section below before starting your upgrade.

Tomcat, Apache and mod_proxy setup

Unlike previous versions of JIRA, the new JIRA 4 Dashboard frequently makes HTTP requests *to itself*. For this reason, the hostname, port and protocol (http/https) must be correct throughout all portions of the request chain. Additionally, if you are using SSL, JIRA's JVM must be able to trust the SSL certificate on a JIRA response. If your setup is not configured correctly, the Dashboard in JIRA will not work. Please read this knowledge base article if you are having problems.

Upgrading from JIRA 3.13.x to 4.0

Licensing Changes

JIRA 4.0 introduces some **significant licensing changes**. Before you begin the upgrade, please go to my.atlassian.com for your upgraded license. Please note, any existing 3.x license files **will not work** with 4.0.

JIRA has moved to a **user-based licensing model** in JIRA 4.0. This means that you will need to calculate the number of users in your JIRA system to determine what license you will need, before you can complete your upgrade. For more information on this, please see the JIRA pricing changes FAQ.

Once you have upgraded your license, please follow the instructions below:

General Upgrade Instructions

Please ensure that you follow the instructions in the general JIRA upgrade guide (non-version specific), as well as the JIRA 4.0 specific instructions in the sections below. The general upgrade guide contains important tasks that are essential for getting your upgraded JIRA instance to work correctly (e.g. merging `jira-application.properties` customisations from the old instance to the upgraded instance).

Scheduling the Upgrade

Please note that upgrading to JIRA 4.0 may take a long time, depending on the size of your instance as well as server and database performance. During the upgrade, several upgrade tasks will need to run to upgrade your data to be ready for JIRA 4.0, such as:

- Calculating a resolution date for all resolved issues in your system
- Re-indexing your issues
- Converting saved filters to use JQL
- Converting existing portlets to gadgets

Please schedule sufficient downtime time for the upgrade in your production environment. It is recommended to run an upgrade first in a test environment to see how long the upgrade will take for your data set and hardware configuration.

Java Version

If you are running JIRA under version 6 (1.6) of the Sun JRE, please ensure that you are running a point release JRE 6 (1.6) Update 10 or higher. The reasons for this are:

- JIRA 4.0 introduces a new REST plugin type based on Jersey, which will not work with JRE 6 - JRE 6 Update 3. If you are running JIRA with one of these versions of the JRE you will see the following errors:

```
java.lang.LinkageError: JAXB 2.0 API is being loaded from the bootstrap classloader,
but this RI (from bundle://16.0:3/com/sun/xml/bind/v2/model/impl/ModelBuilder.class)
needs 2.1 API. Use the endorsed directory mechanism to place jaxb-api.jar in the
bootstrap
classloader. (See http://java.sun.com/j2se/1.5.0/docs/guide/standards/)
```

  Note: JRE 5 (1.5) doesn't have this problem since it doesn't bundle JAXB.
- JIRA 4.0 uses Lucene v2.3, which is affected by a Sun hotspot compiler bug in JRE 6 (1.6) Update 4 and upwards (see JRA-15681). The bug is fixed in JRE 6 (1.6) Update10.

JIRA Home

JIRA 4.0 has a new directory structure — for details, please see Important Directories and Files.

Please ensure that you set the `jira.home` property as described here.

Recommended Browsers

The following browsers are recommended for use with JIRA 4:

- Internet Explorer 7 and 8
- Firefox 3.x
- Safari 4

Tip: If you are looking for our recommended databases and applications servers, you can find them here:

- Supported Platforms

Users May Encounter 'Advanced Search'

If any of your users have saved invalid filters, the new 'Advanced Search' screen may appear when they try to display them.

'mail.mime.decodeparameters' System Property

The following system property must be set in order for the JIRA mail handler to work correctly with emails from RFC 2231-compliant mail clients:

```
mail.mime.decodeparameters=true
```

Refer to Setting Properties and Options on Startup for instructions.

'Resolution Date' System Field

JIRA 4.0 introduces a new system field, the Resolution Date. This field provides the date when an issue last entered into a 'Resolved' workflow state. When upgrading to JIRA 4.0, an upgrade task will run, calculating the Resolution Date for every resolved issue in your system. If you have a large number of issues, this may take a long time. The speed at which this upgrade task runs can be improved by ensuring that your database statistics are up to date for your changegroup and changeitem tables (to ensure the database will select the most effective query plan).

For example, on Postgres this can be done by executing the following commands:

```
jiratest=# ANALYZE changegroup;
ANALYZE
jiratest=# ANALYZE changeitem;
ANALYZE
```

JIRA's RPC interface now provides two new methods to retrieve an issue's Resolution Date:

- getResolutionDateById(String token, Long issueId) – retrieves the Resolution Date given an issue id
- getResolutionDateByKey(String token, String issueKey) – retrieves the Resolution Date given an issue key

The RemoteIssue class was left unchanged, to ensure backwards compatibility of RPC clients.

Database Schema Changes

If you are using an Oracle or MySQL database, please note that two column data types have been changed.

Therefore, the easiest way to upgrade to JIRA 4.0 is to perform an XML backup and restore as described in the Migrating JIRA to Another Server instructions.

If in the past, instead of performing an XML backup and restore, you have been upgrading by "pointing" the new version of JIRA at an old database, this is still possible. However, the procedure is more complicated. You will need to use SQL scripts to perform database schema changes.

For details (and the scripts), please see JIRA 4.0 Database Schema Changes for MySQL and Oracle .

Charting Plugin

JIRA 4.0 now bundles most of the charts previously provided by the JIRA charting plugin. If you currently have the JIRA charting plugin installed (v1.4.1 or previous) in `WEB-INF/lib`, please remove it as otherwise JIRA will fail to start.

The following charts have not been bundled with JIRA 4.0. If you are using any of the following charts, you will need to upgrade to version 1.5 of the JIRA charting plugin:

- Time to First Response Chart
- Average Number of Times in Status Chart
- Average Time in Status Chart
- Workload Pie Chart Report

JIRA Toolkit

If you are using the JIRA Toolkit, you will need to upgrade it to the latest version.

You will also need to install it in your JIRA home directory, rather than your `atlassian-jira/WEB-INF/lib/` directory as it now runs in an OSGi container. Read Managing JIRA's Plugins for more information.

GreenHopper Plugin

GreenHopper for JIRA 4.0 is now available for use with JIRA 4.0. You can download it on the GreenHopper Plugin for JIRA Downloads page. Please follow the GreenHopper Installation and Upgrade Guide for instructions on how to upgrade GreenHopper.

Please note, you need to **upgrade your GreenHopper license** before you can use GreenHopper with JIRA 4.0. Any existing GreenHopper license files **will not work** with JIRA 4.0. You can obtain a license from http://my.atlassian.com.

Issue 'Status' Field Problem

Prior to JIRA 4.0, it was possible to create two statuses whose names differed only in case (e.g. 'Resolved' and 'RESOLVED'). If you upgrade to JIRA 4.0, this will lead to ambiguities. Consider this scenario:

1. You have defined two issue statuses in a project with names that differ only in case, ('In Progress (Services)' and 'IN PROGRESS (SERVICES)'), to use in different workflows.
2. At a point in time, 100 issues are assigned the first status of 'In Progress (Services)' and 50 issues are assigned the second status of 'IN PROGRESS (SERVICES)'.
3. You browse the project's issues. The 'Status Summary' will incorrectly show only one 'In Progress (Services)' status with either 100 or 50 issues (picked randomly). The issue totals in the other summaries (By Priority, etc) will also be incorrect, due to JIRA not recognising the statuses as distinct.

Additionally, you will receive ambiguous results if you attempt to perform a search by name on the status in the Advanced Search (e.g. "Status = In Progress (Services)").

To resolve this issue, we recommend that you **ensure that each issue status is distinct** by renaming the duplicate statuses appropriately. You may also need to update any issue filters that you have set up.

Plugin Notes

JIRA 4.0 introduces several changes that may break existing plugins.

There are now two different types of plugins. Each type of plugin needs to be installed into a different directory to work. Read Managing JIRA's Plugins for more information.

If you are using a plugin that is not shipped with JIRA, the plugin may need to be updated to work with JIRA 4.0. If the plugin was written by you, please read through Updating JIRA Plugins for JIRA 4.0 and see if any of it is relevant to your plugin. If you are using a plugin written by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.0.

Memory

Running JIRA v4.0 may require more RAM than running v3.x.
The default settings (suitable for small to medium usage) for standalone allocates a total of 512MB memory to the JIRA application.

Please ensure your server has enough available RAM to cover this.
If you are installing JIRA as a WAR/EAR, then you may need to increase the amount of "PermGen" memory allocated to JIRA. 256MB PermGen is recommended.

Upgrading from JIRA 3.12 and earlier

In addition to the points listed above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Please also note the following:

Upgrading from JIRA 2.x Data and earlier

If you are upgrading from JIRA 2.x data (or earlier) to JIRA 4.0, you **must upgrade to any JIRA 3.x release first** (3.13.5 recommended). You can then follow the JIRA 4.0 Upgrade Guide to upgrade your JIRA instance to 4.0.

JIRA 4.0 Database Schema Changes for MySQL and Oracle

If you are using the MySQL or Oracle database, two column data types have been changed as described in the JIRA 4.0 Database Schema Changes for MySQL and Oracle section below.

- Recommended method of upgrade
- Alternative method of upgrade
  - MySQL notes for alternative method of upgrade
  - Oracle notes for alternative method of upgrade
- JIRA 4.0 Database Schema Changes
  - Column data types

Recommended method of upgrade

The easiest way to upgrade to JIRA 4.0 is to perform an **XML backup and restore** as described in the Migrating JIRA to Another Server instructions.

Alternative method of upgrade

If in the past, instead of performing an XML backup and restore, you have been upgrading by "pointing" the new version of JIRA at an old database, this is still possible. However, the procedure is more complicated. You will need to use SQL scripts to perform database schema changes.

MySQL notes for alternative method of upgrade

1. **Shutdown your JIRA instance.**
2. **Perform a backup of your MySQL database as follows:**

   ```
   mysqldump --opt db_name > db_name.sql
   ```

   e.g.: `mysqldump --opt jiradb > jiradb_before4.sql`
3. Download the attached mysql_4_0.sql script.
4. Execute the following script:

   ```
   mysql --user=username --password=password db_name < mysql_4_0.sql
   ```

   e.g.: `mysql --user=root --password=password jiradb < mysql_4_0.sql`
5. If you see any errors, please contact Support for further assistance.
6. Point your new installation of JIRA 4.0 at your MySQL database and watch for any errors in the log during the startup sequence.

Note: The following warnings regarding database changes can be ignored. They will only appear the first time you start JIRA after upgrading your JIRA database to 4.0.

```
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "ExternalGadget" has no table in the
database
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "GadgetUserPreference" has no table in the
database
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "Issue" has 24 fields but table
"jiraissue" has 23 columns.
main WARN     [core.entity.jdbc.DatabaseUtil] Field "resolutiondate" of entity "Issue" is
missing its corresponding column "RESOLUTIONDATE"
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "PortletConfiguration" has 7 fields but
table "portletconfiguration" has 6 columns.
main WARN     [core.entity.jdbc.DatabaseUtil] Field "gadgetXml" of entity
"PortletConfiguration" is missing its corresponding column "GADGET_XML"
```

Oracle notes for alternative method of upgrade

1. **Shutdown your JIRA instance.**
2. **Perform a backup of your Oracle database. There are multiple strategies here, so we will leave this up to your DBA.**
3. Download the attached oracle_4_0.sql script.
4. Connect to SQL*Plus and execute the following script:

```
SQL*Plus: Release 10.2.0.1.0 - Production on Thu Oct 19 12:56:11 2006
Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> @/home/oracle/oracle_4_0.sql
```

5. If you see any errors, please contact Support for further assistance.
6. Point your new installation of JIRA 4.0 at your Oracle database and watch for any errors in the log during the startup sequence.

Note: The following warnings regarding database changes can be ignored. They will only appear the first time you start JIRA after upgrading your JIRA database to 4.0.

```
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "ExternalGadget" has no table in the
database
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "GadgetUserPreference" has no table in the
database
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "Issue" has 24 fields but table
"jiraissue" has 23 columns.
main WARN     [core.entity.jdbc.DatabaseUtil] Field "resolutiondate" of entity "Issue" is
missing its corresponding column "RESOLUTIONDATE"
main WARN     [core.entity.jdbc.DatabaseUtil] Entity "PortletConfiguration" has 7 fields but
table "portletconfiguration" has 6 columns.
main WARN     [core.entity.jdbc.DatabaseUtil] Field "gadgetXml" of entity
"PortletConfiguration" is missing its corresponding column "GADGET_XML"
```

JIRA 4.0 Database Schema Changes

The table below summarises the changes to the database schema. Please note that if you have developed any custom utilities which query or modify the JIRA database directly (i.e. without using the JIRA API), you need to check whether the utilities need to be updated.

Column data types

The following database column data types have been changed. Their column name, old and new data types, as well as the database table they belong to, are shown below:

For MySQL:

TABLE NAME	COLUMN NAME	OLD DATA TYPE	NEW DATA TYPE
propertytext	propertyvalue	TEXT	LONGTEXT
searchrequest	reqcontent	TEXT	LONGTEXT

For Oracle:

TABLE NAME	COLUMN NAME	OLD DATA TYPE	NEW DATA TYPE
propertytext	propertyvalue	VARCHAR2	CLOB
searchrequest	reqcontent	VARCHAR2	CLOB

Updating JIRA Plugins for JIRA 4.0

Plugin Developer Notes

JIRA 4.0 introduces several changes that may break existing plugins. If you are using a plugin that is not shipped with JIRA, the plugin may need to be updated to work with JIRA 4.0. If the plugin was written by you, please read through the information below and see if any of it is relevant to your plugin. If you are using a plugin written by a third party, please check with the plugin's author to see if the plugin has been tested with JIRA 4.0.

- RPC plugin
- Responses from Servlet Plugin Modules are no longer decorated
- Combined JavaScript servlet has been removed
- Project/Component/Version Tab Panel Plugins
- Issue View Plugins
- Issue Tab Panel Plugins
- Search Request View Plugins
- PortalManager and PortalPageConfiguration removed
- New Searching
- Examples
- Plugging into JQL and what happened to my Custom Field Searchers
  - What is a JQL Clause?
  - Integrating with JQL
  - Integrating into the Issue Navigator
  - JIRA 3.x to 4.0 Filter Upgrade
- Converting Portlets to Gadgets

RPC plugin

A number of methods in the RPC plugin were refactored to use services provided by JIRA's core services layer. As a result they may now perform stricter validation on input data, in line with behaviour exhibited by JIRA's UI. A side effect of these changes is that method calls will now throw `RemoteValidationException` instead of `RemotePermissionException` for certain method calls. This change won't break client stubs, since all methods throw `RemoteException` which is the superclass for `RemoteValidationException` and `RemotePermissionException`. However, if client code depends on `RemotePermissionException`, it will need to be updated to expect a `RemoteException` or `RemoteValidationException`.

If you have developed custom code that uses JIRA's SOAP interface, the client code may need to be updated.

Responses from Servlet Plugin Modules are no longer decorated

The response generated by Servlet Plugin Modules served under `/plugins/servlet` will no longer be decorated by SiteMesh. This means that if you are using servlets to display contents directly in the browser, they may be missing the JIRA header and footer. If the response from your servlet needs to be decorated, you have two possible solutions:

1. The best is to convert the servlet to a Webwork Plugin Modules as this is better suited for processing requests that generate HTML responses.
2. Alternatively, add

   ```
   <meta content="decorator_name" name="decorator" />
   ```

   in the `<head>` element of your HTML response, where `decorator_name` is the name of the SiteMesh decorator that should be applied.

Combined JavaScript servlet has been removed

In JIRA 4.0 we cleaned up a lot of the JavaScript resources that are included on every page. As a result, the `combined-javascript` servlet was removed, in favour of Web Resources. This means that if your plugin defines javascript resources of the form:

```
<resource type="javascript">/path/to/my/resource.js</resource>
```

they will no longer be included. They should be replaced by Web Resources.

Project/Component/Version Tab Panel Plugins

The API for this plugin has changed. We removed the action being passed in (what were we thinking) and made it a cleaner, more consistent interface. If you have any custom Tab Panel Plugins plugins, you will need to update them to use the new interface:

```
/**
 * Unified interface for all fragment-based tab panels.
 *
 * @since v4.0
 */
public interface TabPanel<D extends AbstractTabPanelModuleDescriptor, C extends BrowseContext>
{
    /**
     * Initialize the tab panel panel with the plugins ProjectTabPanelModuleDescriptor.  This
is usually used for
     * rendering velocity views.
     *
     * @param descriptor the descriptor for this module as defined in the plugin xml
descriptor.
     */
    void init(D descriptor);

    /**
     * Used to render the tab.
     *
     * @param ctx The current context the tab is rendering in.
     * @return Escaped string with the required html.
     */
    String getHtml(C ctx);

    /**
     * Determine whether or not to show this.
     *
     * @param ctx The current context the tab is rendering in.
     * @return True if the conditions are right to display tab, otherwise false.
     */
    boolean showPanel(C ctx);
}
```

The specific plugin endpoints extend this in the following manner:

```
/**
 * A Tab panel to be displayed on the Browse Project page.
 */
public interface ProjectTabPanel extends TabPanel<ProjectTabPanelModuleDescriptor,
BrowseContext>

/**
 * A Tab panel to be displayed on the Browse Component page.
 */
public interface ComponentTabPanel extends TabPanel<ComponentTabPanelModuleDescriptor,
BrowseComponentContext>

/**
 * A Tab panel to be displayed on the Browse Version page.
 */
public interface VersionTabPanel extends TabPanel<VersionTabPanelModuleDescriptor,
BrowseVersionContext>
```

If you are using `WebResourceManager.requireResource("...")`, your javascript will not be loaded when your tab is loaded via AJAX. You can include it via `WebResourceManager.getStaticPluginResource()` in your actual content. Note: this will be fixed in the next beta.

Issue View Plugins

The `com.atlassian.jira.plugin.issueview.IssueView` interface has changed such that the following methods:

```
public String getContent(Issue issue, IssueViewRequestParams issueViewRequestParams);

public void writeHeaders(Issue issue, RequestHeaders requestHeaders, IssueViewRequestParams
issueViewRequestParams);
```

now take in the `IssueViewRequestParams` parameter. This allows the plugin to access the parameters that were submitted with the request.

If you have written an Issue View plugin, you will need to update it such that in conforms to the new interface.

Issue Tab Panel Plugins

In JIRA 4.0, a new 'sortable' property was introduced to distinguish if the contents of an issue tab panel are sortable. If they are not, the sortable link in the top right corner will not be shown. By default issue tab panels are now **not** sortable. To make a tab panel sortable, plugin developers will have to add the following attribute:

```
<issue-tabpanel key="all-tabpanel" i18n-name-key="admin.issue.tabpanels.plugin.all.name"
name="All Tab Panel" class="com.atlassian.jira.issue.tabpanels.AllTabPanel">
        <description key="admin.issue.tabpanels.plugin.all.desc">Display all tab panels as
one</description>
        <label key="viewissue.tabs.all">All</label>
        <order>0</order>
        <sortable>true</sortable>
    </issue-tabpanel>
```

Search Request View Plugins

In JIRA 4.0, the `com.atlassian.jira.plugin.searchrequestview.SearchRequestView` has the following new method:

```
/**
 * Prints the HTML headers for non-typical HTML such as Word or Excel views. (e.g.:
requestHeaders.addHeader("content-disposition", "attachment;filename="sample.doc";");)
 *
 * @param searchRequest the original search request submitted by the user
 * @param requestHeaders subset of HttpServletResponse responsible for setting headers only
 * @param searchRequestParams context about the current search request
 */
public void writeHeaders(SearchRequest searchRequest, RequestHeaders requestHeaders,
SearchRequestParams searchRequestParams);
```

If you have written a Search Request View Plugin, and the plugin implements the interface without extending `com.atlassian.jira.plugin.searchrequestview.AbstractSearchRequestView`, you will need to update the plugin and implement the new method. The easiest thing to do is to proxy the call straight to the existing method:

```
/**
 * Prints the HTML headers for non-typical HTML such as Word or Excel views. (e.g.:
requestHeaders.addHeader("content-disposition", "attachment;filename="sample.doc";");)
 *
 * @deprecated since v3.13.3 please use {@link
#writeHeaders(com.atlassian.jira.issue.search.SearchRequest, RequestHeaders,
SearchRequestParams)}
 * @param searchRequest the original search request submitted by the user
 * @param requestHeaders subset of HttpServletResponse responsible for setting headers only
 */
public void writeHeaders(SearchRequest searchRequest, RequestHeaders requestHeaders);
```

Note that the `SearchRequestParams` object used by Search Request View Plugins now extends `IssueViewRequestParams` and therefore allows the plugin to access request parameters.

PortalManager and PortalPageConfiguration removed

The deprecated components `PortalManager` and `PortalPageConfiguration` have been removed. Developers should now be using the `JiraDashboardStateStoreManager` to obtain similar functionality.

The `PortalPageConfiguration` had methods that made changes directly to the database (e.g. `store`, `addPortletConfig`, `deletePortletConfig`, `deletePortletConfigs`, `reload`). The `PortalPage` does not have such methods. All persistent changes

must now be made through the `JiraDashboardStateStoreManager` passing the required `DashboardState` as an argument.

The `PortalPageManager & PortalPageService` may also be used to manipulate a `PortalPage` within JIRA. These classes should no longer be used however since they will be re-written or removed for JIRA 4.1.

New Searching

The way a search is performed in JIRA has significantly changed. The introduction of advanced searching (JQL) necessitated a rewrite of the JIRA searching subsystem. In the process, the API for searching has also been changed (and improved) significantly. Unfortunately these changes will almost certainly mean that plugins that search will need to be updated for JIRA 4.0.

In JIRA 3.x and earlier, searching was achieved using a `SearchRequest` in combination with `SearchParameters` and `SearchSorts`. While the `SearchRequest` still continues to exist in JIRA 4.0, the `SearchParameters` have been replaced with the `Query` object.

```
/**
 * The representation of a query.
 *
 */
public interface Query
{
    /**
     * @return the main clause of the search which can be any number of nested clauses that
will make up the full
     * search query. Null indicates that no where clause is available and all issues should be
returned.
     */
    Clause getWhereClause();

    /**
     * @return the sorting portion of the search which can be any number of
     * {@link com.atlassian.query.order.SearchSort}s that will make up the full order by
clause. Null indicates that
     * no order by clause has been entered and we will not sort the query, empty sorts will
cause the default
     * sorts to be used.
     */
    OrderBy getOrderByClause();

    /**
     * @return the original query string that the user inputted into the system. If not
provided, will return null.
     */
    String getQueryString();
}
```

The `Query` is JIRA's internal representation of a JQL search. It contains the search condition (i.e. the "where" clause) and the search order (i.e. the "order by" clause). The `Query` object can be created using the `JqlQueryBuilder`. For example, to create a query "find all issues assigned to either Dylan or Tokes that are unresolved and due in the next week" you would call:

```
final JqlQueryBuilder builder = JqlQueryBuilder.newBuilder();
  builder.where().assignee().in("Dylan",
"Tokes").and().unresolved().and().due().lt().string("+1w");
  builder.orderBy().dueDate(SortOrder.ASC);
  Query query = builder.buildQuery();
```

Once the `Query` has been obtained, it can be used to execute a search. In JIRA 4.0 a new `SearchService` has been added to provide a central location for `Query` related operations. To run the search you can simply call `SearchService.search()` as documented on the `SearchService`. The `SearchProvider` is still available for those who need to control the finer details of searching.

The `Query` object is immutable; once it is created it cannot be changed. The `JqlQueryBuilder` represents the mutable version of a `Query` object. The `JqlQueryBuilder` can be primed with an already existing `Query` by calling `JqlQueryBuilder.newBuilder(existingQuery)`.

In JIRA 3.x the `SearchRequest` was the object that was passed to the searching system to perform a search. The `Query` object has taken over this role in JIRA 4.0; the `SearchProvider` and `SearchService` now take in `Query` objects rather than `SearchRequests`. The `SearchRequest` object has been reworked in JIRA 4.0 to significantly reduce its responsibility. For instance, ordering information is now stored on the `Query` object rather than on the `SearchRequest` object. The `SearchRequest` really represents a saved search (aka. filter). You should only need to deal with `SearchRequests` if you are working with filters. Even in this case, all searching operations need to be performed on `Query` objects by calling `SearchRequest.getQuery()`.

It is often necessary to get a URL for a particular `Query`. The `SearchService` provides the `getQueryString(query)` method for this. The method returns a parameter snippet of the form `jqlQuery=<jqlUrlEncodedQuery>`, which can be appended safely to an existing

URL that points at the Issue Navigator. Note that the links that JIRA 4.0 generates are JQL based, so are incompatible with JIRA 3.x and before. Old valid JIRA 3.x URLs will still work with JIRA 4.0.

Given a `Query` object it is possible to retrieve its JQL representation by calling either `getGeneratedJqlString(query)` or `getJqlString(query)` on the SearchService. The service makes sure that any values in the `Query` that need to be escaped are handled correctly. Importantly, the `Query.toString()` method does not return valid JQL (on purpose).

The `SearchService.parseQuery(jqlString)` method can be used to turn a JQL string into its `Query` representation. The return from this method has details on any parse errors encountered.

A `Query` object, especially those parsed directly from the user, may not be valid. For example, the user may be trying to find issues in a status that does not exist. The `SearchService.validateQuery(query)` method can be used to see if a particular `Query` object is valid. Errors are returned with messages that can be displayed to the user. Executing an invalid `Query` will not result in any errors and in fact may return results. To run an invalid query, JIRA will just make the invalid conditions equate to false and run the query. For example, searching for `status = "I don't Exist" or user = bbain` will result in the query `<false> or user = bbain` actually being run.

There are some methods on the `SearchService` that we did not discuss here. Check out documentation on the `SearchService` for more information.

Examples

Here's a complete example how to obtain search results for the query "project is JRA and the reporter is the currently logged in user and custom field with id 10490 contains 'xss'":

```
String jqlQuery = "project = JRA and reporter = currentUser() and cf[10490] = xss";
        final SearchService.ParseResult parseResult =
                searchService.parseQuery(authenticationContext.getUser(), jqlQuery);
        if (parseResult.isValid())
        {
            try
            {
                final SearchResults results =
searchService.search(authenticationContext.getUser(),
                        parseResult.getQuery(), PagerFilter.getUnlimitedFilter());
                final List<Issue> issues = results.getIssues();

            }
            catch (SearchException e)
            {
                log.error("Error running search", e);
            }
        }
        else
        {
            log.warn("Error parsing jqlQuery: " + parseResult.getErrors());
        }
```

The preceding search could have also been written using the QueryBuilder:

```
final JqlQueryBuilder builder = JqlQueryBuilder.newBuilder();

builder.where().project("JRA").and().reporterIsCurrentUser().and().customField(10490L).eq("xss");
Query query = builder.buildQuery();
        try
        {
            final SearchResults results = searchService.search(authenticationContext.getUser(),
                    query, PagerFilter.getUnlimitedFilter());
            final List<Issue> issues = results.getIssues();

        }
        catch (SearchException e)
        {
            log.error("Error running search", e);
        }
```

Plugging into JQL and what happened to my Custom Field Searchers

The introduction of advanced searching (JQL) necessitated a rewrite of the JIRA searching subsystem. Unfortunately these changes will certainly mean that any `CustomFieldSearchers` will need to be updated to work in 4.0.

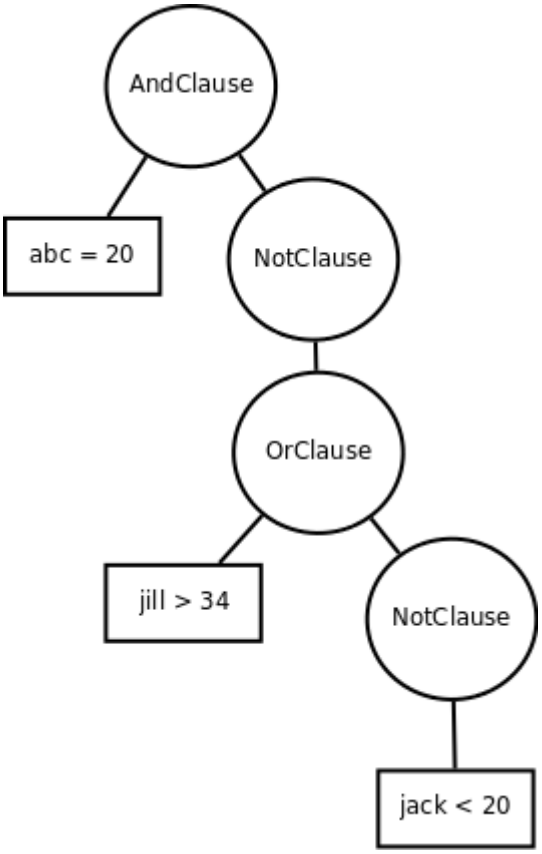
The most fundamental change is that all JIRA 4.0 searching is implemented using JQL. A JQL search consists of two components: firstly, a number of conditions, or `Clauses`, that must be matched for an issue to be returned; and secondly, a collection of search orderings that

define the order in which the issues should be returned. The `Query` object is JIRA's internal representation of a search. It is now the responsibility of the `CustomFieldSearcher` to take a relevant `Query`, validate its correctness and generate a Lucene query to find issues that match it. By doing this your custom field becomes searchable using JQL.

The `CustomFieldSearcher` and/or the custom field is also responsible for ordering results if the order in the search includes the custom field. If your custom field ordered correctly in JIRA 3.x, then it will order correctly in JIRA 4.0. While the internal representation of an order has changed in JIRA 4.0, it still uses the same interfaces to order the search results. We will not address ordering again.

What is a JQL Clause?

A custom field must process the `Clauses` from a JQL search to integrate into JQL. Each `Clause` consists of a number of conditions (e.g. `abc != 20`) combined by the AND and OR logical operators (e.g. `abc = 20 AND (jack < 20 OR jill > 34`). In JIRA a condition is represented by a `TerminalClause`, the logical AND by an `AndClause` and a logical OR by an `OrClause`, all of which implement the `Clause` interface. Finally, the logical NOT operator can be used to negate any other `Clause`. It is represented by a `NotClause` that also implements Clause. These `Clause` objects are composed together to represent a complex conditions. For example, the condition `abc = 20 AND NOT(jill > 34 OR NOT jack < 20)` is represented by the following tree:



A `Clause` can be navigated by passing an instance of a `ClauseVisitor` to the `accept` method of a Clause. This follows the traditional visitor pattern.

The `TerminalClause` represents a `Clause` of the form "field operator value". Inside the `TermincalClause` the "operator" is one of the values from `Operator` enumeration while the "value" is represented as an `Operand`. An `Operand` can represent a single value (e.g. `field = "single"`), a list of values (e.g. `field in ("one", 1235)`), a function (e.g. `field = function(arg1, arg2)`) or EMPTY (e.g. `field is EMPTY`). In the end, all you want is the values from the `Operand`. These can be obtained as a list of `QueryLiteral` (see below) by calling `JqlOperandResolver.getValues()`. The `JqlOperandResolver` also has the `isEmptyOperand`, `isFunctionOperand`, `isListOperand` and `isValidOperand` methods that can be used to determine the type of the `Operand`.

A `QueryLiteral` represents either a `String`, `Long` or `EMPTY` value. These three represent JQL's distinguishable types. It is up to the `CustomFieldSearcher` to convert these values into something that makes sense to it. The type of a `QueryLiteral` can be determined by calling its `isEmpty`, `getLongValue` or `getStringValue` methods. The get methods will return `null` or `false` when the method and the `QueryLiteral` type do not match.

Integrating with JQL

In JIRA 3.x a `CustomFieldSearcher` was the way to provide customised searching functionality for custom fields. In JIRA 4.0 it is still the plugin point for searching; however, the `CustomFieldSearcher` interface has changed significantly to accommodate the introduction of JQL. One of the major changes is that the `CustomFieldSearcher` must return a `CustomFieldSearcherClauseHandler` in JIRA 4.0. This object is a composition of a `ClauseValidator` and a `ClauseQueryFactory`.

The `ClauseValidator` is used by JIRA to ensure that a JQL query is valid according to the `CustomFieldSearcher`.

```
/**
 * Validates a clause and adds human readable i18n'ed messages if there is a problem.
 *
 * @since v4.0
 */
public interface ClauseValidator
{
    /**
     * Validates a clause and adds human readable i18n'ed messages if there is a problem.
     *
     * @param searcher the user who is executing the search.
     * @param terminalClause the clause to validate.
     *
     * @return an MessageSet that will contain any messages relating to failed validation. An
empty message set must
     * be returned to indicate there were no errors. null can never be returned.
     */
    @NotNull
    MessageSet validate(User searcher, @NotNull TerminalClause terminalClause);
}
```

It is up to the validator to ensure that the operator and the value from the passed `TerminalClause` makes sense for the `CustomFieldSearcher` and its associated custom field. Any errors can be placed in the returned `MessageSet`. They should be internationalised with respect to the passed user.

The `validate` method must always return a `MessageSet` as its result. A `null` return is not allowed. A `MessageSet` is an object that contains all of the errors and warnings that occur during validation. All messages in the `MessageSet` need to be translated with respect to the passed searching user. An empty `MessageSet` indicates that no errors have occurred. A `MessageSet` with errors indicates that the JQL is invalid and should not be allowed to run. The returned messages will be displayed to the user so that any problems may be rectified. A `MessageSet` with warnings indicates that the JQL may have problems but that it can still be run. Any warning messages will be displayed above the results.

The `ClauseValidator` does **not** need to check if the passed `TerminalClause` is meant for the for it, JIRA makes sure that it only passes `TerminalClauses` that the `ClauseValidator` is meant to process. It does that by only passing `TerminalClauses` whose "field" matches one of the names the custom field must handle.

`ClauseValidators` need to respect JIRA security. A `ClauseValidator` should not leak information about JIRA objects that the searcher does not have permission to use. For example, a `ClauseValidator` should not differentiate between an object not existing and an object that the user has no permission to see. A `ClauseValidator` that behaves badly will not cause JQL to expose issues that the searcher is not allowed to see (since JQL does permission checks when it runs the filter), though it does open up an attack vector for information disclosure.

The `ClauseValidator` must be thread-safe and re-entrant to ensure correct behavior. JIRA will only create one instance of the `ClauseValidator` per custom field instance. This means that multiple threads may be calling the validator at the same time.

The `ClauseQueryFactory` is used by JIRA to generate the Lucene search for a JQL Clause.

```
public interface ClauseQueryFactory
{
    /**
     * Generates a lucene query for the passed {@link TerminalClause}....
     *
     * @param queryCreationContext the context of the query creation call; used to indicate
that permissions should be
     * ignored for "admin queries"
     * @param terminalClause the clause for which this factory is generating a query.
     * @return QueryFactoryResult contains the query that lucene can use to search and metadata
about the query. Null
     *  cannot be returned.
     */
    @NotNull
    QueryFactoryResult getQuery(@NotNull QueryCreationContext queryCreationContext, @NotNull
TerminalClause terminalClause);
}
```

It is the responsibility of the `ClauseQueryFactory` to create the Lucene search for the passed `TerminalClause` and `QueryCreationContext`. The generated Lucene search is returned in the `QueryFactoryResult`. The result contains the search (a Lucene Query object which is not related the the JQL Query object) and a flag to indicate whether or not the Lucene search should be negated. When set to true, JIRA will actually only match issues that do not match the returned Lucene search. For example, a

ClauseQueryFactory may decide to implement a condition like field != value by returning a Lucene search that matches field = value and setting the flag to true. You can also implement this condition by returning a Lucene search that matches field != value and setting the flag to false.

The new argument here is the QueryCreationContext. This object contains the variables that may be necessary when creating the query. The QueryCreationContext.getUser method returns the user that is running the search and as such should be used to perform any security checks that may be necessary. The QueryCreationContext.isSecurityOverriden method indicates whether or not this function should actually perform security checks. When it returns true, the factory should assume that the searcher has permission to see everything in JIRA. When it returns false, the factory should perform regular security checks.

A ClauseQueryFactory should try to limit the queries so that issues that the user cannot see are excluded. Consider the query affectsVersion = "1.0". The ClauseQueryFactory might detect that there are two versions named "1.0", one from project1 and the other from project2. The factory might then notice that the user doing the search cannot see project1. The factory can then return a query that contains only the version from project2. This is mainly an efficiency concern as JIRA filters all search results to ensure users cannot see issues they are not allowed to.

The ClauseQueryFactory does **not** need to check if the passed ClauseQueryFactory is meant for it; JIRA makes sure that it only passes TerminalClauses that the ClauseQueryFactory is meant to process. It does that by only passing TerminalClauses whose "field" matches one of the JQL names the custom field must handle. Put simply, the ClauseQueryFactory must handle any passed TerminalClause.

The ClauseQueryFactory must also handle the situation when an invalid TerminalClause is passed to it. An invalid TerminalClause is one whose associated ClauseValidator would not validate. The ClauseQueryFactory must return an empty Lucene search if the passed TerminalClause is invalid. Most importantly, the ClauseQueryFactory must not throw an exception on an invalid TerminalClause.

A ClauseQueryFactory needs to be careful when implementing any of the negating operators (i.e. !=, !~, "not in"). These operators should not match what is considered empty by the custom field and CustomFieldSearcher. For example, the JQL query resolution is EMPTY will return all unresolved issues in JIRA. The query resolution != fixed will only return all resolved issues that have not been resolved as "fixed", that is, it will not return any unresolved issues. The user has to enter the query resolution != fixed or resolution is EMPTY to find all issues that are either unresolved or not resolved as "fixed".

A ClauseQueryFactory also needs to consider field visibility. A CustomFieldSearcher should not match any issues where its associated custom field is not visible. Importantly, asking for EMPTY should not match issues where the custom field is not visible. For example, the JQL query resolution is EMPTY will not return issues from a project whose resolution field has been hidden. A hidden field is assumed not to exist.

There are some extra interfaces that the CustomFieldSearcherClauseHandler may also implement to provide optional functionality to the searching subsystem:

- ValueGeneratingClauseHandler: Gives the CustomFieldSearcher the ability to suggest some values during JQL entry auto-complete. This is really only useful for custom fields whose values come from an allowable finite set.
- CustomFieldClauseSanitiserHandler: Gives the CustomFieldSearcher the ability pre-process the query and remove sensitive information from the query before it is displayed to the passed user.
- CustomFieldClauseContextHandler: Gives the CustomFieldSearcher the ability to customise JIRA's query context calculation. This interface is best left alone, unexplained and unimplemented.

Integrating into the Issue Navigator

The good old Issue Navigator still exists. The Issue Navigator actually has two modes: simple and advanced. The simple mode is what was considered the Issue Navigator in JIRA 3.x. Each searcher on the simple Issue Navigator represents a Clause. For example, selecting "JIRA" in the project searcher produces the Clause project = JIRA. Using multiple searchers is achieved by ANDing the multiple implied Clauses together. In this way the simple Issue Navigator actually generates JQL.

The advanced mode shows the raw JQL to the user. It allows a user to search by entering arbitrary JQL. Since it simply shows JQL, it is possible to create a query using the simple Issue Navigator and then view it in the advanced Issue Navigator. However, it may not always be possible to go from the advanced Issue Navigator to the simple Issue Navigator, as the simple view only allows a very limited set of JQL. A JIRA user will be able to move from the advanced to the simple Issue Navigator when the current JQL can be represented in the simple view. JIRA will stop a user from transitioning from the advanced to the simple Issue Navigator when the JQL is just too complicated to represent correctly.

The CustomFieldSearcher itself is still responsible for integrating into the Issue Navigator. The CustomFieldSearcher extends from the IssueSearcher, which has undergone major cosmetic surgery in JIRA 4.0. The main change is that the methods on the IssueSearcher have been relocated to new interfaces that the IssueSearcher composes. For example, JIRA 3.x used to call issueSeacher.getEditHtml() to get the searcher's HTML but now in 4.0 it calls issueSeacher.getSearchRenderer().getEditHtml(). The following table shows a summary of all the changes:

Old Searcher Method	New Seacher Interface	New Seacher Method	Comments
getEditHtml	SearchRenderer	getEditHtml	Inserted a new User parameter as the first argument.

getViewHtml	SearchRenderer	getViewHtml	Inserted a new User parameter as the first argument.
isShown	SearchRenderer	isShown	Inserted a new User parameter as the first argument.
isRelevantForSearchRequest	SearchRenderer	isRelevantForQuery	Inserted a new User parameter as the first argument. See description below.
getId	SearcherInformation	getId	
getNameKey	SearcherInformation	getNameKey	
	SearcherInformation	getField	Added in JIRA 4.0. See description below.
getRelatedIndexers	SearcherInformation	getRelatedIndexers	
	SearcherInformation	getSearcherGroupType	Added in JIRA 4.0. See description below.
populateFromParams	SearchInputTransformer	populateFromParams	Inserted a new User parameter as the first argument.
validateParams	SearchInputTransformer	validateParams	Inserted a new User parameter as the first argument.
populateFromSearchRequest	SearchInputTransformer	populateFromQuery	Changed the method name and arguments to work with JQL. See the discussion below.
	SearchInputTransformer	doRelevantClausesFitFilterForm	Added in JIRA 4.0. See discussion below.
populateSearchRequest	SearchInputTransformer	getSearchClause	Changed the method name and arguments to work with JQL. See the discussion below.
register			Removed as it is no longer necessary.
getQuerySnippet			Removed as it is no longer necessary.
getStringValue			Removed as it is no longer necessary.
getName			Removed as it is no longer necessary.

The SearcherRender interface groups together the rendering related IssueSearcher actions. The new method isRelevantForQuery takes over the role from the isRelevantForSearchRequest method. Its job is to take a complete Query object and determine if the CustomFieldSearcher is relevant for that Query. The result is used to decide if the HTML from the getViewHtml is included on some JIRA pages. As a general rule, this essentially involves walking the Query and looking for TerminalClauses related to the CustomFieldSearcher. For example:

```
final NamedTerminalClauseCollectingVisitor clauseVisitor = new
NamedTerminalClauseCollectingVisitor(clauseNames.getJqlFieldNames());
if (query != null && query.getWhereClause() != null)
{
    query.getWhereClause().accept(clauseVisitor);
}
return clauseVisitor.containsNamedClause();
```

This code essentially walks the tree looking for all `TerminalClauses` that have a particular set of names. The `Query` is relevant if such a `Clause` exists or is not relevant otherwise.

The `isRelevantForQuery` method is only called if the passed Query fits in the simple Issue Navigator.

The `SearcherInformation` interface groups together methods that return data about the `IssueSeacher` into a single interface. The `SearcherInformation.getField` method simply returns the `Field` associated with the searcher. This information is available to the searcher once the `CustomFieldSearcher.init()` is called by JIRA.

`SearcherInformation.getSearcherGroupType` is a method that returns the group the searcher should be seen in on the navigator. The custom field has to return `SearcherGroupType.CUSTOM`. JIRA will always force this value even if it is specified as something different.

The `SearchInputTransformer` interface groups together those methods on the `IssueSearcher` that convert `Query` objects into different forms so that they can be displayed and manipulated using the simple Issue Navigator. The simple Issue Navigator does not have the ability to represent all possible JQL queries. The `SearchInputTransformer.doRelevantClausesFitFilterForm` method allows JIRA to ask the `CustomFieldSearcher` if the passed `Query` can be represented in the simple Issue Navigator. This is used by JIRA to stop people trying to view complex JQL in the simple Issue Navigator. When this call is made, the `CustomFieldSearcher` must decide if the relevant sections of the passed `Query` can be represented in the simple Issue Navigator form. Irrelevant `Clauses` (i.e. those `Clauses` unrelated to the Searcher) should be ignored. The method must return true when the `Query` is not at all relevant. This method is normally implemented by walking the `Query` and checking that any relevant `TerminalClauses` are connected via the correct set of logical conditions. For example, here is some common code encountered with JIRA's internal searchers:

```
if (query != null && query.getWhereClause() != null)
{
    final Clause whereClause = query.getWhereClause();
    final SimpleNavigatorCollectorVisitor collector = new
SimpleNavigatorCollectorVisitor(clauseNames.getJqlFieldNames());
    whereClause.accept(collector);
    if (!collector.isValid() || collector.getClauses().size() > 1)
    {
        return false;
    }
    else if (collector.getClauses().size() == 1)
    {
        final TerminalClause terminalClause = collector.getClauses().get(0);
        return checkOperator(terminalClause.getOperator()) &&
checkOperand(terminalClause.getOperand(), true);
    }
}
return true;
```

The code starts by creating a `ClauseVisitor` that will find all the `TerminalClauses` with particular names. This visitor will also detect whether or not the all paths from the root `Clause` of the tree to the `TerminalClauses` are only through `AndClauses`. This check is made to ensure that these `TerminalClauses` form part of a simple AND expression since the simple Issue Navigator can only support AND operators between Clauses. The code also ensures that only one `TerminalClause` is found since this is what the `CustomFieldSearcher` generates for the simple Issue Navigator. Note that the method will return true if no relevant `TerminalClauses` are found.

The new `SearchInputTransformer.populateFromQuery` method replaces the old `populateFromSearchRequest`. It essentially takes the passed `Query` and serialises the relevant parts into their associated `FieldValuesHolder` representation. It is up to this `CustomFieldSearcher` to work out which parts of the `Query` are relevant to it. It must ignore those parts of the `Query` that it was not designed to handle. This method will only be called if it is known that the `Query` fits in into the simple Issue Navigator. It is generally implemented by walking the tree and looking for the relevant `TerminalClauses` and subsequently serialising them into the passed `FieldValuesHoldler`. For example:

```
if (query.getWhereClause() != null)
{
    final ClauseVisitor visitor = new DateSerializer();
    query.getWhereClause().accept(visitor);
    fieldValuesHolder.put(dateSeacherConfig.getPreviousField(), visitor.getPreviousDate());
    fieldValuesHolder.put(dateSeacherConfig.getNextField(), visitor.getPreviousDate());
}
```

In this example we used a `ClauseVisitor` that walks the `Query` and calculates the parameters for a date-based searcher. Once the visitor is run, we simply add the calculated parameters to the `FieldValuesHolder`.

The `SearchInputTransformer.getSearchClause` method replaces the old `populateSearchRequest`. Its job it to take a take the relevant values from the `FieldValuesHolder` and generate a `Clause` for them. This `Clause` will be combined with the `Clauses` from other active searchers using the AND operator to produce the final Query on the simple Issue Navigator. The irrelevant values from the `FieldValueHolder` must be ignored. This method is generally called after JIRA has called `SearchInputTransformer.populateFromParams` with the web parameters returned from the filter form, that is, this method is how the filter form is converted into a `Clause` and subsequently a Query. Consider the following example:

```
final Clause relativeClause = createPeriodClause((String)
fieldValuesHolder.get(dateSearcherConfig.getPreviousField()),
    (String) fieldValuesHolder.get(dateSearcherConfig.getNextField()));
final Clause absoluteClause = createDateClause((String)
fieldValuesHolder.get(dateSearcherConfig.getAfterField()),
    (String) fieldValuesHolder.get(dateSearcherConfig.getBeforeField()));
return createCompoundClause(relativeClause, absoluteClause);
```

This example demonstrates how a date field looks in the `FieldValueHolder` for its relevant properties and uses them to create a `Clause`. This example also shows that the returned `Clause` can be as complex as the `CustomFieldSearcher` wants.

The `SearchInputTransformer.getSearchClause` and `SearchInputTransformer.populateFromQuery` really form a pair. The `Clause` returned from `SearchInputTransformer.getSearchClause` must be correctly processed by `SearchInputTransformer.populateFromQuery`. If this does not occur, then it would be possible to generate a query in the simple Issue Navigator view that cannot actually be viewed in it. This also implies passing the `Clause` object returned from `SearchInputTransformer.getSearchClause` to the `SearchInputTransformer.doRelevantClausesFitFilterForm` must return true.

JIRA 3.x to 4.0 Filter Upgrade

In JIRA 3.x saved seaches (aka. filters) were stored in the database as XML. In JIRA 4.0, all searchers are stored directly as JQL. An upgrade task has been written to convert 3.x filters into JQL. Unfortunately, there is no way for plugin developers to integrate into this upgrade task. This essentially means that the upgrade may fail if you have a custom `SearchParameter` or use an existing `SearchParameter` in an unorthodox way. JIRA will inform users through e-mail if any of their filters could not be upgraded cleanly. The administrator is also made aware of any problems through JIRA's log files.

Converting Portlets to Gadgets

JIRA 4.0 introduces a new dashboard based on the OpenSocial specification. Legacy portlets will still be supported, but they will miss out on a lot of new features (e.g. displaying the gadget on iGoogle). As such you may wish to convert your plugin's portlets to gadgets. To do so please follow the documentation available in the Gadget Development Hub, as well as the instructions for writing a plugin upgrade task to convert any portlet settings that users may have saved.

Upgrading JIRA 2.x Data to JIRA 4.0

If you are upgrading from JIRA 2.x data (or earlier) to JIRA 4.0, you **must upgrade to any JIRA 3.x release first** (3.13.5 recommended). You can then follow the JIRA 4.0 Upgrade Guide to upgrade your JIRA instance to 4.0.

Writing a Plugin Upgrade Task for JIRA 4.0

Overview

JIRA 4.0 will introduce a new dashboard, effectively making the Portlet Plugin Module obsolete. Legacy portlets will still be supported via a Legacy Gadget bridge; however, they will miss out on a lot of the new features that gadgets offer (e.g. the ability to share gadgets with other apps such as iGoogle). It therefore makes sense to convert portlets over to gadgets. Information about how to write a gadget can be found in the Gadget Development Hub, and specifically the page about gadgets and JIRA portlets.

If you've converted a portlet to a gadget, you will most likely need an upgrade task to convert existing data of your users into the new format used by the gadget you have written. This page describes the process of creating such an upgrade task.

Why an upgrade task?

Portlets generally have some configuration data associated with them by their users. For example, the First Response Time chart portlet,

available in the Charting Plugin, allows users to configure how many days previous to draw the chart for (among other things). For efficiency reasons, gadgets do not use the same storage mechanism as portlets do to store these user preferences. An upgrade task is thus needed to convert existing user data over to the new format required by the new gadget.

Upgrade framework

JIRA 4.0 introduces a new plugin framework (version 2.2 or later of the Atlassian Plugin Framework, affectionately known as 'Plugins2'), which provides an events system that lets plugins register to listen for certain events (such as a 'Framework started' event). JIRA 4.0 also bundles SAL, which already includes a plugin upgrade framework. SAL provides a plugin upgrade manager that listens for the 'Framework started' event and will look for Plugin Upgrade Tasks to run in order to upgrade data for plugins.

What does all this mean? Effectively, plugin writers don't have to worry about providing an upgrade task framework. They can simply provide a Plugin Upgrade Task component and SAL will guarantee that their upgrade task is run on startup.

Example

Let's look at what needs to be done to run an upgrade task to convert the First Response Time chart portlet data over to gadget data.

1. Convert your Portlet to a Gadget

Please follow the documentation available in the Gadget Development Hub for this step, and specifically the page about gadgets and JIRA portlets.

2. Add dependency on SAL

First we'll need access to the SAL API in the charting plugin project. Add the following dependency to the plugin's `pom.xml`:

pom.xml

```
...
  <dependency>
     <groupId>com.atlassian.sal</groupId>
     <artifactId>sal-api</artifactId>
     <version>2.0.17</version>
     <scope>provided</scope>
  </dependency>
...
```

Re-generate your IDE's project descriptor (`mvn idea:idea` or `mvn eclipse:eclipse`) after this step to allow you to access the new SAL API classes in your project.

3. Convert your plugin to Plugins2

SAL is a Plugins2 bundle and your plugin will have to be converted to the Plugins2 format first before you can write an upgrade task that will be picked up by the PluginUpgradeManager. Gadgets are also only supported in Plugins2 bundles.

There are generic instructions available for how to do this, but let's look specifically at the Charting plugin example. The only thing that is needed is to add the **`plugins-version="2"`** attribute in `atlassian-plugins.xml`:

atlassian-plugins.xml

```
<atlassian-plugin key="${atlassian.plugin.key}" name="${project.name}" system="true"
plugins-version="2">
...
```

4. Writing your upgrade task

Now that all the prerequisites are done, the Upgrade task for the plugin can be written. This class simply needs to implement the PluginUpgradeTask interface provided by SAL.

Here's an example implementation:

GadgetUpgradeTask.java

```
package com.atlassian.jira.ext.charting.upgrade;

import com.atlassian.configurable.ObjectConfigurationException;
import com.atlassian.gadgets.dashboard.Color;
import com.atlassian.jira.ComponentManager;
import com.atlassian.jira.portal.OfbizPortletConfigurationStore;
import com.atlassian.jira.portal.PortletConfiguration;
import com.atlassian.jira.portal.PortletConfigurationImpl;
```

```
import com.atlassian.jira.portal.PortletConfigurationStore;
import com.atlassian.jira.propertyset.JiraPropertySetFactory;
import com.atlassian.jira.upgrade.util.SimpleLegacyPortletUpgradeTask;
import com.atlassian.jira.util.Consumer;
import com.atlassian.jira.util.NotNull;
import com.atlassian.jira.util.collect.EnclosedIterable;
import com.atlassian.sal.api.message.Message;
import com.atlassian.sal.api.upgrade.PluginUpgradeTask;
import com.opensymphony.module.propertyset.PropertySet;
import org.apache.log4j.Logger;

import java.net.URI;
import java.util.Collection;
import java.util.Map;

public class GadgetUpgradeTask implements PluginUpgradeTask
{
    private static final Logger log = Logger.getLogger(GadgetUpgradeTask.class);

    private final PortletConfigurationStore portletConfigurationStore;
    private final JiraPropertySetFactory propertySetFactory;

    public GadgetUpgradeTask(JiraPropertySetFactory propertySetFactory)
    {
        //NOTE: Can't get the portletConfigStore injected here since it is not made available
to plugins2
        this.portletConfigurationStore =
ComponentManager.getComponentInstanceOfType(PortletConfigurationStore.class);
        this.propertySetFactory = propertySetFactory;
    }

    /**
     * The build number for this upgrade task. Once this upgrade task has run the plugin
manager will store this
     * build number against this plugin type.  After this only upgrade tasks with higher build
numbers will be run
     */
    public int getBuildNumber()
    {
        return 1;
    }

    public String getShortDescription()
    {
        return "Upgrades legacy portlet configuration to new gadget user prefs.";
    }

    public Collection<Message> doUpgrade() throws Exception
    {
        final SimpleLegacyPortletUpgradeTask upgradeTask =
                new
SimpleLegacyPortletUpgradeTask("com.atlassian.jira.ext.charting:firstresponsetime",

URI.create("rest/gadgets/1.0/g/com.atlassian.jira.ext.charting:firstresponsetime/firstresponsetime.
//First get all the portletConfigurations in the database.
        final EnclosedIterable<PortletConfiguration> iterable =
portletConfigurationStore.getAllPortletConfigurations();
        iterable.foreach(new Consumer<PortletConfiguration>()
        {
            public void consume(@NotNull final PortletConfiguration pc)
            {
                //for each portletconfiguration, check if it's key matches the portlet key we
want to upgrade
                if (pc.getKey() != null && pc.getKey().startsWith(upgradeTask.getPortletKey()))
                {
                    log.info("Upgrading portletconfig with id '" + pc.getId() + "'");
                    //first lets convert the preferences for this portlet to the new prefs
format used for gadgets.
                    final Map<String, String> prefs;
                    try
                    {
                        prefs = upgradeTask.convertUserPrefs(pc.getProperties());
                    }
                    catch (ObjectConfigurationException e)
                    {
```

```
                        throw new RuntimeException(e);
                    }
                    //then create essentially a copy of the old portletConfig.  This new copy
no longer needs to have
                    //the portletKey and propertySet set to any values.  It however does
require the GadgetUri and user prefs to be set.
                    final PortletConfiguration newConfig =
                            new PortletConfigurationImpl(pc.getId(), pc.getDashboardPageId(),
null, null, pc.getColumn(), pc.getRow(),
                                    null, upgradeTask.getGadgetUri(), Color.color8, prefs);
                    //Now lets store this new config back to the database.
                    portletConfigurationStore.store(newConfig);
                    //clear out the old properties for this portlet
                    removePropertySet(pc);
                }
            }
        });

        return null;
    }

    private void removePropertySet(final PortletConfiguration pc)
    {
        final PropertySet livePropertySet =
propertySetFactory.buildNoncachingPropertySet(OfbizPortletConfigurationStore.TABLE,
pc.getId());
        @SuppressWarnings ("unchecked")
        final Collection<String> keys = livePropertySet.getKeys();
        for (String propertyKey : keys)
        {
            livePropertySet.remove(propertyKey);
        }
    }

    /**
     * Identifies the plugin that will be upgraded.
     */
    public String getPluginKey()
    {
```

```
            return "com.atlassian.jira.ext.charting";
        }
    }
```

There are a few things to note about this implementation:

- `getBuildNumber()` and `getPluginKey()` determine if this upgrade task will run. `getPluginKey()` needs to match the key of the plugin that is being upgraded (in this case the charting plugin). `getBuildNumber()` returns the buildnumber for this upgrade task. `'1'` will do for any plugin that hasn't had any upgrade tasks run against it yet. SAL's PluginUpgradeManager will run this upgrade task and store the buildnumber against the plugin once completed. After this, **only** upgrade tasks with a higher build number than '1' will be executed.
- `doUpgrade()` uses some helpers provided by JIRA (i.e. the SimpleLegacyPortletUpgradeTask) to convert the legacy portlet to a gadget. This is entirely optional, however, and plugin authors are free to implement this method however they like.

Please ensure that the plugin upgrade task **ONLY** upgrades `portletConfigurations` for the plugin that's being upgraded! Any other `portletConfigurations` MUST be left untouched, as otherwise there's a risk of clobbering other portlets' data!

5. Register the upgrade task

Now we simply need to register the upgrade task as a component in the plugin:

atlassian-plugin.xml

```
...
 <component key="gadgetUpgradeTask" name="Gadget Upgrade Task"
 class="com.atlassian.jira.ext.charting.upgrade.GadgetUpgradeTask" public="true">
     <interface>com.atlassian.sal.api.upgrade.PluginUpgradeTask</interface>
 </component>
...
```

The PluginUpgradeManager in SAL will automatically scan for components that implement the `PluginUpgradeTask` interface. Please note that they have to be declared as **`public="true"`**.

That's it. Simply re-package the plugin, deploy it to the instance of JIRA to upgrade and restart the JIRA instance. The plugin upgrade task should be executed when JIRA starts up.

It's highly recommended that you perform a backup of your JIRA instance before attempting this!

JIRA 4.0.2 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

26 February 2010

The Atlassian JIRA team is proud to announce the release of **JIRA 4.0.2**. This point release contains over 40 bug fixes and improvements, notably including:

- The gadget resizing issue.
- Greatly improved stability of JIRA gadgets in Confluence.
- Support for WebLogic 9.2.

JIRA 4.0.2 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4 yet?
Take a look at all the new features in the JIRA 4.0 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.0.2 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.0.2 includes the following updates and bug fixes:

JIRA Issues (37 issues)				
Type	**Key**	**Summary**	**Priority**	**Status**
	JRA-20681	wiki documentation: Dashboard JIRA 4.0 ... Configuring JIRA Standalone		Closed
	JRA-20503	Improvement in "deactivate user' documentation		Closed
	JRA-20437	Only System Administrators can add gadgets to the directory		Closed
	JRA-20136	Add a note about deleting the work directory when upgrading the war distribution from a previous installation of JIRA		Closed
	JRA-19926	Doc: Installing JIRA on websphere 6.1 doc needs updating		Closed
	JRA-19862	Add "/rest" to default URLs on screen for adding a new trusted application		Resolved
	JRA-20038	Pcounter value not kept correctly between project description changes		Resolved
	JRA-19940	Fogbugz importer is not compatible with Fogbugz 7.0 release		Resolved
	JRA-19919	JQL function "linkedIssues" is case-sensitive or not depending on the DB		Resolved
	JRA-19916	Gadget is resizing at oauth button screen in IE and Safari causing the button to be obscured		Resolved
	JRA-19861	If you use a screen in a workflow draft (but not a workflow), you are able to delete the screen which makes the Administration --> Workflow screen impossible to access.		Resolved
	JRA-19856	WordCurlyQuotesRequestWrapper breaks OAuth signing process		Resolved
	JRA-19820	Unix shell startup script doen't set PermGen option for Sun JVM		Resolved
	JRA-19713	Startup Failure on Weblogic: Plugin Event Listener 'com.atlassian.jirafisheyeplugin.rest.fisheye.FishEyeRestApiManagerImpl@1fecc91' threw an error on event 'com.atlassian.plugin.event.events.PluginFrameworkShutdownEvent@2f179cc': erroneous handlers		Resolved
	JRA-19367	JIRA v4.0.0 does not work on Weblogic Server		Resolved
	JRA-20223	LDAP Configurer doesn't use new JIRA-specific Profile and Access providers		Resolved
	JRA-19966	Most JIRA gadgets cannot be configured on Dashboard in IE7 due to resizing bug		Resolved
	JRA-19899	"Edit" and "OAuth" icon is not properly visible when user customize "Assigned to me" JIRA gadget in Confluence		Resolved
	JRA-19886	Dashboard gadgets ajax requests are cached by IE and thus sometimes shows outdated data		Resolved
	JRA-19764	The OSUser User object is not serializable, and this can cause JIRA users to get logged out.		Resolved
	JRA-19753	addFreeTextCondition() assumes environment field is visible to user performing the query		Resolved
	JRA-19726	System Error when setting up JIRA for the first time with Profiling Filter set to auto start		Resolved
	JRA-19699	NumberFormatException thrown when trying to import project with no assigneetype attribute set		

				Resolved
	JRA-19627	Switching between tabs after editing a field on IE causes the "Are you sure you want to navigate away" popup dialog box to appear every time		Resolved
	JRA-19520	Issue not displaying in full in I.E. 7		Resolved
	JRA-19473	Australian English in the US English language pack		Resolved
	JRA-19351	Translating JIRA doco page is outdated		Closed
	JRA-21150	Soap deleteProject call may try to delete an issue more than one time causing it to fail with an Exception		Resolved
	JRA-20376	Unresolved png (!workflow-publishdraft.png!) in online doc		Closed
	JRA-20128	Include XSS security warning on HTML macro description in Wiki Markup Renderer		Resolved
	JRA-19988	JIRA keyboard shortcuts incorrect in documentation		Closed
	JRA-19852	The package scanner used for OSGi plugins can extract incorrect version numbers when there is a number in the library name.		Resolved
	JRA-19836	Broken link in jira standalone windows installer		Resolved
	JRA-19817	Default permission scheme -- not selected as default on project creation		Resolved
	JRA-19801	Czech translation typo in issue view page (prolému)		Resolved
	JRA-19605	Gadget iframe height sometimes doesnt expand enough to fit entire gadget contents		Resolved
	JRA-10934	'Assign to me' shortcut on bulk edit page doesn't autotick the 'Change assign to' checkbox		Resolved

JIRA 4.0.2 Upgrade Guide

Please note the following before performing this upgrade:

> **Please test your business-critical gadgets**
>
> As with any new and rapidly evolving technology, gadgets offer exciting opportunities — and potential technology incompatibilities. If your cross-server gadgets are working successfully with JIRA 4.0, and you rely on these gadgets for business purposes, please test them with JIRA 4.0.2 on a non-production server before upgrading.

Upgrading from JIRA 4.0 or 4.0.1 to 4.0.2

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.13.x and earlier

In addition to the above, please read the JIRA 4.0 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 4.0.1 Release Notes

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

8 December 2009

The Atlassian JIRA team is proud to announce the release of **JIRA 4.0.1**. This point release contains over **60** bug fixes and improvements,

notably including the gadget loopback issue. We are also very pleased to announce support for WebSphere 6.1.0.27.

JIRA 4.0.1 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 4 yet?
Take a look at all the new features in the JIRA 4.0 Release Notes and see what you are missing out on!

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 4.0.1 Upgrade Guide.

Updates and Fixes in this Release

JIRA 4.0.1 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 4.0.1 Upgrade Guide

Upgrading from JIRA 4.0 to 4.0.1

Please note the following before performing this upgrade:

WebLogic 9.2 not supported

Please note that the WebLogic 9.2 patch issued for JIRA 4.0 in JRA-19367 may not work with JIRA 4.0.1 in all environments. We are addressing WebLogic 9.2 support and will provide a solution as soon as possible. If you are using WebLogic, please stay on JIRA 3.13.x in the interim.

Websphere version is important

WebSphere 6.1.0.27 is supported with JIRA 4.0.1. The version of WebSphere is important as we have seen issues running with WebSphere 6.1.0.3. Please see JRA-19421 for details. WebSphere 7.0 is not supported

Gadgets served to other servers may experience issues

As a result of fixing the 'loopback' problem (where a server had problems serving a gadget to itself), issues such as JRA-19890 may now be encountered when serving JIRA gadgets to iGoogle. As with any new and rapidly evolving technology, gadgets offer exciting opportunities — and potential technology incompatibilities. If your cross-server gadgets are working successfully with JIRA 4.0, and you rely on these gadgets for business purposes, please test them with JIRA 4.0.1 on a non-production server before upgrading.

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.13.x and earlier

In addition to the above (particularly regarding Websphere and WebLogic), please read the JIRA 4.0 Upgrade Guide and the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.13 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

9 September 2008

The Atlassian JIRA team is proud to present **JIRA 3.13**.

This release fulfils some of the most popular JIRA feature requests. Dashboards can now be shared, and filter sharing has been improved —

so it's easy to set up multiple 'template' dashboards, each with specific portlets and filters. New JIRA users can then simply select the dashboards most suited to them.

People using customised workflows will be pleased to learn that JIRA 3.13 provides the ability to edit active workflows — that is, workflows that are currently being used. So workflow logic, transitions, screens and post-functions can now be tweaked on the fly, and a JIRA wizard will guide decisions on how active issues should be handled.

By popular request, you can now restore individual projects from a backup, making it much easier to merge projects back into your existing JIRA instance. We are also happy to announce that personal licenses are available with this release of JIRA.

Please be aware of JIRA Security Advisory 2008-08-26, which is relevant to this release.

Upgrading to JIRA 3.13 is free for all customers with active JIRA software maintenance.

Highlights of JIRA 3.13

- Shareable dashboards
- Improved filter sharing
- Favourite filters and dashboards
- Restoring projects
- Editable active workflows
- Enhanced sub-task quick creation
- Personal licenses
- Plugins
- Progress bar for long-running operations
- Application improvements
- Plus more than **200** other fixes and improvements

Thank you for your feedback

85 new feature and improvement requests implemented!

1485 votes fulfilled!

Your votes and issues help us keep improving our products, and are much appreciated.

DOWNLOAD latest version

Upgrading to JIRA 3.13
You can download JIRA 3.13 from the JIRA Download Center. Before upgrading, please refer to the JIRA 3.13 Upgrade Guide.

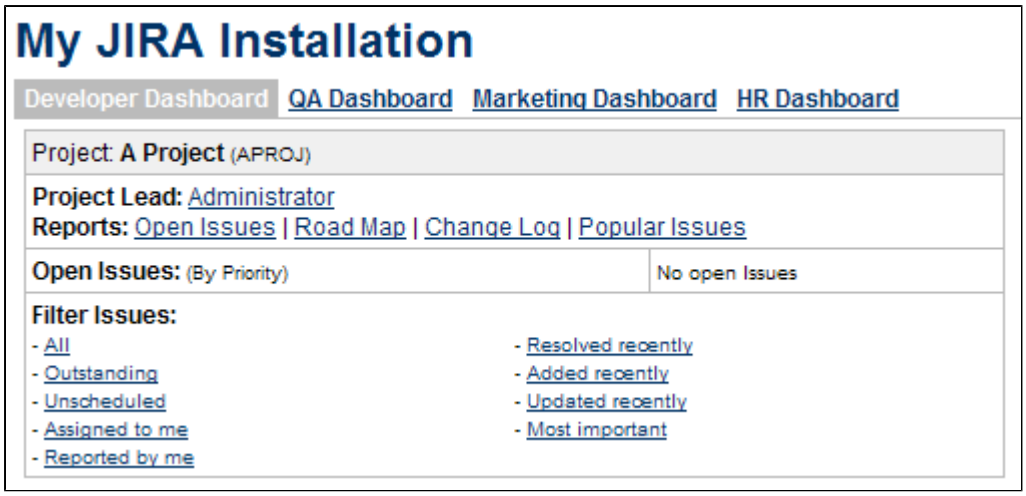
Highlights of JIRA 3.13

1

Shareable dashboards

In many organisations, different departments use different JIRA projects and so need a customised dashboard for people in each departmen to use. For instance, at Atlassian our HR team uses their own special 'Recruitment' project and workflow to manage the hiring process. JIRA 3.13 resolves this popular issue (434 votes!) by letting you set up dashboard pages that you can share with your user group, project or all JIRA users.

- Read more about shareable dashboards in the dashboard pages documentation.



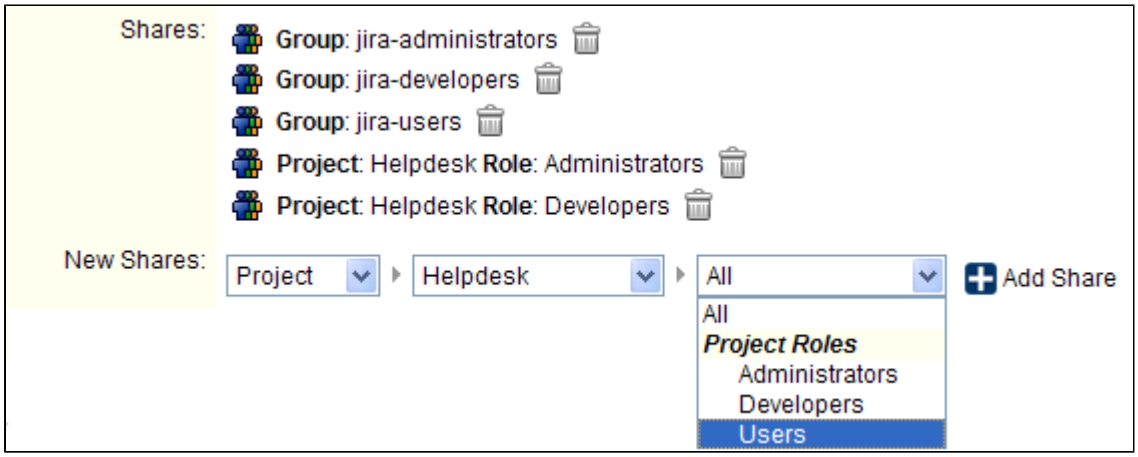
^Top

2

Improved filter sharing

Previous versions of JIRA allowed you to share filters, but filter sharing was restricted to sharing with a single user group or sharing with all JIRA users. Filter sharing is much more flexible in this release. You can now also share your filter with multiple user groups, projects, specific project roles or any combination of these.

- Read more about shareable filters in the issue filters documentation.



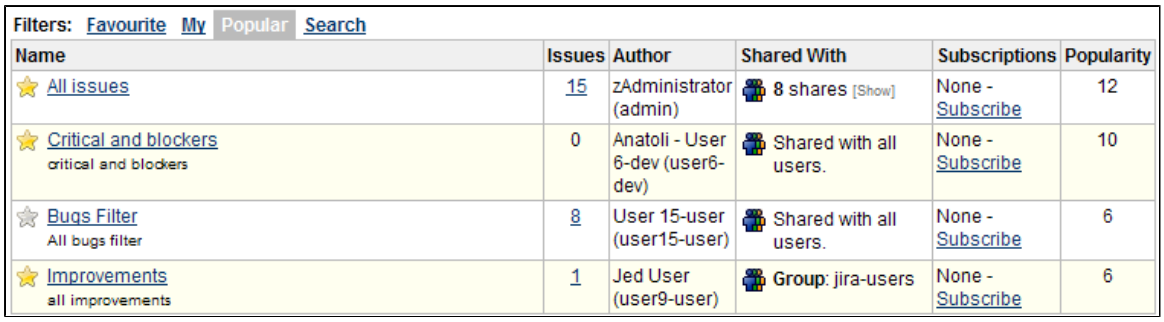
^Top

3

Favourite filters and dashboards

Keep your most commonly used filters and dashboards at your fingertips by adding them as favourites. You can add your own filters and dashboards as favourites in JIRA 3.13, as well as filters and dashboards shared with you by other users. We have also added search for filters and dashboards to help you find the tools you need to manage your information.

- Read more about favourite filters and dashboards in the issue filters and dashboard pages documentation.

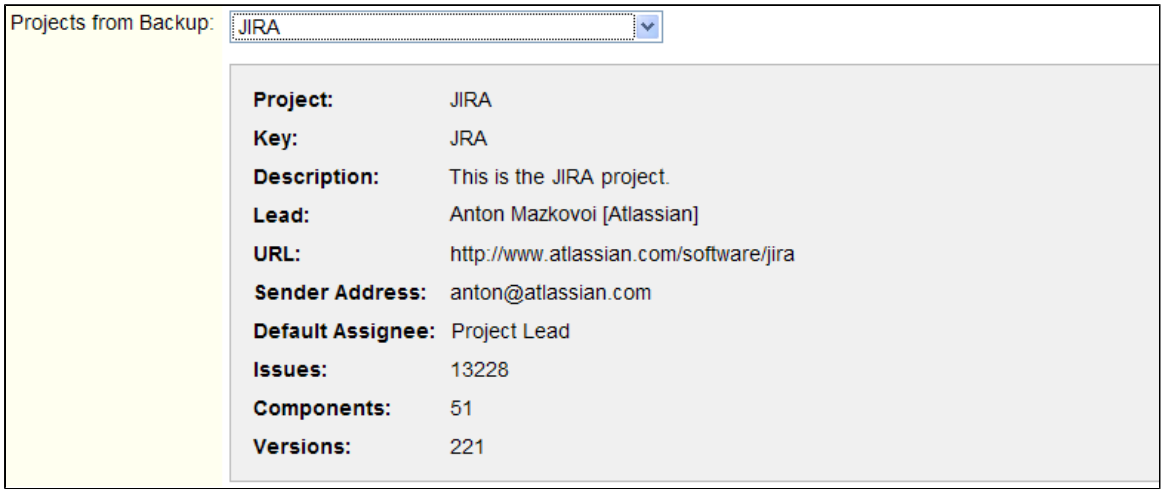


^Top

Restoring projects

One of the most voted for JIRA features has been included in this release — the ability to restore individual projects from a backup file (409 votes!). The project import feature allows you to select a project from a backup file and restore it into an existing JIRA instance, without losing the existing projects or data.

- Read more about restoring a project from backup.

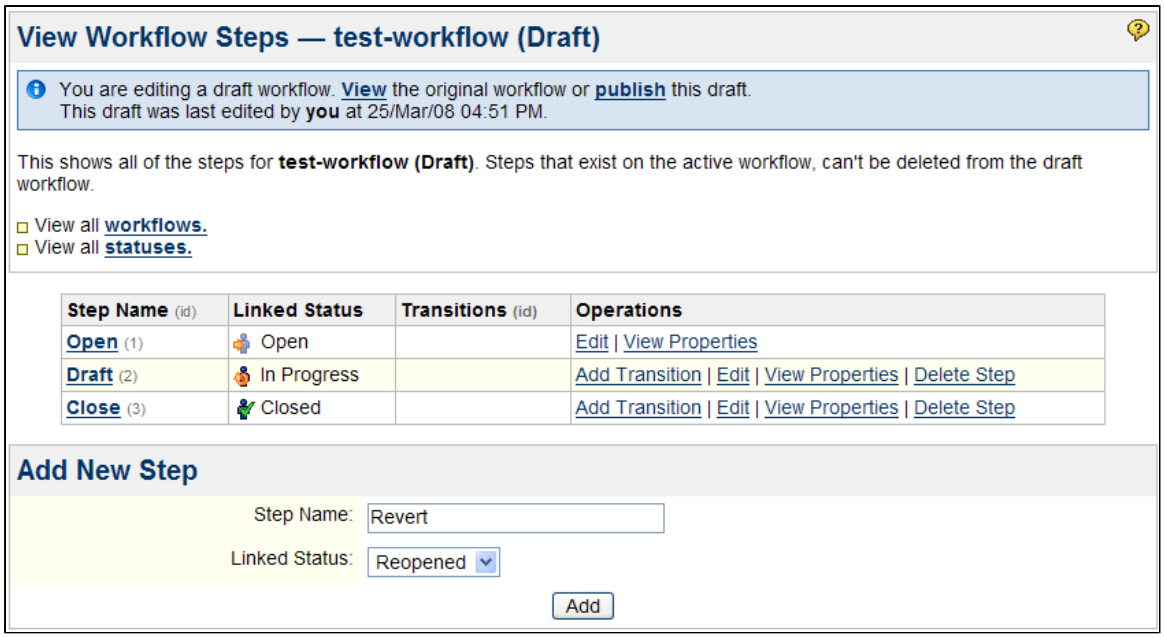


^Top

Editable active workflows

We have added the ability to edit active workflows in this release. You can now change most features of your active workflow without having to edit a separate copy of it.

- Read more about editing active workflows in the JIRA workflow documentation.

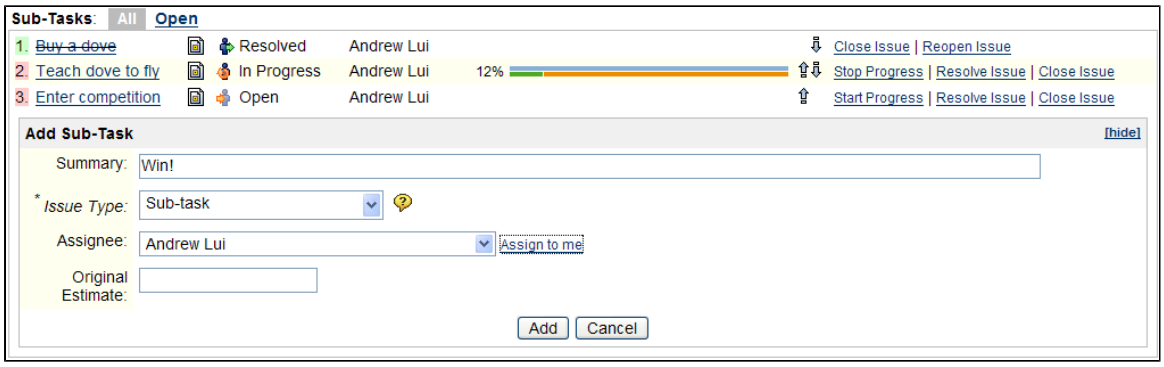


^Top

Enhanced sub-task quick creation

The sub-task 'quick creation' form included on the issue page in JIRA has been enhanced, making it easier for you to quickly add sub-tasks to an issue without having to navigate to a new page. The sub-task quick creation form includes new fields, such as the 'Original Estimate' field and can be easily hidden to reduce clutter on your screen.

- Read more about the sub-task 'quick creation' form in the creating a sub-task documentation.



^Top

Personal licenses

Personal Licenses are now available with JIRA 3.13. Run your own individual non-commercial instance of JIRA under this free license. You can run your fully functional instance of JIRA indefinitely under this license, but you will not be eligible for Atlassian support.

- Read more about personal licenses.

^Top

Plugins

We are bundling a new version of the **FishEye plugin for JIRA** in this release, helping your JIRA instance to work even more closely with your FishEye and Crucible applications. You can now use the FishEye plugin for JIRA to integrate your JIRA instance with Perforce. In addition, we've added a bunch of cool new features to the plugin, including FishEye and Crucible specific charting portlets for JIRA, the integration of reviews and code commits into JIRA workflow and trusted applications support.

- Read more about the FishEye plugin for JIRA

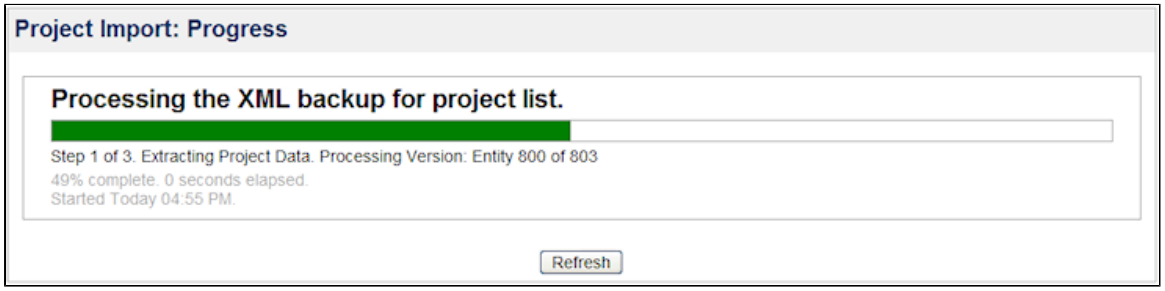
You may also be interested in integrating your JIRA instance with your Bamboo application. We have recently updated the **JIRA Bamboo plugin** with a host of new functionality to let you monitor your builds and issues. (Note that the JIRA Bamboo plugin is currently not bundled with JIRA.)

- You can get the plugin from the JIRA BAMBOO Plugin page.

^Top

Progress bar for long-running operations

We have added a progress bar to a number of long-running operations in JIRA, including workflow migration, project import and re-indexing. This simple visual cue provides you with helpful information about your task, such as the time elapsed, percentage complete and the time you started it.



^Top

Application improvements

The version of Tomcat that is shipped with JIRA Standalone has also been upgraded to version 5.5.26 (previously Tomcat 5.5.20). Refer to the 5.5.x Release Notes on the Apache Tomcat website.

^Top

Plus more than 200 other fixes and improvements

The top 50 most popular issues resolved in JIRA 3.13 are listed below. See all the issues here.

JIRA Issues (50 issues)					
Key	**Type**	**Summary**	**Priority**	**Status**	**Votes**

JRA-2509	Save, reuse and share Dashboards (like dashboard picklist)	Resolved	435
JRA-1604	Import / Export (backup / restore) individual projects	Resolved	409
JRA-4817	Ability to share custom Portals	Resolved	91
JRA-4139	Share filters with *multiple* groups	Resolved	82
JRA-10393	Personal JIRA license	Resolved	50
JRA-7887	Add saved filter types / categories - personal, favourite and all saved filters	Resolved	18
JRA-5803	Global portals, configurable project portlet, project groups	Resolved	13
JRA-9983	CreateOrCommentHandler - cc to watchers	Resolved	4
JRA-5806	Ability to subscribe to shared filters	Resolved	2
JRA-15117	Add user photos to JIRA action item headers for Studio	Resolved	0
JRA-7661	Ability to Edit Active Workflow - Editing Workflows is too inflexible	Resolved	177
JRA-3322	Multiple default dashboards	Resolved	131
JRA-11882	Filter sharing does not currently allow you to share with a role.	Resolved	41
JRA-2394	Change attachment size limit in the Web Interface / GUI	Resolved	16
JRA-6178	shared filters and "favorites"	Resolved	16
JRA-3021	Ability to hide a shared filter	Resolved	11
JRA-10859	Create a generic mechanism within JIRA that allows a user to fire off long running tasks but still receive feedback about the progress of the task.	Resolved	11
JRA-3769	Security levels should be sorted alphabetically or be orderable	Resolved	9
JRA-6124	Share filter per project (or project categorie)	Resolved	7
JRA-10525	Add progress bar to Workflow migration	Closed	6
JRA-5615	Allow default dashboard pages to be updated when multi portlet pages are used.	Resolved	5
JRA-12712	Allow users to select which filters appear in the "List ALL filters" portlet	Resolved	4
JRA-14547	Bypass 1st page of Create Issue (and Sub Task) wizard when there is no "choice" for issue type and project	Resolved	2
JRA-10292	Filters should be hideable	Resolved	2
JRA-12017	Workflow editor needs to be able to handle step="-1" for common-actions	Resolved	2
JRA-10211	Allow specification of permissions on porlets	Resolved	2
JRA-14980	Suboptimal performance for Lucene queries within JIRA instances with a large number of projects and issue security levels	Resolved	2

JRA-12466	Bulk Change should preserve configured columns	Resolved	2
JRA-15314	Put the version of JIRA in the head element of all XML/RSS results	Resolved	1
JRA-13346	Bugzilla importer should not ask for project key and project lead for existing JIRA projects	Resolved	1
JRA-12271	Print system info to the logs on startup	Resolved	1
JRA-15098	Watches and Votes cannot handle users "disappearing" (ie being deleted from external user management).	Resolved	1
JRA-15190	Deterioration of performance due to CachingVersionStore lock	Resolved	1
JRA-12978	Retain CVS information even if issue key changes	Resolved	1
JRA-9524	edit name for Workflow	Resolved	1
JRA-15313	Saved Filter URLs do not respect additional parameters	Resolved	1
JRA-13913	Make the banner a background image rather than an object in the cell	Resolved	1
JRA-15357	Translation from UK to FR	Resolved	0
JRA-15370	Add smarts to choosing the default tab in Manage Dashboard and Manage Filter	Resolved	0
JRA-14552	Make the "Fix Version", "Component", "Assignee" and "Reporter" fields in Issue Navigator results always link to the relevant Browse pages.	Resolved	0
JRA-13918	Add a help icon for issue security levels	Resolved	0
JRA-14466	Place Extra JIRA system Info into exported XML	Resolved	0
JRA-13439	Smart querying : use syntax 'c:' to search for components	Resolved	0
JRA-14554	Ensure that the "Attachment" input field is on the default screen when the system is setup anew	Resolved	0
JRA-14546	Remove "Manage Attachments" link and place the link on "File Attachments" text	Resolved	0
JRA-13886	Filter Statistics Portlet: Options "Sort By" and "Sort Direction" same as in "Two Dimensional Filter Statistics" portlet	Resolved	0
JRA-14198	Key Comparator is slow and inefficient	Resolved	0
JRA-15311	Create RemoteSecurityLevel for SOAP API	Resolved	0
JRA-14290	Make JIRA SOAP client work against jira.atlassian.com	Resolved	0
JRA-14564	The Bugzilla importer needs a build.xml file	Resolved	0

^Top

JIRA 3.13 Upgrade Guide

Upgrading from JIRA 3.12.xx to 3.13

Please follow the JIRA general upgrade instructions, plus note the following:

1. Introduction of Favourite Dashboards and Filters

Favourite Dashboards

JIRA 3.13 introduces the favourite dashboards feature, which allows you to add dashboard pages that are owned by you or shared by other users as favourites (and hence, are displayed as tabs on your dashboard). On upgrade to JIRA 3.13, all your dashboard pages will be added as your favourites and displayed on your dashboard. If you do not wish any of your dashboards to be added as favourites, then you can remove them as favourites after the upgrade. See the dashboards documentation for details.

Favourite Filters

Similar to favourite dashboards, JIRA 3.13 introduces the favourite filters feature, which allows you to add issue filters that are owned by you or shared by other users as favourites. On upgrade to JIRA 3.13, all your issue filters will be added as your favourites. If you do not wish any of your filters to be added as favourites, then you can remove them as favourites after the upgrade. See the issue filters documentation for details.

please note, this change will not affect issue filter sharing, e.g. if you are using a shared issue filter in one of your dashboard portlets, it will still be shared with you after the upgrade.

please also note, that any custom developed portlets (or other JIRA objects that use filters that have been developed by 3rd parties) that have a dropdown list (not a pop-up picker) for filters, will now only show a list of the user's favourite filters, instead of all shared filters.

Favourite Filters portlet

The 'List All Filters' portlet has been replaced with the 'Favourite Filters' portlet in this release. Your dashboard will be automatically upgraded if it is currently configured to display the 'List All Filters' portlet.

2. Tomcat, MySQL database connection dropouts

Please note, if you wish to use a MySQL database with JIRA Standalone you must set up the bundled Tomcat server (version 5.5.26) to survive connection closures. You must also do this if you are running JIRA EAR/WAR in Tomcat 5.5.25 or later, or Tomcat 6.0.13 or later. Versions 5.5.25 and above of Tomcat 5, and versions 6.0.13 and above of Tomcat 6, have been noted to exhibit problems maintaining connections to MySQL databases. Please read this document for details on the changes required.

3. Changes to jira-application.properties

jira.subscription.email.max.issues property

The `jira.subscription.email.max.issues` property has been added to the `jira-application.properties` file. This property allows you to specify the maximum number of issues that can be included in an email subscription. The default value for this property is 200. You may wish to update this property after the upgrade if you wish to set a different limit on the number of issues that can be included in an email subscription. See the documentation on Advanced JIRA Configuration for further details on this file.

4. Support for Portlet Plugins with JSP Views Discontinued

Portlet plugins with JSP views are no longer supported. If you have written a custom Portlet plugin and have used a JSP as the view template, you will need to convert your JSP to Velocity.

5. Updates to JIRA SOAP and XML-RPC APIs

com.atlassian.jira.rpc.soap.JiraSoapService

- replaced

```
RemoteProject[] getProjects(String token) throws RemoteException;
```

  with

```
RemoteProject[] getProjectsNoSchemes(String token) throws RemoteException
```

*You should use **getProjectsNoSchemes()** instead because it much more memory efficient and quicker.*

- added

```
RemoteProject getProjectWithSchemesById(String token, Long projectId) throws
RemoteException;
```

- deprecated

```
RemoteFilter[] getSavedFilters(String token) throws RemoteException;
```

- added

```
RemoteFilter[] getFavouriteFilters(String token) throws RemoteException;
```

com.atlassian.jira.rpc.xmlrpc.XmlRpcService

- replaced

```
Vector getProjects(String token) throws Exception;
```

  with

```
Vector getProjectsNoSchemes(String token) throws Exception;
```

- deprecated

```
Vector getSavedFilters(String token) throws Exception;
```

- added

```
Vector getFavouriteFilters(String token) throws Exception;
```

6. Crowd Cache Timeout

This is only applicable if you are using Crowd.

The default timeout for caching user details has changed from 5 minutes to 2 hours. This will improve the performance of the application but will mean that it will take longer for changes to user details to reach the application. Details on how to configure the Crowd caches can be found here.

Upgrading from JIRA 3.12 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.13.5 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

21 July 2009

The Atlassian JIRA team is proud to announce the release of **JIRA 3.13.5** in Standard, Professional and Enterprise editions. This point release includes over **30** bug fixes and improvements.

JIRA 3.13.5 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.13 yet?
Take a look at all the new features in the JIRA 3.13 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.13.5 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.13.5 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.13.5 Upgrade Guide

Upgrading from JIRA 3.13.x to 3.13.5

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.12.x and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.13.4 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

5 May 2009

The Atlassian JIRA team is proud to announce the release of **JIRA 3.13.4** in Standard, Professional and Enterprise editions. This point release includes over **20** bug fixes and improvements.

JIRA 3.13.4 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.13 yet?
Take a look at all the new features in the JIRA 3.13 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.13.4 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.13.4 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.13.4 Upgrade Guide

Upgrading from JIRA 3.13.x to 3.13.4

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.12.x and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.13.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

2 April 2009

The Atlassian JIRA team is proud to announce the release of **JIRA 3.13.3** in Standard, Professional and Enterprise editions. This point release includes over **85** bug fixes and improvements, including an important security fix — please see JIRA Security Advisory 2009-04-02 for details.

JIRA 3.13.3 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.13 yet?
Take a look at all the new features in the JIRA 3.13 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.13.3 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.13.3 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.13.3 Upgrade Guide

Upgrading from JIRA 3.13.2 to 3.13.3

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.12 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.13.2 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

9 December 2008

The Atlassian JIRA team is proud to announce the release of **JIRA 3.13.2** in Standard, Professional and Enterprise editions. This point release includes over 45 bug fixes and improvements, including an important security fix — please see JIRA Security Advisory 2008-12-09 for details.

JIRA 3.13.2 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.13 yet?
Take a look at all the new features in the JIRA 3.13 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.13.2 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.13.2 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.13.2 Upgrade Guide

Upgrading from JIRA 3.13.1 to 3.13.2

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.12 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.13.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

29 October 2008

The Atlassian JIRA team is proud to announce the release of **JIRA 3.13.1** in Standard, Professional and Enterprise editions. This point release includes over 35 bug fixes and improvements, including important security fixes — please see JIRA Security Advisory 2008-10-29 for details.

JIRA 3.13.1 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.13 yet?
Take a look at all the new features in the JIRA 3.13 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.13.1 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.13.1 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.13.1 Upgrade Guide

Upgrading from JIRA 3.13 to 3.13.1

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.12 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Known Issues

IMAP message handling produces issues with no description

If your JIRA instance is set up to create issues and/or comments from e-mails that come from an IMAP mail box, you will need to deploy a patch to address a problem with issue creation (as described in JRA-15954). Please deploy the patch file attached to JRA-15954 (instructions for deploying the patch are contained in this comment of JRA-15954).

JIRA 3.12 Release Notes

Atlassian Software Systems is proud to present JIRA 3.12.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.12 provides a number of enhancements for the upcoming JIRA Studio. Because some of these enhancements may be of benefit to you, we have decided to provide them as a public release rather than making you wait until JIRA 4.0.

The major feature of this release is the ability to 'trust' Confluence. For people who use both JIRA and Confluence, the ability to configure a 'trust' relationship between the two will allow for a seamless end-user experience, e.g. the 'JIRA Issues' macro will now display exactly the same list of issues on a Confluence page that the user would see in the JIRA Issue Navigator. No longer is there a need to hard-code JIRA user names and passwords on a Confluence page. Note that you will need Confluence 2.7 (which is due for release this month) or later.

Also included in 3.12 is a new global permission, '**JIRA System Administrators**'. This will be particularly useful for organisations where the JIRA administrators are not necessarily the same people who are responsible for maintaining the file system and network environment. Granting the 'JIRA System Administrators' permission to only a controlled number of people will give your Windows or UNIX administrators greater peace of mind, while people with the 'JIRA Administrators' permission can enjoy full control over JIRA-specific administration.

Upgrading to JIRA 3.12 is free for all customers with active JIRA software maintenance as at 30 November 2007.

Highlights of JIRA 3.12:

- 'Trusted' Confluence
- 'JIRA System Administrators' permission
- FishEye plugin now bundled with JIRA
- Improvements to the Subversion plugin
- Improvements to the 'Project Statistics' and 'Filter Statistic' portlets
- New post function for workflows: 'Assign to Current User'
- Enhanced language support for searching
- Visual SourceSafe plugin
- Plus more than 100 other fixes and improvements

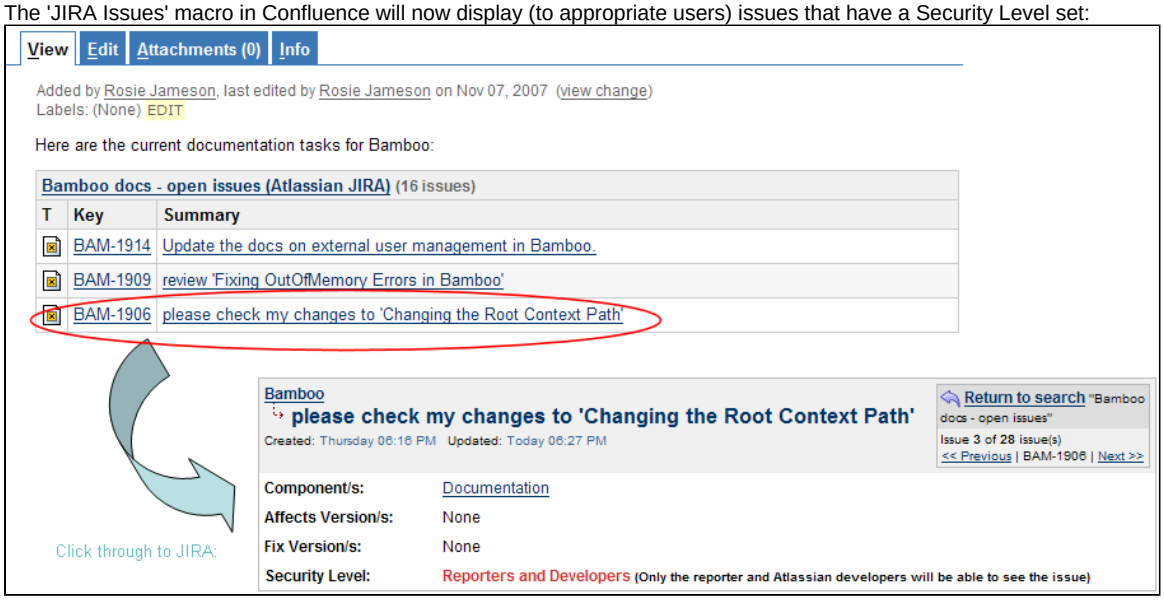
Upgrading to JIRA 3.12
JIRA 3.12 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 3.12 Upgrade Guide.

Highlights of JIRA 3.12

'Trusted' Confluence

For people who use both JIRA and Confluence, the ability to configure a 'trust' relationship between the two will allow for a seamless end-user experience, e.g. the 'JIRA Issues' macro will now display exactly the same list of issues on a Confluence page that the user would see in the JIRA Issue Navigator. No longer is there a need to hard-code JIRA user names and passwords on a Confluence page. (Note that you will need Confluence 2.7 or later.)

The 'JIRA Issues' macro in Confluence will now display (to appropriate users) issues that have a Security Level set:



^Top

'JIRA System Administrators' permission

A new global permission has been added to JIRA, to allow for separation of duties.

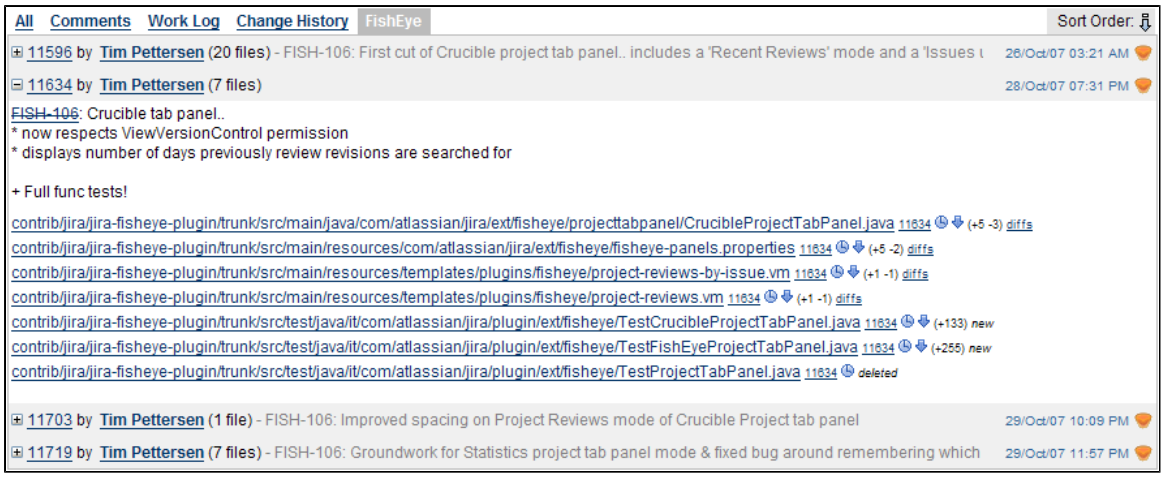
- People who are granted the new '**JIRA System Administrators**' permission can perform *all* of the administration functions in JIRA, including functions which could affect the application environment or network (e.g. data import/export, SMTP configuration, database connection).
- People with only the '**JIRA Administrators**' permission can now perform most administration functions (e.g. creating new JIRA users; creating projects), but not functions which could affect the application environment or network.

This will be useful for organisations which need to delegate JIRA-specific administration privileges to particular people, without granting them total system administration privileges.
Note that everyone who had the '**JIRA Administrators**' global permission before the upgrade will automatically receive the new '**JIRA System Administrators**' global permission during the upgrade. This will ensure that everyone can still perform the same functions they could previously.

^Top

FishEye plugin now bundled with JIRA

The FishEye plugin automatically detects JIRA issue-keys in your code commit messages. Within your JIRA issues and projects, relevant commit messages will be displayed along with links to the FishEye changesets and files — and (optionally) Crucible code reviews.

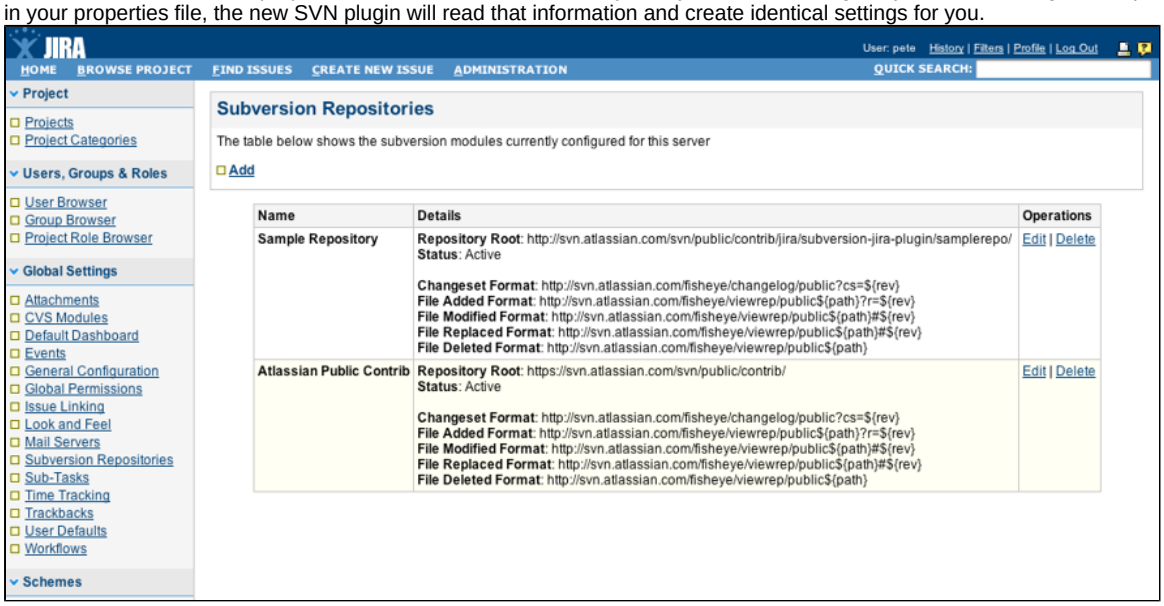


- Click here for a live example.

^Top

Improvements to the Subversion plugin

Using the Subversion plugin (available separately), it is now possible to configure Subversion repositories from within JIRA. No longer do you need to muck around with .properties files and bounce JIRA every time you make a change! If you have existing SVN repositories configured in your properties file, the new SVN plugin will read that information and create identical settings for you.

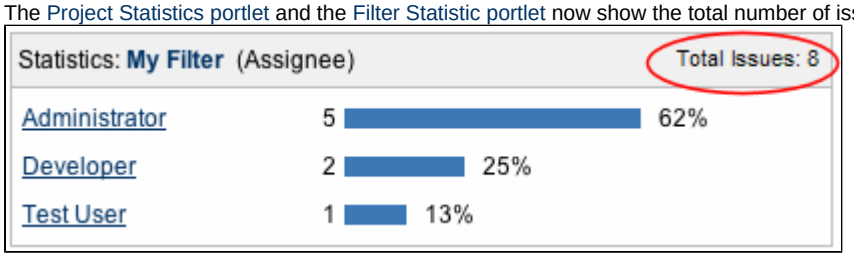


Click to enlarge

^Top

Improvements to the 'Project Statistics' and 'Filter Statistic' portlets

The Project Statistics portlet and the Filter Statistic portlet now show the total number of issues that match the filter, e.g.:



^Top

New post function for workflows: 'Assign to Current User'

With the new post function 'Assign to Current User', you can now automatically assign an issue to the logged-in user when the issue moves through a particular workflow transition.

This is useful if you need to assign an issue to the logged-in user under particular circumstances, but not give them full rights to assign issues. For example, on Atlassian's support system, when a support specialist clicks 'Start Investigating', the issue is automatically assigned to them — even if they don't have 'Assign Issues' permission.

^Top

Enhanced language support for searching

The range of available languages for JIRA search indexes has been expanded. This means that even more people around the world can nov choose to have JIRA index their issue data in their native language.

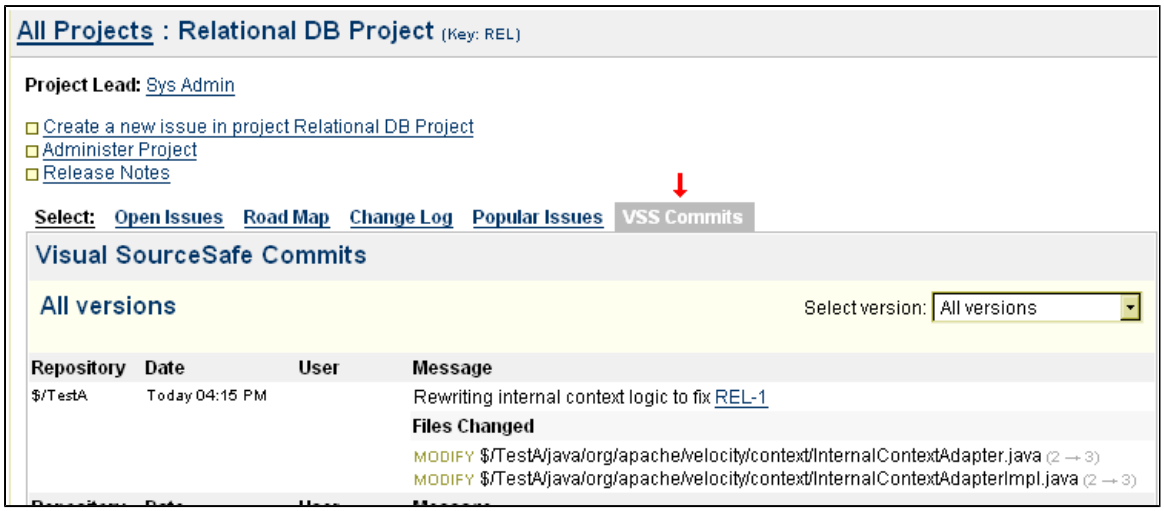
This provides more meaningful search results for end-users, because:

- 'stop' words (i.e. words that are deliberately ignored by the JIRA search engine, such as 'the') are now recognised in more non-English languages (Brazilian, Chinese, Czech, Greek, French, Dutch, Thai).
- 'stemming' (i.e. the derivation of related words, such as 'archived', from a stem such as 'archive) is now supported in French, Brazilian, German, French, Dutch, Russian and English. For example, if your index language is set to French:
  - a search for "marchera" will find "L'enfant a marché"; and
  - a search for "marché" will find "l'enfant marchera".

^Top

Visual SourceSafe plugin

The new VSS plugin displays Microsoft Visual SourceSafe commit information (along with the changed paths) related to JIRA issues, projects or project versions. This plugin is in **beta** and available for a separate download.



^Top

9

Plus more than 100 other fixes and improvements

JIRA Issues (120 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-4085		links in jira issues should be relative ratherr than use the BASE URL		Resolved
JRA-5819		Assign to current user workflow post-function		Resolved
JRA-5900		Error when trying to edit a group		Resolved
JRA-7136		Statistics are wrong when using a filter that searches through comments		Resolved
JRA-8457		Cannot administer my own profile: "The user does not exist. Please try another"		Resolved
JRA-8872		Provide a wrap option with the noformat markup		Resolved
JRA-9171		A timed out session causes stacktraces in half-completed actions		Resolved
JRA-10489		JIRA does not process multipart/alternative mails properly		Resolved
JRA-10508		Insecure "Remember my Login" cookie on https-sites		Resolved
JRA-10515		Adding CSV field to a Mutli User Custom Field causes error		Resolved
JRA-10546		Moving a subtask doesn't inherit the security level of its new parent		Resolved
JRA-10989		WIKI renderer preview does not display correctly		Resolved
JRA-11358		Make plugins configurable		Resolved

JRA-11396	Redundant JIRA Global Permission (Manage Group Filter Subscriptions) in Standard Edition		Resolved
JRA-11446	AccessLogFilter logs everything twice		Resolved
JRA-11767	Allow code and noformat sections in Wiki text to scroll horizontally		Resolved
JRA-11788	Filter window is too small when it opens		Resolved
JRA-11892	Support for Dutch Language		Resolved
JRA-12042	Typo in an RPC API parameter name		Resolved
JRA-12091	Issue's parentId not set on subtask deletion event		Resolved
JRA-12143	Improve doc on hiding fields		Closed
JRA-12513	Allow NotificationType objects to be registered dynamically		Resolved
JRA-13040	Log critical system operations like reindexes		Resolved
JRA-13090	Multiple file upload fails when the sum of files size exceed the upload limit, even if each file is smaller than the limit.		Resolved
JRA-13128	Save position and size of Filter/History/Help pop up windows		Resolved
JRA-13155	Change renewal hyperlink to use new website redirects		Resolved
JRA-13187	Show Total in Caption of "Filter Statistics" portlet		Resolved
JRA-13188	Trailing + characters are truncuated from URLS in the Description / Environment / Comment fields.		Resolved
JRA-13205	Full content Word export from Issue navigator duplicates wiki style table		Resolved
JRA-13229	controlfooter.jsp always closes the html tr tag even when the nolabel param is set		Resolved
JRA-13231	JIRA installer ships with client JRE instead of server JRE		Resolved
JRA-13263	IssueNotFoundException is thrown right out to the user if a workflow action is taken on a deleted issue		Resolved
JRA-13282	'JIRA System Administrator' Permission		Resolved
JRA-13284	Problem in printing project portlet: red section displayed as white.		Resolved
JRA-13315	Non default permission types can cause Stack overflow if added to wrong permissions		Resolved
JRA-13386	Hide Log Work operation and Work Log tab if Time Tracking Field is hidden in Field Config		Resolved
JRA-13400	Remember me cookie issue with Glassfish; integrate latest Seraph into JIRA		Resolved
JRA-13402	Retain state of attachment comments when switching from single attachment to multiple		Resolved
JRA-13430	When invalid search term was entered in custom field, error message highlights Text Search Query: field.		Resolved
JRA-13436	Small French translation problem		Resolved

JRA-13442	Improve UI for component admin		Resolved
JRA-13465	Right border does is missing on Add Portlet screen on Safari		Resolved
JRA-13473	Double quotes allowed in transition name while editing a transition		Resolved
JRA-13475	Update issue field post function in Create Transition must be the first post function executed in order to actually set the issue field		Closed
JRA-13509	Special character in group name causes permalink do not function properly		Resolved
JRA-13516	FieldLayoutSchemeImpl caching is not thread-safe		Resolved
JRA-13521	Need to add some unit tests for the bulk edit issue count limit in BulkEdit1.doValidate()		Resolved
JRA-13522	Need to improve the way the calendar-<locale>.js files are served		Resolved
JRA-13523	Multi user custom field cannot be used with the assignable user permission		Resolved
JRA-13542	Format the relative Today and Yesterday as per configured Day Format		Resolved
JRA-13553	Misleading permission violation message when attempting to edit a closed issue		Resolved
JRA-13554	Make Version Workload Report styled like Time Tracking report (nicer)		Resolved
JRA-13558	MailingListCompiler trying to send email with empty "To"		Resolved
JRA-13567	Improvement on Jelly tag documentation		Closed
JRA-13572	in the 'Add Priority' form, should 'Status Color' be 'Priority Colour' ?		Resolved
JRA-13578	Typo in Issue Navigator in Slovak language		Resolved
JRA-13581	Replace hard-coded string in date pickers		Resolved
JRA-13592	Setting transport to SMTPS in JNDI mail resource is broken		Closed
JRA-13597	Loading Event Listeners is not synchronized		Resolved
JRA-13598	The EmoticonRendererComponent uses the incorrect IconManager		Resolved
JRA-13613	Time Tracking Report's summary field should be linked		Resolved
JRA-13626	Renderer component does not work with profiling enabled		Resolved
JRA-13654	Allow AttachFile jelly tag to specify the created date for an attachment		Resolved
JRA-13655	The "attach" button in screenshot applet is not translated properly		Resolved
JRA-13656	Time Tracking Label for "Issue"		Resolved
JRA-13666	Missing i18n keys in notification scheme		Resolved
JRA-13673	Admin portlet can show null date for license expirery		Resolved

JRA-13677	The property field for JIRA's portlets and reports are not in order sequence		Resolved
JRA-13687	French Translation Incorrect "traitement"		Resolved
JRA-13699	Deleting a group does not check if there are any worklogs with the group restriction like it does for comments		Resolved
JRA-13703	getFieldsForEdit does not included "Reporter" field, "Due Date" field and "FixVersion" field		Resolved
JRA-13712	user value of JiraAuthenticationContext not set is SOAP service getIssue()		Resolved
JRA-13718	Update AttachFile jelly tag documentation		Closed
JRA-13724	Caledar popup doesn't work in several languages		Resolved
JRA-13727	Trusted Applications: Support Authentication Context Passing from Confluence or another Application to JIRA		Resolved
JRA-13742	minor grammatical error in bulk move		Resolved
JRA-13744	IssueLevelSecurity permission check does not work with a DocumentIssueImpl if no security level has been set.		Resolved
JRA-13748	Clean View Issue page by moving (View) links for voters and watchers to link on actual value		Resolved
JRA-13750	Help link in browse projects page references version management page in the documentation		Closed
JRA-13752	Issue Linking docs out of date		Closed
JRA-13766	Deleting a version can leave gaps in the version sequence		Resolved
JRA-13784	Update Bugzilla import guide to ask users to run Bugzilla 'Sanity Check' tool first		Closed
JRA-13792	Adding Greek support in Full-Text search		Resolved
JRA-13794	broken link on the 'Trackback Settings' screen		Resolved
JRA-13805	In quicksearch, issue type has higher priority than project key		Resolved
JRA-13809	Add more indexing and search languages		Resolved
JRA-13818	Username with # character breaks on "Assign to me" operation		Resolved
JRA-13823	Move mysql-guide-linux.html page to Confluence		Closed
JRA-13824	identify entries in site.xml that have no label, and move to CAC where appropriate		Closed
JRA-13829	Jelly Documentation - error in comment tags		Closed
JRA-13840	Filter parameters panel on the the left should be collapsed by default when I come to Issue navigator from dashborad plugins		Resolved
JRA-13851	List of available colours for {color} tag in Wiki Style Renderer		Closed
JRA-13853	No space above the Road Map portlet		Resolved
JRA-13856	Upgrade atlassian-extras for new license types.		Resolved

JRA-13881	Sub-tasks are visibile in Issue Finder while the Parent task is not		Resolved
JRA-13905	Unable to remove group at the Assign Groups to Project Role page if the group name has the double quote		Resolved
JRA-13906	Duplicate i18n-keys in the same language-files		Resolved
JRA-13910	Update the comment in jira-application.properties to indicate that a hyphen should not be used in the project key.		Resolved
JRA-13911	Projects portlet sometimes displays Components and Versions links and sometimes doesn't		Resolved
JRA-13916	"Manage Portal" screen is missing the default template info		Resolved
JRA-13920	Page title is incorrect when the user logs out		Resolved
JRA-13921	Resetting custom version picker field results in incorrect search results		Resolved
JRA-13932	Document [permlink]		Closed
JRA-13935	Anonymous reporter makes rss feeds throw NullPointerException		Resolved
JRA-13941	Add New Issue Type Scheme form does not validate name nicely		Resolved
JRA-13949	Custom Field Type: "Version Picker" - Scroll Bar for displaying List Not Working		Resolved
JRA-13952	Set up redirects for JIRA doc pages moved to Confluence		Resolved
JRA-13974	Review changes to Profiling documentation - new content on making Profiling permanent		Closed
JRA-13977	Correct Jelly Tags documentation for new permissions in AddPermission		Closed
JRA-13991	Translation for French and German breaks in the change password screen		Resolved
JRA-13994	Document that users importing from other systems such as via CSV should backup their data first		Closed
JRA-13998	New german translation is "buggy" concerning Bulkchange		Resolved
JRA-14012	Authenticating security providers fails due to ClassLoader bugs		Resolved
JRA-14022	JIRA Tomcat 6.0 doco changes		Closed
JRA-14130	charting plugin ignores text query term		Resolved
JRA-14138	User picker still shows user after being removed from all groups		Closed
JRA-14278	Update documentation - PersistenceManager has to be disabled for Tomcat		Resolved
JRA-15199	Formatting of code sections of LDAP debugging documentation has gone awry		Closed
JRA-15230	Typo on Downgrading JIRA doc		Closed
JRA-15587	Link to Issue Type in 'What is an Issue?' page points to the incorrect anchor		Closed

^Top

JIRA 3.12 Upgrade Guide

Upgrading from JIRA 3.11 to 3.12

Please follow the JIRA general upgrade instructions, plus note the following:

1. Everyone who had the '**JIRA Administrators'** global permission before the upgrade will automatically receive the new '**JIRA System Administrators**' global permission during the upgrade. This will ensure that everyone can still perform the same functions they could previously.
2. The following new Seraph property can be used to fix JRA-10508:

```
<!--  If this parameter is set to true, the cookie will never be set secure.  This is
useful if you're logging
              into JIRA via https, but want to browse JIRA over http.  This flag will
ensure that the remember me option
              works correctly. -->
        <init-param>
            <param-name>insecure.cookie</param-name>
            <param-value>true</param-value>
        </init-param>
```

3. Due to the Seraph upgrade, to fix JRA-10508 all users will be prompted to log in again. This will also affect users who have the 'Remember me' checkbox ticked.
4. If you are building JIRA from source, please note that Maven2 is now required for a build. This is because the JIRA Fisheye Plugin requires Maven2.
5. If you are using the JIRA Toolkit, it is recommended that you upgrade to the latest version in order to fix JRA-13553
6. Please note that the new Trusted Applications feature is not supported on Orion versions prior to 2.0.5. Also note that Resin2 has problems and you will need to update the Resin extra jars.
7. There is a new database table. Please see the following page for details

Upgrading from JIRA 3.10.2 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Using the Trusted Applications feature with Crowd

Please note that older versions of the Crowd client, (i.e. version 1.2.1 or earlier), can interfere with the correct operation of the Trusted Applications feature. If you are enabling Trusted Applications and using Crowd, please ensure that your Crowd client is **version 1.2.2 or later**.

JIRA 3.12 DB Schema Changes

The table below summarises the changes to the database schema. Please note, that if you have developed any custom utilities which query or modify the JIRA database directly (i.e. without using the JIRA API), please check whether the utilities need to be updated.

New Database Table Table - TRUSTEDAPP

The database table `trustedapp` has been added to schema support Trusted Applications:

It has the following columns:

Column Name	Type
id	numeric
applicationId	long-varchar
name	long-varchar
publicKey	very-long
ipMatch	very-long

urlMatch	very-long
timeout	numeric
created	date-time
createdBy	long-varchar
updated	date-time
updatedBy	long-varchar

The ID column is the primary key.

For a mapping of the above type to your particular database, please see the appropriate `fieldtype-*.xml` file in JIRA's `WEB-INF/classes/entitydefs/` directory.

JIRA 3.12.3 Release Notes

30 April 2008

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.12.3 Release Notes

The Atlassian JIRA team is proud to announce the release of **JIRA 3.12.3** in Standard, Professional and Enterprise editions. This point release includes over 40 bug fixes and improvements which can be viewed below. Click a specific issue to see details of the fix, and to download patches where relevant.

JIRA 3.12.3 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.12 yet?
Take a look at all the new features in the JIRA 3.12 Release Notes and see what you are missing out on!

Download Latest Version

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.12.3 Upgrade Guide.

Updates and Fixes in this Release

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.12.3 Upgrade Guide

Upgrading from JIRA 3.12.2 to 3.12.3

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.11 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.12.2 Release Notes

21 February 2008

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.12.2 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.12.2 in Standard, Professional and Enterprise editions. This point release includes over 30 bug fixes and improvements, including important security fixes: please see JIRA Security Advisory 2008-02-21 for details. Additionally, the FishEye plugin now supports **trusted applications** for increased security over the existing username and password authentication. Read more about using the FishEye plugin.

JIRA 3.12.2 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.12 yet?
Take a look at all the new features in the JIRA 3.12 Release Notes and see what you are missing out on!

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.12.2 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.12.2 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.12.2 Upgrade Guide

Upgrading from JIRA 3.12.1 to 3.12.2

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.11 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.12.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

24 December 2007

JIRA 3.12.1 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.12.1 in Standard, Professional and Enterprise editions. This point release includes over 10 bug fixes and improvements, including important security fixes: please see JIRA Security Advisory 2007-12-24 for details.

JIRA 3.12.1 is of course free to all customers with active JIRA software maintenance.

Don't have JIRA 3.12 yet?
Take a look at all the new features in the JIRA 3.12 Release Notes and see what you are missing out on!

Upgrading from a Previous Version of JIRA

If you are upgrading, please read the JIRA 3.12.1 Upgrade Guide.

Updates and Fixes in this Release

JIRA 3.12.1 includes the following updates and bug fixes:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.12.1 Upgrade Guide

Upgrading from JIRA 3.12 to 3.12.1

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.11 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.11 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is delighted to present JIRA 3.11.

Upgrading to JIRA 3.11 is free for all customers with active JIRA software maintenance as at 24th September 2007.This release focuses on time-tracking. Time-tracking data (that is, the estimated and actual time spent on an issue) now includes the issue's sub-tasks. The aggregated time-tracking data is displayed both within individual 'parent' issues and in the Issue Navigator, so it can be easily reported on, exported to Excel, etc.
Being able to track your project's Road Map (scheduled issues) has long been a useful feature of JIRA. But how do you manage programs o multiple, related projects? In JIRA 3.11, the new Road Map portlet shows upcoming milestones across multiple projects of your choice.

Upgrading to JIRA 3.11
JIRA 3.11 can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 3.11 Upgrade Guide.

Thank you for your feedback:

31 new feature and improvement requests implemented

223 votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.

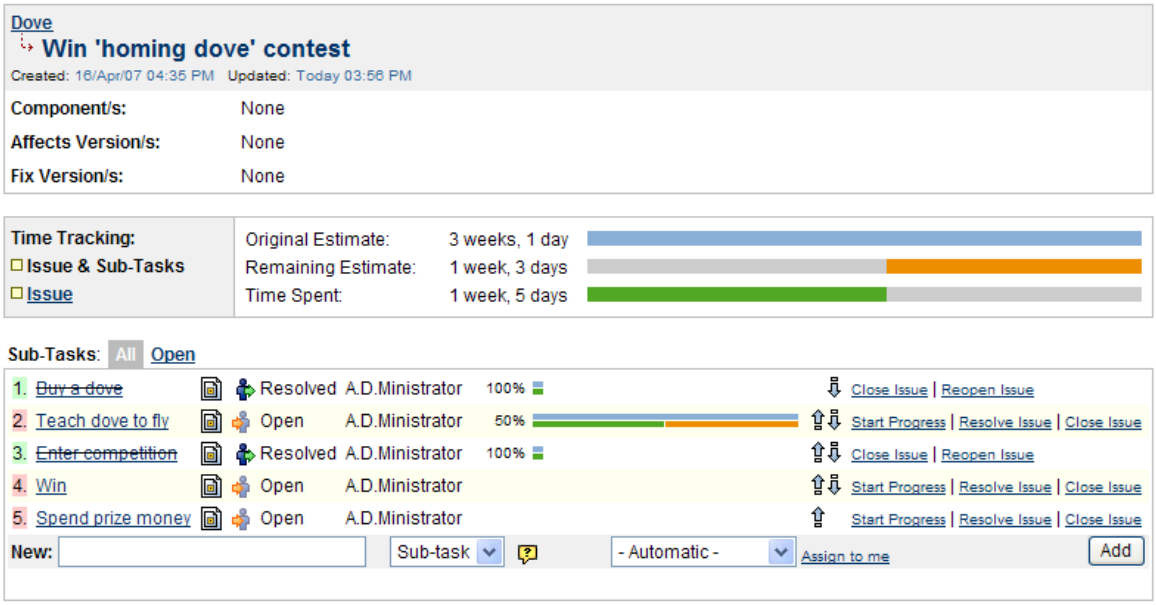
Highlights of JIRA 3.11:

- Sub-task progress shown within issues
- Issue Navigator offers sub-task aggregates
- Time Tracking reports now include sub-tasks
- Multi-project 'Road Map' portlet
- Performance improvements
- Indexing improvements
- JIRA Labels Plugin
- Plus over 70 other fixes and improvements

Highlights of JIRA 3.11:

Sub-task progress shown within issues

When viewing an issue, you can now choose whether to view time tracking data for the issue only, or for the issue plus its sub-tasks:



- Click '**Issue**' (in the Time Tracking box) to show time tracking data for the 'parent' issue only, or '**Issue & Sub-Tasks**' to include the issue's sub-tasks.
- Time tracking data comprises:
  - **Original Estimate** (blue) — the amount of time the issue was expected to take to resolve, when it was first created.
  - **Remaining Estimate** (orange) — the remaining amount of time the issue is currently expected to take to resolve.
  - **Time Spent** (green) — the amount of time logged working on the issue so far.

Please note that sub-tasks are only available in the Enterprise and Professional editions of JIRA.

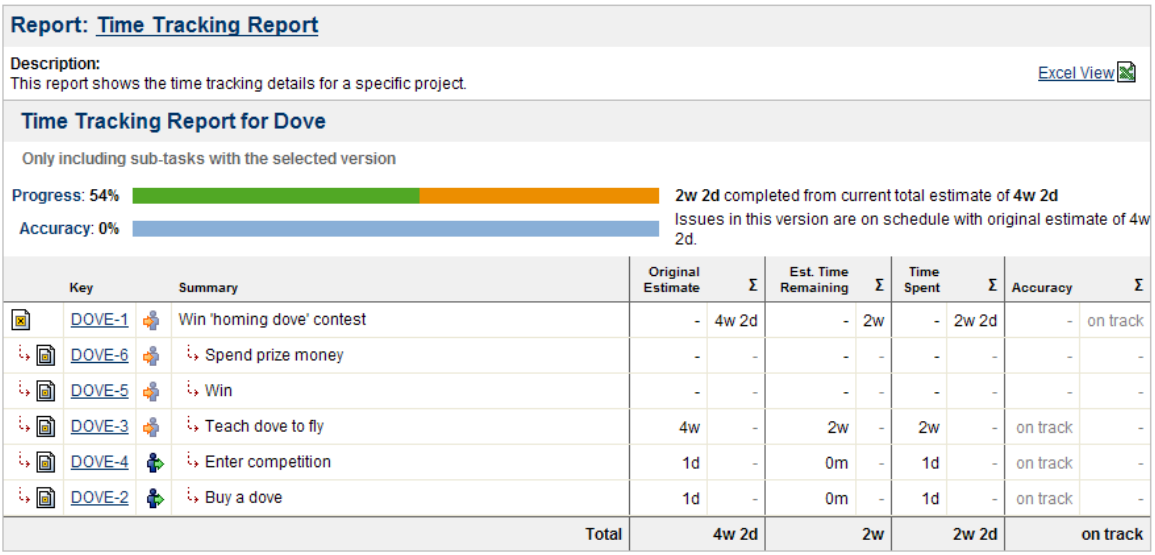
Issue Navigator offers sub-task aggregates

To take advantage of the new sub-task aggregates, the following time tracking fields are now available in the Issue Navigator:

- **Progress** — an issue's Time Spent, as a percentage of the issue's Original Estimate.
- **Progress** — the aggregate time spent on an issue's sub-tasks, as a percentage of the sub-tasks' aggregate Original Estimate.
- **Original Estimate** — the aggregate Original Estimate for an issue's sub-tasks.
- **Remaining Estimate** — the aggregate Remaining Estimate for an issue's sub-tasks.
- **Time Spent** — the aggregate Time Spent for an issue's sub-tasks.

Time Tracking reports now include sub-tasks

- A 'parent' issue now has two separate sets of time tracking data, if applicable: (1) its own; and (2) an *aggregate* that includes the issue's own time-tracking plus all sub-tasks that the user has permission to see.
- The Time Tracking report now includes the aggregate data as shown in the new '' columns:



Report: Time Tracking Report

Description:
This report shows the time tracking details for a specific project.

Excel View

Time Tracking Report for Dove

Only including sub-tasks with the selected version

Progress: **54%** — **2w 2d** completed from current total estimate of **4w 2d**

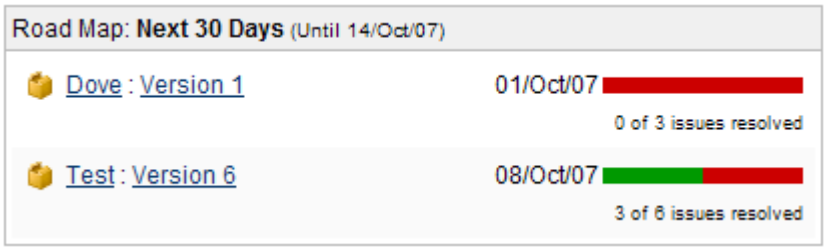
Accuracy: **0%** — Issues in this version are on schedule with original estimate of 4w 2d.

Key	Summary	Original Estimate	Σ	Est. Time Remaining	Σ	Time Spent	Σ	Accuracy	Σ
DOVE-1	Win 'homing dove' contest	-	4w 2d	-	2w	-	2w 2d	-	on track
DOVE-6	Spend prize money	-	-	-	-	-	-	-	-
DOVE-5	Win	-	-	-	-	-	-	-	-
DOVE-3	Teach dove to fly	4w	-	2w	-	2w	-	on track	-
DOVE-4	Enter competition	1d	-	0m	-	1d	-	on track	-
DOVE-2	Buy a dove	1d	-	0m	-	1d	-	on track	-
	Total		**4w 2d**		**2w**		**2w 2d**		**on track**

- Additionally, both the Time Tracking report and the Version Workload report now include options for choosing which sub-tasks you would like to include in your reports.

Multi-project 'Road Map' portlet

The new Road Map portlet is a handy addition to your JIRA dashboard. It shows upcoming project milestones (i.e. versions which are due fo release within a specified period of time), and a summary of progress made towards completing the issues in those versions.



Road Map: **Next 30 Days** (Until 14/Oct/07)

Dove : Version 1 — 01/Oct/07

0 of 3 issues resolved

Test : Version 6 — 08/Oct/07

3 of 6 issues resolved

You can:

- Click the name of a project (e.g. 'Dove') to browse the project.
- Click the name of a version (e.g. 'Version 1') to browse the version.
- Click the progress bar (shown in red and/or green) to view the version's issues in the Issue Navigator.

Performance improvements

JIRA 3.11 includes some significant performance tuning which should improve the experience of every JIRA user.

- Page size has been reduced.
- The effect of GZip compression has been improved due to optimised handling of Javascript and CSS. This will be of benefit to people using Firefox or Internet Explorer 7 browsers.
- Caching has been optimised (see the Developer Blog).

Indexing improvements

- 'Bulk operations' now re-index issues one at a time instead of all at once. This allows JIRA to better handle concurrent operations and higher user loads while maintaining index integrity.
- JIRA now uses Lucene 2.2.0. This has two main benefits:
  - JIRA can now handle the input of and search on dates before January 1st, 1970.
  - JIRA now performs atomic updates to issue and comment indexes, providing greater consistency when searching.

JIRA Labels Plugin

Alongside JIRA 3.11, we're announcing a major overhaul of the JIRA labels plugin. The plugin implements a labels (or tags, sometimes known as folksonomy) custom field for JIRA. Labels or tags make it easier to organise a large set of data by arbitrary, user-defined criteria.

You can read more details on the developer blog.

Plus over 70 other fixes and improvements

JIRA Issues (90 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-15056		Editing Version info adds incorrect schedule data .		Resolved
JRA-14086		Setup page is accessable after JIRA instance has been setup already		Resolved
JRA-13864		Error is thrown when session timeout expired on editing a comment		Resolved
JRA-13560		Getting permgen OOME on eac/jira during startup		Resolved
JRA-13557		Create Labels build for 3.11		Resolved
JRA-13556		setHeader(null) causing NullPointer on Resin		Resolved
JRA-13510		An extra warning info for the JIRA standalone installation page.		Closed
JRA-13502		Sub-task creation fails when browse project permission is give to CurrentAssignee, GroupCF or UserCF		Resolved
JRA-13500		Upgrade urlrewritefilter		Resolved
JRA-13496		Error message generated in Add Comment Panel when adding comment to an issue		Resolved
JRA-13491		Only include calendar javascript and css on pages that require the calendar		Resolved
JRA-13490		Ship minified versions of javascript and css files		Resolved
JRA-13489		Don't use scriptaculous loader		Resolved
JRA-13488		Remove dashboard dependency on scriptaculous		Resolved
JRA-13466		TransitionWorkFlow Jelly Tag doco has a bad example attribute - commentLevel		Closed

JRA-13443	Subtask quick create submit can submit multiple times with multiple clicks		Resolved
JRA-13435	Base URL ending in slash breaks filter subscription HTML email links		Resolved
JRA-13425	Create Road Map Portlet		Resolved
JRA-13422	German translation for "Log work done" ist bad		Resolved
JRA-13417	'None' hard-coded in /templates/plugins/fields/edit/edit-multiselect.vm : should be internationalized		Resolved
JRA-13412	TransitionWorkflow Jelly tag will not work if there is no transition screen		Resolved
JRA-13411	JIRA XML import does not correctly ignore the unicode non-characters \uFFFF and \uFFFE		Resolved
JRA-13408	Pop up History/Filters windows when clicking History/Filters if they are open but under the main window		Resolved
JRA-13387	Upgrade atlassian-extras for the VSS plugin		Resolved
JRA-13382	combined.css and js files are loaded from the cache even after an upgrade of JIRA		Resolved
JRA-13367	Absence of an identifier on the comment element in per-issue generated XML		Resolved
JRA-13364	Wrong onClick location. Differs from href links.		Closed
JRA-13354	Subtask quick creation breaks when spaces available in between the property values.		Resolved
JRA-13345	Jira RSS 2.0 does not work with standard java parser - pubDate elements are incorrect		Resolved
JRA-13343	Problems with 'raw' rssMode when producing XML view of issue filter (in single-xml.vm)		Resolved
JRA-13335	Remove 'back to previous view' link on the printable view from printable media		Resolved
JRA-13332	Concurrent modification exception in com.atlassian.jira.web.tags.UserTag		Resolved
JRA-13319	Remove a link to Excel view from printable view of Time Tracking report		Resolved
JRA-13318	Hardcoded English terms on reports		Resolved
JRA-13295	JIRA standalone does't run as a Windows service - Failed creating java ... jvm.dll		Resolved
JRA-13291	Modify Time Tracking report to include aggregate time information		Resolved
JRA-13260	Make SOAP addWorklog method return the id of the created worklog		Resolved
JRA-13249	JIRA Turkish Language Property Files		Resolved
JRA-13248	Make email address in Support Request success/error page configurable		Resolved
JRA-13243	"Your Watches" translated as "Your Spies" in French....		Resolved
JRA-13227	Hide the priority of linked issues if the priority field is hidden		Resolved
JRA-13219	IntegrityChecker crashes with missing Portlet data.		Resolved

JRA-13210	Description in "Delete Issues" permission is inaccurate		Resolved
JRA-13179	ExceptionInInitializerError executing PopService		Resolved
JRA-13172	Separate out the searchers from the ThreadLocalQueryProfilingFilter		Resolved
JRA-13154	Upgrade to Lucene 2.x		Resolved
JRA-13138	Source release build needs to specify the 'Source' release info by default		Resolved
JRA-13113	BulkEditUserGroups: Provide helpful technique to prune out erroneous entries		Resolved
JRA-13110	SOAP/RPC getIssueTypes() should accept Project ID		Resolved
JRA-13109	Add a license file for the mindprod CSV parser to every distribution of JIRA		Resolved
JRA-13102	Calculate description text field lenght instead of hardcoding to 40 chars		Resolved
JRA-13085	Worklog Service should not allow lightweight issues to be passed in and persistened. It wrecks the index view of the world		Resolved
JRA-13083	Enabling "External user management" should not disable "View Project Roles" on user		Resolved
JRA-13069	Project Administrators do not get to edit "System Default Field Configuration "		Resolved
JRA-13057	UnsupportedOperationException with hasPermissionToCreate when called with DocumentIssueImpl		Resolved
JRA-13054	Display all installed languages and highlight default on the 500, system info and support request page		Resolved
JRA-13049	JIRA crashes when subscribing to a filter, and not logged in.		Resolved
JRA-13044	Upgrade EasyMock and DynaMock libraries		Resolved
JRA-13033	Make font size smaller for version / component descriptions on Versions / Components browse project tabs		Resolved
JRA-13017	Ensure SearchParameters can handle non-GV values in constructor.		Resolved
JRA-12985	Jelly doco for Create Issue tag says that default value of reporter tag is the current logged in user. But it ain't that simple.		Closed
JRA-12948	Incorrectly reporting Installation type as EAR/WAR instead of Standalone when running as a Windows service		Resolved
JRA-12925	HTML issue event notification emails render poorly in Outlook 2007		Resolved
JRA-12917	Improve on-line documentation on move permission		Resolved
JRA-12912	CommentService validation methods do not check user's security level		Resolved
JRA-12868	Customfield User Picker "corrupted" after a user is deleted		Resolved
JRA-12864	Trivial UI bug		Resolved
JRA-12863	Malformed Ressource Bundle properties files		Resolved
JRA-12839	When session expires and a login is forced the add commet operation loses the original comment text		Resolved

JRA-12837	Bugzilla importer breaks when summary contains over 255 characters		Resolved
JRA-12807	Create clear docs explaining options for import/export project		Closed
JRA-12790	Allow versions to be rescheduled to an arbitrary position on the Edit Version Details page		Resolved
JRA-12775	investigate getting JIRA working on Tomcat 6		Resolved
JRA-12687	Need to add Tomcat 6.0 documentation		Closed
JRA-12584	Submitting a Support Request via the admin interface should verify that the mail server has been configured correctly		Resolved
JRA-12469	Improving UrlRewrite to better handle HTTP request parameters		Resolved
JRA-12354	Version Control shows logs of wrong issue		Resolved
JRA-12336	Remove MailSender (no longer used)		Resolved
JRA-11877	Automated JIRA backup failes without proper warning		Resolved
JRA-11588	Replace usage of deprecated class DateField with DateTools		Resolved
JRA-10461	Deleting an issue type breaks custom fields that had used it		Resolved
JRA-10353	Enable GZip compression on SOAP interface		Resolved
JRA-10326	JIRAs HTTP transfers are not efficient - improve web performance		Resolved
JRA-10254	Warning message when an issue is created with 0 bytes or corrupted file		Resolved
JRA-9461	Increase the scope of the gzip compression to include JS and CSS files		Resolved
JRA-9103	Investigate support for pre-1970 dates within JIRA		Resolved
JRA-6332	Comments override updated date in CSV importer		Resolved
JRA-6249	Add more custom icons for issue constants		Resolved
JRA-6007	Can not delete double-byte group in Group Browser		Resolved
JRA-3009	Calculate issue estimates using subtask estimates		Resolved

JIRA 3.11 Upgrade Guide

- Upgrading from JIRA 3.10.x to 3.11
- Upgrading from JIRA 3.9.x and earlier

Upgrading from JIRA 3.10.x to 3.11

Please follow the JIRA general upgrade instructions, plus note the following:

Administrative notes

- To take advantage of the performance enhancements in JIRA 3.11, it is recommended that you enable GZip compression (unless you are using mod_proxy).
- The jira-application.properties file has a new option, 'progress', for the following attribute:

```
jira.table.cols.subtasks
```

The 'progress' option controls the display of the '**Progress**' field in issues and reports.

- JIRA 3.11 introduces a bug fix for JRA-12354. This means that the CVS and Perforce plugin will perform better at detecting commits for a particular issue key, avoiding partial matches on similar project keys. If users have taken advantage of the previous relaxed key matching, they can revert to the old behaviour by simply setting the following application property in the jira-application.properties file and restarting JIRA:

```
jira.option.key.detection.backwards.compatible=true
```

Plugins

Updating plugins
If you are using any of the following plugins, you will need to update them to their latest versions when performing the upgrade:

- Perforce plugin
- Subversion plugin
- Toolkit Plugin
- Charting Plugin
- RPC Plugin

3rd Party and personal plugins may also be affected (esp. if using lucene to store dates). These will need to be updated as well.

If these are updated after the upgrade (instead of as part of the upgrade), you will need to do a reindex.

A failure to update these plugins will result in lots of errors that look like:

Error 1

```
2007-07-25 15:23:27,553 http-8090-Processor4 ERROR
[charting.charts.createdvsresolved.CreatedVsResolvedChart] Could not create velocity parameters
For input string: "20070725144811"
java.lang.NumberFormatException: For input string: "20070725144811"
 at java.lang.NumberFormatException.forInputString(NumberFormatException.java:48)
 at java.lang.Long.parseLong(Long.java:415)
 at org.apache.lucene.document.DateField.stringToTime(DateField.java:100)
 at org.apache.lucene.document.DateField.stringToDate(DateField.java:104)
 at
com.atlassian.jira.ext.charting.data.DatePeriodStatisticsMapper.getValueFromLuceneField(DatePeriodS
at
com.atlassian.jira.ext.charting.data.OneDimensionalObjectHitCollector.adjustMapForValues(OneDimensi
at
com.atlassian.jira.ext.charting.data.OneDimensionalObjectHitCollector.collect(OneDimensionalObjectH
at org.apache.lucene.search.IndexSearcher$1.collect(IndexSearcher.java:137)
 at org.apache.lucene.search.Scorer.score(Scorer.java:49)
 at org.apache.lucene.search.IndexSearcher.search(IndexSearcher.java:146)
 at org.apache.lucene.search.Searcher.search(Searcher.java:118)
 at
com.atlassian.jira.issue.search.providers.LuceneSearchProvider.search(LuceneSearchProvider.java:111
```

Error 2

```
Caused by: java.lang.NoSuchMethodError:
org.apache.lucene.document.Document.add(Lorg/apache/lucene/document/Field;)V
 at com.atlassian.jira.plugin.labels.LabelSearcher.index(LabelSearcher.java:95)
 at
com.atlassian.jira.issue.index.indexers.impl.DefaultCustomFieldIndexer.addIndex(DefaultCustomFieldI
at com.atlassian.jira.issue.index.IssueDocument.getDocument(IssueDocument.java:34)
 at
com.atlassian.jira.issue.index.IssueDocumentBuilderImpl.get(IssueDocumentBuilderImpl.java:14)
 at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$IssueAndCommentCreator.handleIssueIndexin
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$AbstractIssueAndCommentHandler.indexIssue
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer.indexIssuesAndComments(SingleThreadedIssu
at
com.atlassian.jira.issue.index.MultiThreadedIssueIndexer.indexIssuesAndComments(MultiThreadedIssueI
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$2.perform(SingleThreadedIssueIndexer.java
at
com.atlassian.bonnie.ConcurrentLuceneConnection.withWriter(ConcurrentLuceneConnection.java:296)
 at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$1.perform(SingleThreadedIssueIndexer.java
at
com.atlassian.bonnie.ConcurrentLuceneConnection.withWriter(ConcurrentLuceneConnection.java:296)
 at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer.indexIssues(SingleThreadedIssueIndexer.ja
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$6.perform(SingleThreadedIssueIndexer.java
```

If you see these errors, please ensure that you are using the latest compatible version of the plugin for 3.11. If there is no supported version for 3.11, please contact the plugin developer via the plugin's homepage.

Developer notes

Modification to SOAP clients
If you have written a SOAP client for any JIRA version prior to 3.11 and are invoking any methods to get `RemoteIssueType` you will encounter the bug JRA-13529. The reason for this is that we have added extra information to the `RemoteIssueType` object that indicates if the issue type is a subTask issue type. To avoid the problem you will need to regenerate your remote object stubs against the updated JIRA 3.11 wsdl.

If you would like your SOAP client to work against multiple versions of JIRA then you need to use the latest stubs that have been generated against JIRA 3.11. You will need to **not** use any of the new functionality and you will need to remember that the `isSubTask` variable in the `RemoteIssueType` objects will be defaulted to `false`.

ThreadLocalQueryProfiler searchers have been moved to ThreadLocalSearcherCache
There may be a number of plugins that reference the ThreadLocalQueryProfiler searcher methods directly. These need to now reference the ThreadLocalSearcherCache.

Lucene Upgrade
We upgraded our version of Lucene to 2.2. If your plugin uses to Lucene to index/read data, please ensure that it works with JIRA 3.11. If you are indexing/reading dates, more than likely it will have broken and you will need to use the new Lucene 2 methods.

Database changes

There were no database changes in this release.

Upgrading from JIRA 3.9.x and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.10 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian is proud to announce **JIRA 3.10**, the latest release of our award winning issue tracking, workflow and project management software.

Major new features include:

- Editable worklogs
- 'Start Date' for worklogs
- New ways to browse Components and Versions
- AJAX-based 'User-picker' and 'Issue-picker'

This release also includes several bug fixes.

To see a list of *all* new features and improvements in this release — ask JIRA!

JIRA 3.10 is a free upgrade for any customer who purchased/renewed JIRA after the 9th of July, 2006. This release can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 3.10 Upgrade Guide.

Thank you for your feedback

Thank you to all the people who help us improve our products by 'voting' and providing ongoing feedback about what is important to you. JIRA 3.10 resolves nearly 360 of your votes.

In particular, our thanks to all those who voted for JRA-2411 (Ability to edit and remove work logs) and JRA-1959 (Allow date selection for work log). It gives us great satisfaction to deliver these features to you, and we hope you will find them useful (we know we will!)

Editable worklogs

Yes, worklogs can now be edited and deleted — with the issue's 'Time Spent' and 'Remaining Estimate' being adjusted appropriately in both cases.

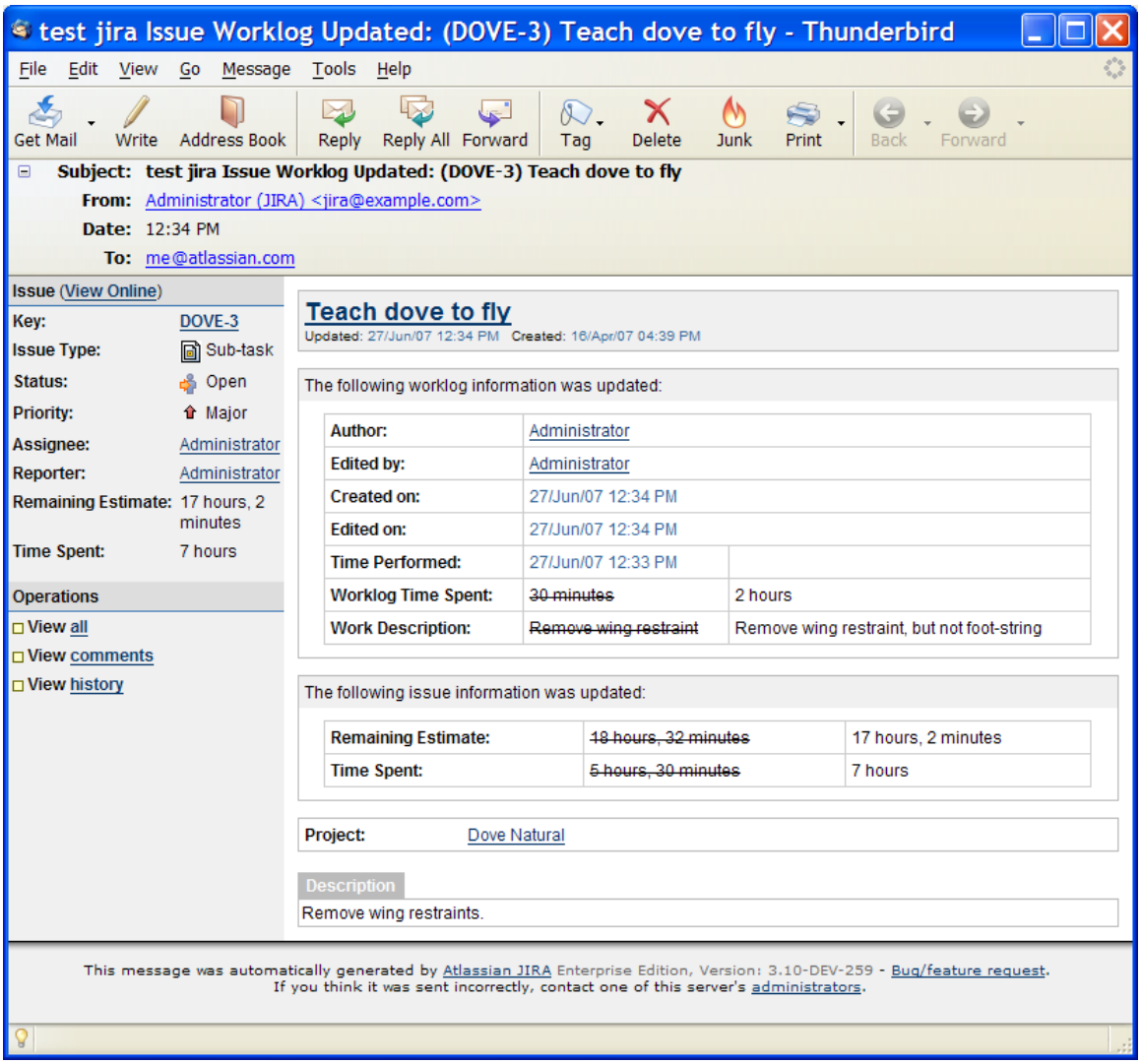
To ensure that only appropriate people can edit/delete worklogs, four new permissions have been added:

- 'Edit Own Worklogs'
- 'Edit All Worklogs'
- 'Delete Own Worklogs'
- 'Delete All Worklogs'

Similarly, to ensure that email notifications only get sent to the appropriate recipients, there are two new events:

- 'Issue Worklog Updated'
- 'Issue Worklog Deleted'

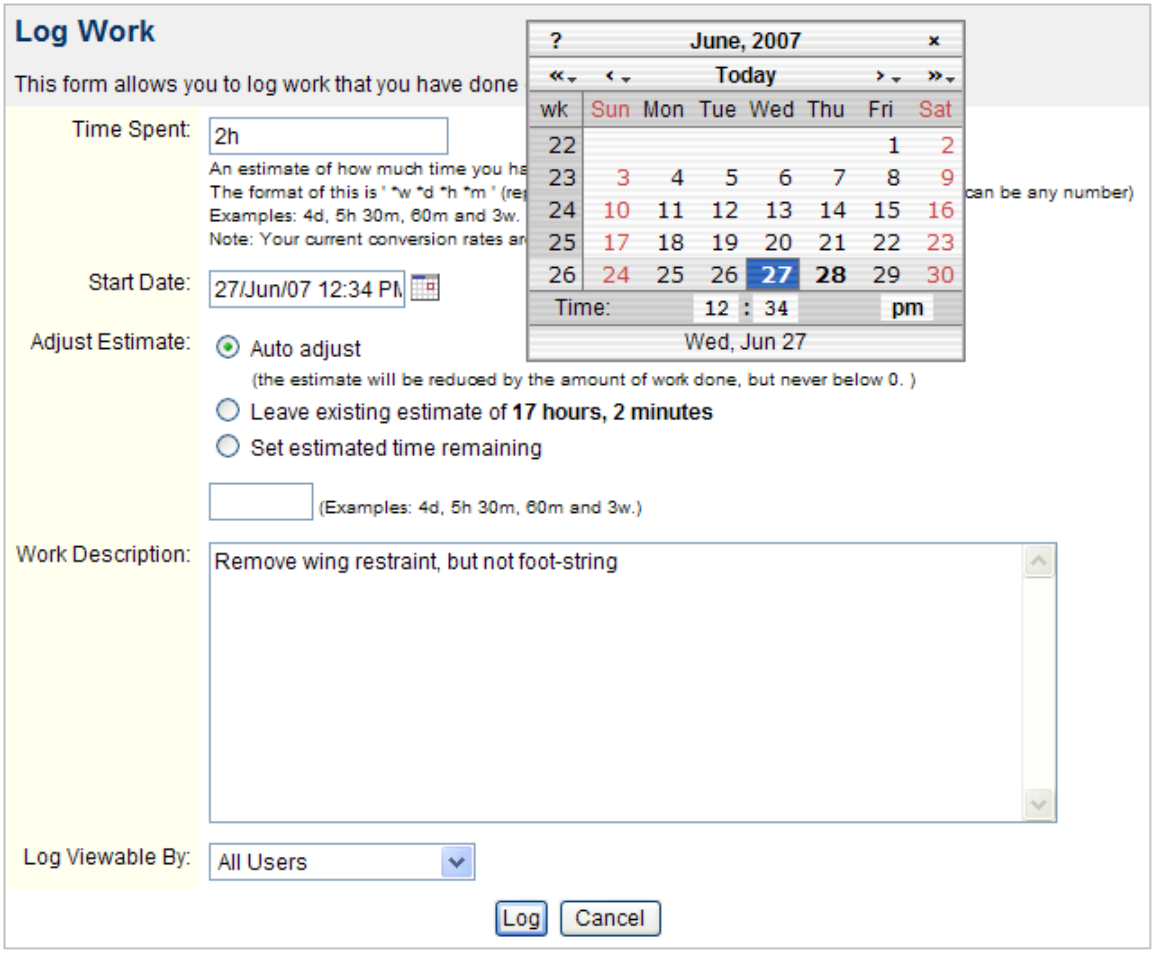
The email notification for an edited worklog looks like this:



All Worklog functions (create, retrieve, update and delete) are also available via the SOAP interface.

'Start Date' for worklogs

When logging work on an issue, you can now specify a 'Start Date'. Simply click the calendar icon to select the date/time when you started work. The calendar popup will be displayed:



... where you can:

- scroll back ('<') or forward ('>') to choose a different date.
- click the hour to increase it (or <Shift> click to decrease it).
- click the minute to increase it (or <Shift> click to decrease it).
- click 'am' / 'pm' to toggle between them.

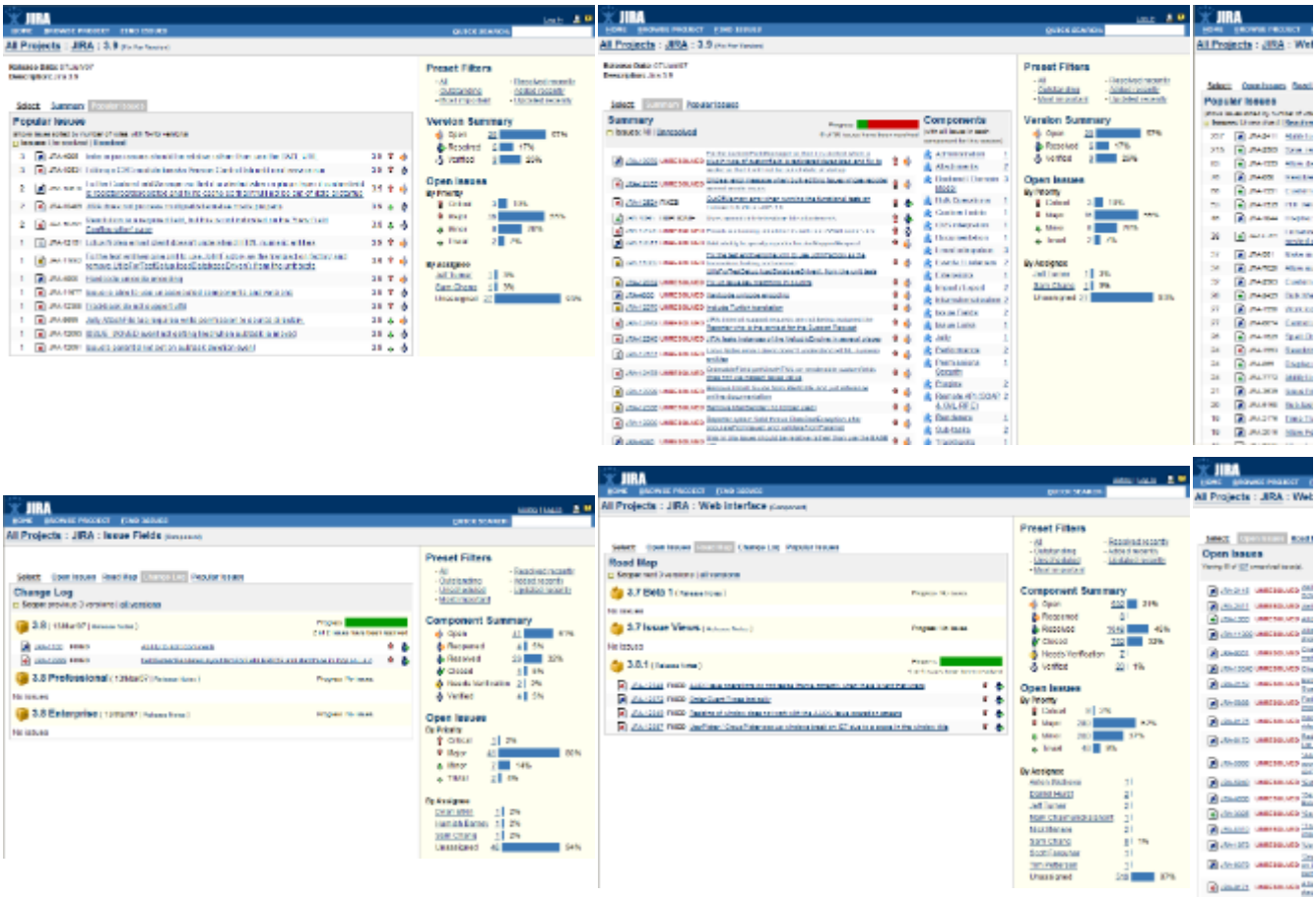
New ways to browse Components and Versions

We're all familiar with browsing a project to see a list of 'Open Issues', 'Popular Issues', and various other screens showing you important statistics about your project.

Now you can drill-down to an individual component or version of a project, by browsing a:

- Component's 'Open Issues'
- Component's 'Road Map'
- Component's 'Change Log' (i.e. resolved issues)
- Component's 'Popular Issues'
- Version's 'Summary' (i.e. all issues for that version, regardless of issue status)
- Version's 'Popular Issues'

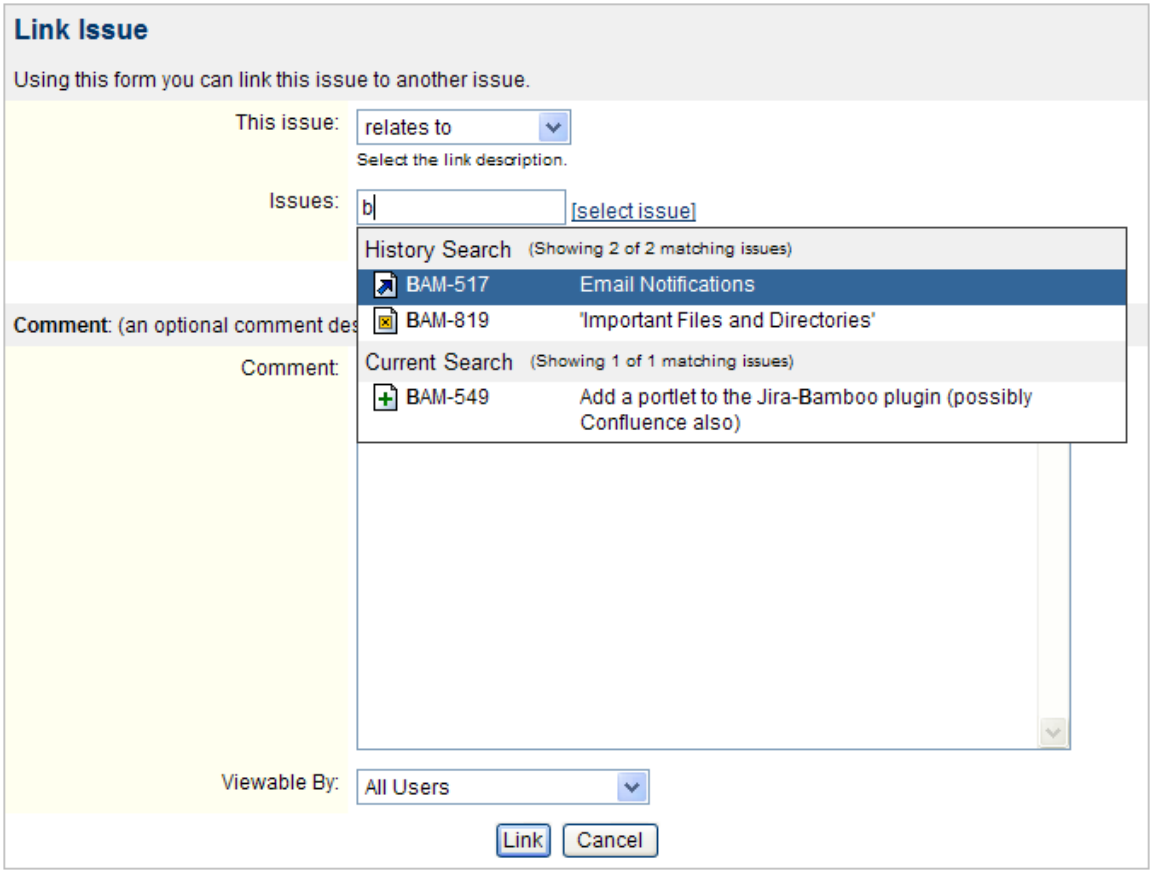
You can give this a try right now on JIRA. Just click on a component or version you're interested in!



The information on the new Component and Version summary pages is displayed using the Component Tab Panel and Version Tab Panel plugins. See the plugin types in the JIRA Plugin Guide for more information.

Auto-complete 'User-picker' and 'Issue-picker'

The 'Issue-picker' and 'User-picker' now have an AJAX-based auto-completion feature:



You're now able to simply start typing a user's name, or an issue's key or summary, and JIRA will provide a drop-down list of possible matches for you to select from. This should make selecting users and issues a lot quicker as you no longer need to click on the 'User-picker' icon or the '[select issue]' link and wait for the relevant pop-ups. The 'Issue-picker' will find matches within your latest search, as well as any matching issues you've been browsing recently.

This feature is enabled by default (though not for the 'User-picker' if you have more than 5,000 users).

If you wish to disable this feature (e.g. if you have very large numbers of users, or if your users' browsers are incompatible with AJAX), you can easily do so at the 'General Configuration' screen.

^Top

JIRA 3.10 Upgrade Guide

Upgrading from JIRA 3.9.3 to 3.10

Please follow the JIRA general upgrade instructions, plus note the following:

1. Plugins

There is a new version of the JIRA Calendar Plugin that links to the new 'Project Version' pages. This new version of the plugin is not backwards compatible.

Please note that the Kaamelot plugin for JIRA has not yet been updated. If you are currently using this plugin, you may want to hold off the upgrade to JIRA 3.10 until a compatible version of this plugin has been released.

2. Developer Notes

The ordering of the ListOrderedMap returned by SchemePermissions.getSchemePermissions() has changed. This also means that the order of the RemotePermission[] array returned by the RPC Plugin's JiraSoapService.getAllPermissions() method has changed. If you have extended your instance of JIRA please confirm that any remote applications retrieving permissions via SOAP still work. You may encounter problems if you have been retrieving specific permissions by their array index.

Database changes

In JIRA 3.10, the worklog records have moved from the 'jiraactions' database table to the new 'worklog' table. This new table contains the following columns:

```
Table "public.worklog"
    Column    |           Type           | Modifiers
--------------+--------------------------+-----------
 id           | numeric(18,0)            | not null
 issueid      | numeric(18,0)            |
 author       | character varying(255)   |
 grouplevel   | character varying(255)   |
 rolelevel    | numeric(18,0)            |
 worklogbody  | text                     |
 created      | timestamp with time zone |
 updateauthor | character varying(255)   |
 updated      | timestamp with time zone |
 startdate    | timestamp with time zone |
 timeworked   | numeric(18,0)            |
```

Upgrading from JIRA 3.9.2 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.10.2 Release Notes

JIRA 3.10.2 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.10.2 in Standard, Professional and Enterprise editions. This point release includes 24 bug fixes and improvements.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.10.2 can be downloaded here, and is of course free to all customers who purchased their JIRA licence or maintenance since August 17, 2006.

If upgrading, please refer to the JIRA 3.10.2 Upgrade Guide.

Not using 3.10? Learn about all the new features you're missing out on!

JIRA 3.10.2 includes the following bug fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.10.2 Upgrade Guide

Upgrading from JIRA 3.10.1 to 3.10.2

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.9.3 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.10.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.10.1 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.10.1 in Standard, Professional and Enterprise editions. This point release includes 26 bug fixes and improvements.

JIRA 3.10.1 can be downloaded here, and is of course free to all customers who purchased their JIRA licence or maintenance since August 1, 2006.

If upgrading, please refer to the JIRA 3.10.1 Upgrade Guide.

Not using 3.10? Learn about all the new features you're missing out on!

JIRA 3.10.1 includes the following bug fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.10.1 Upgrade Guide

Upgrading from JIRA 3.10 to 3.10.1

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.9.3 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.9 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian is proud to announce **JIRA 3.9**, the latest release of our award winning issue tracking, workflow and project management software.

New features include:

- Ability to convert sub-tasks to issues (and vice versa)
- Convenient new scheduler for filter subscriptions
- Separate permissions for 'Delete Comment', 'Delete Attachment' and 'Delete Issue'
- Performance Improvements for Project Roles

This release also includes over 30 bug fixes.

To see a list of all new features and improvements — ask JIRA!

494 of your votes have been addressed in this release. As always, thank you for taking the time to cast your vote and tell us what is important to you. We appreciate your feedback.

JIRA 3.9 is a free upgrade for any customer who purchased/renewed JIRA after 9 May, 2006. This release can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 3.9 Upgrade Guide.

Convert sub-tasks to issues (and issues to sub-tasks)

In JIRA 3.9, sub-tasks can now be converted to issues, and vice versa.

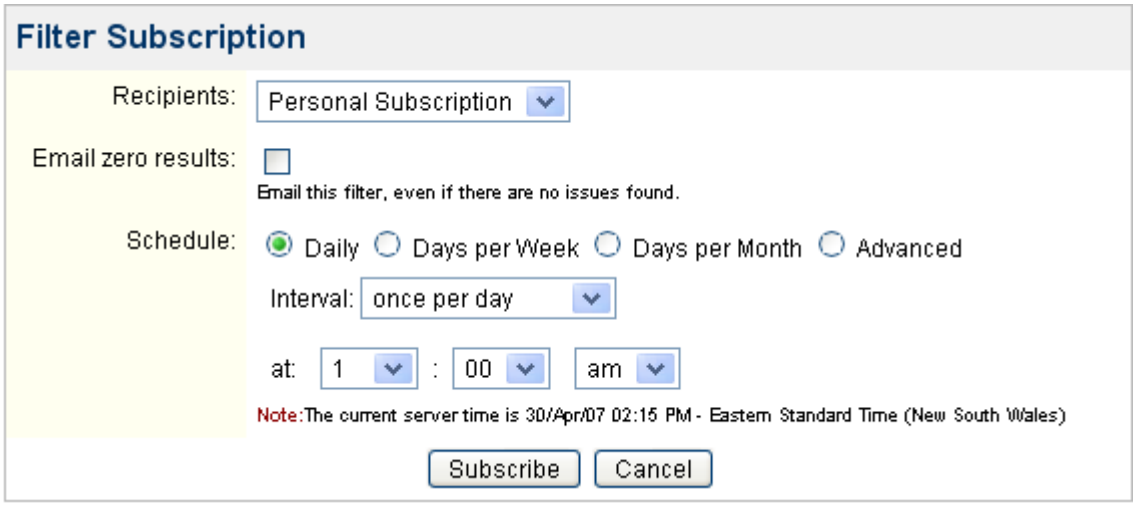
- Perhaps a particular sub-task has become important enough to be an issue in its own right, with its own sub-tasks? Simply go to the sub-task and click 'Convert to Issue'. You can now create sub-tasks for the converted issue.
- Or perhaps an issue should really have been created as a sub-task of an existing issue. No problem: simply go to the issue and — you've guessed it — click 'Convert to Sub-task'.

^Top

Convenient new scheduler for filter subscriptions

If you like to have your search results emailed to you, you will be pleased to see the new and improved scheduler in JIRA 3.9.

It's now even easier to choose exactly when and how often you would like to receive your emails, e.g. 'Every day at 1.00am', 'Every hour between 9.00am and 5.00pm, Monday to Friday'.



For the adventurous, we have also included a cron-based scheduler.

^Top

Separate permissions for 'Delete Comment', 'Delete Attachment' and 'Delete Issue'

You can now give people the ability to delete comments and/or attachments, without giving them the ability to delete entire issues.

JIRA 3.9 has four new project-level permissions:

- 'Delete All Attachments': This permission gives the user the ability to delete any attachments, regardless of who added them.
- 'Delete Own Attachments': This permission gives the user the ability to delete attachments that they created.
- 'Delete All Comments': This permission gives the user the ability to delete any comments, regardless of who added them.
- 'Delete Own Comments': This permission gives the user the ability to delete comments that they created.

Project Role Permission check performance improvements

The performance of permission checks against project roles has been significantly improved.

This improvement allows much faster load times for pages such as Dashboard, especially when several users are hitting JIRA at the same time. The performance improvement is most noticeable with large numbers of projects containing large numbers of project role user members. See JRA-12610 for details.

Previously this check was a CPU intensive operation that involved the intermediate creation of many intermediate objects - and degraded badly under concurrent access. The operation is now performed in constant time per project (basically a hash lookup).

^Top

JIRA 3.9 Upgrade Guide

Upgrading from JIRA 3.8.1 to 3.9

Please follow the JIRA general upgrade instructions. Additionally, please note the following:

In this version, there has been a change to the database which may cause problems for some customers.

The Recommended Upgrade Method

If you follow the recommended export/import upgrade procedure you should not experience any problems!

Pointing JIRA 3.9 at an existing, non-empty database

Some customers have a good reason for not following the recommended upgrade method. Using this method may result in database errors in your logs. You can avoid this if you modify your table structure manually, but the procedure is different depending on whether you have already started JIRA.

To avoid this, **BEFORE** you upgrade JIRA using this method, you can just drop the qrtz_cron_triggers table. This table has not been used by JIRA before 3.9, so it should be empty.

If you have **ALREADY** started JIRA 3.9 using your existing database, you may see the following log messages when JIRA starts up:

```
2007-04-18 15:31:53,345 main WARN [core.entity.jdbc.DatabaseUtil] Column  "CRON_EXPERSSION" of
table "public.qrtz_cron_triggers" of entity  "QRTZCronTriggers" exists in the database but has
no corresponding field
2007-04-18 15:31:53,347 main WARN [core.entity.jdbc.DatabaseUtil] Entity  "QRTZCronTriggers"
has 3 fields but table "public.qrtz_cron_triggers"  has 4 columns.
```

The reason for this is that we have incorrectly changed a column in the `qrtz_cron_triggers` table. The intention was to fix a misspelling, but all we did was remove an underscore ("_")! The old column name is "CRON_EXPERSSION". The new column name is "CRONEXPERSSION". Note that both columns spell the word "expression" incorrectly.

To remove the error message, you must remove the old column as it is redundant. This column will not contain any data. The following table shows all columns in the qrtz_cron_triggers table. Columns that should be present are in **green** and columns that should be deleted are in **red**.

Keep	Keep	Keep	Delete
ID	TRIGGER_ID	CRONEXPERSSION	CRON_EXPERSSION

To delete the column, you can use SQL, but this may be slightly different between databases. Here's how it might look:

```
alter table qrtz_cron_triggers drop column CRON_EXPERSSION;
```

The data in this table

If you have users who have subscribed to issue filters, note that existing SimpleTriggers (time intervals) will be automatically converted into CronTriggers during the JIRA upgrade. In some cases, there may not be an exact mapping of time intervals to Cron Expressions, and approximations will be made (e.g. 'Every 5 weeks' will be converted to 'Once a month'). If this happens, the JIRA upgrade process will send an email to the user to inform them of the new schedule.

Upgrading from JIRA 3.8 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade

Guides is available here.

JIRA 3.9.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.9.3 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.9.3 in Standard, Professional and Enterprise editions. This point release includes:

- 7 bug fixes.
- professional French and German translations (see below)

JIRA 3.9.3 can be downloaded here, and is of course free to all customers who purchased their JIRA licence or maintenance since June 28, 2006.

If upgrading, please refer to the JIRA 3.9.3 Upgrade Guide.

Not using 3.9? Learn about the new features you're missing out on!

What's new in JIRA 3.9.3?

JIRA 3.9.3 includes the following bug fixes and improvements:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Professional French and German translations

The French and German language packs have been completely rewritten and are much more comprehensive than ever before. The administration sections of JIRA are now completely translated. To achieve this, we recently engaged a professional translation company to provide German and French versions of JIRA. These translations are now available in JIRA 3.9.3, and we hope they will make your experience with JIRA even better.

Thank you, danke and merci to all those people who have provided the previous translations over the years, and also to those who have recently been helping us to check the translations for style, consistency and correctness.

While we hope you enjoy the new more comprehensive translations, if the language changes are not ideal for you it is possible to use JIRA 3.9.3 with the old translations. Administrators can revert to the translations from JIRA 3.9.2 and earlier, simply by replacing the new language pack jar file with the corresponding jar file from the earlier version. The French jar file is `language_fr_FR.jar` and the German one is `language_de_DE.jar`, located in `atlassian-jira/WEB-INF/lib` in JIRA standalone.

JIRA 3.9.3 Upgrade Guide

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Upgrading from JIRA 3.9.2 to 3.9.3

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.9.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.9.2 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.9.2 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.9.2 in Standard, Professional and Enterprise editions. This point release includes 13 bug fixes.

JIRA 3.9.2 can be downloaded here, and is of course free to all customers who purchased their JIRA licence or maintenance since June 18, 2006.

If upgrading, please refer to the JIRA 3.9.2 Upgrade Guide.

Not using 3.9? Learn about all the new features you're missing out on!

JIRA 3.9.2 includes the following bug fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.9.2 Upgrade Guide

Upgrading from JIRA 3.9/3.9.1 to 3.9.2

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.8.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.9.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.9.1 Release Notes

Recommended Upgrade

JIRA 3.9.1 contains a security update and is highly recommended. This fix is related to issue level security schemes and a way that users can see details of issues that they are not meant to. If you do not have issue level security schemes or do not use Project Roles you do not need to worry. If you do we recommend you upgrade immediately.

Note that the bug list below does not contain details of the bug as it would reveal how to exploit it as well.

For installations running 3.7.x or 3.8.x who cannot upgrade to 3.9.1, there is a patch available.

Atlassian Software Systems is proud to announce the release of JIRA 3.9.1 in Standard, Professional and Enterprise editions. This point release includes 10 bug fixes and some internationalisation improvements.

JIRA 3.9.1 can be downloaded here, and is of course free to all customers who purchased their JIRA licence or maintenance since May 30, 2006.

If upgrading, please refer to the JIRA 3.9.1 Upgrade Guide.

Not using 3.9? Learn about all the new features you're missing out on!

JIRA 3.9.1 includes the following bug fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Important Security Patch for JIRA versions 3.7.x & 3.8.x

Please find attached an important security patch for JIRA 3.7.x and 3.8.x

If you are using Project Roles and have Issue Security schemes that use Project Roles and you cannot upgrade to 3.9.1 then **you must install this patch**.

Installation Instructions.

If you are using JIRA Standalone please do the following:

1. Download the attached patch zip file
2. Extract the contained files to <jira_install_dir>/atlassian-jira/WEB-INF/classes/ overwriting the files there
3. Restart JIRA

If you are using the WAR distribution of JIRA:

1. Download the attached patch zip file
2. Extract the contained files to <jira_install_dir>/atlassian-jira/WEB-INF/classes/ overwriting the files there
3. Run 'build.sh clean' on unix or 'build.bat clean' on windows
4. Run 'build.sh' on unix or 'build.bat' on windows
5. Redeploy the JIRA web app into your application server
6. Restart the application server

JIRA 3.9.1 Upgrade Guide

Upgrading from JIRA 3.9 to 3.9.1

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.8.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.8 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian is proud to announce **JIRA 3.8**, the latest release of our award winning issue tracking, workflow and project management software.

New features include:

- Editable comments
- Self-installer for JIRA
- CAPTCHA for new account signup
- Integration with Crowd
- Improvements to the Bugzilla importer

Feature preview:

- DHTML-loading of Issue screens

This release also includes more than 30 bug fixes.

To see a list of all new features and improvements — ask JIRA!

Weblogic Users

Please note that there is a known Weblogic and Firefox issue that affects JIRA 3.8.x when using Weblogic and Firefox. See the issue for more detail.

Upgrading

JIRA 3.8 is a free upgrade for any customer who purchased/renewed JIRA after 9 March, 2006. This release can be downloaded from the JIRA Download Center. Before upgrading, please refer to the JIRA 3.8 Upgrade Guide.

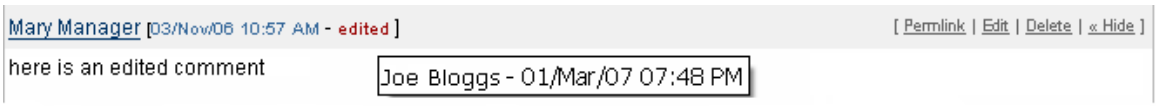
Editable comments

SPECIAL NOTE: Thank you to the 175 people who voted on this feature request. Your input is vital to planning the JIRA development roadmap, and we appreciate you taking the time to tell us what is important to you.

Issue comments can now be edited. To ensure that comments can only be edited appropriately, two new project permissions have been provided so that you can restrict the ability to edit comments:

- 'Edit Own Comments' -- this allows users to edit comments which they have created. This permission is typically granted to end-users.
- 'Edit All Comments' -- this allows users to edit comments which other people have created. This permission is typically granted to administrators.

If a comment has been edited, the word 'Edited' will appear in the comment trail. You can hover your mouse over the word 'Edited' to see who edited the comment and when, e.g.:



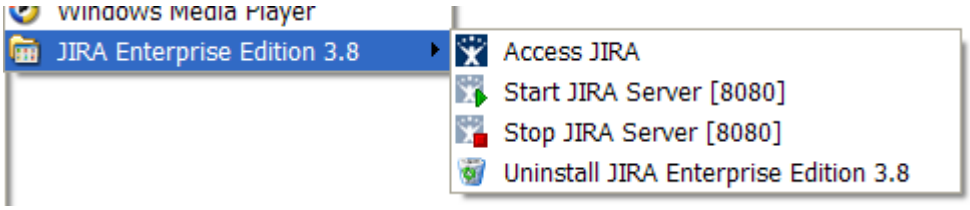
You can also configure email notifications to be sent when the 'Comment Edited' event occurs.

^Top

Self-installer for JIRA

New and existing customers can get the latest version of JIRA up and running on Windows within minutes, using our new installer. No more setting environment variables, installing Java, and running things from the command line. Even novice users will be able to access JIRA in as little as 5 clicks after the download.

For your convenience, we have even added controls to the Start Menu to make life as easy as possible:



We also bundle JIRA with the latest Sun JRE (Java 6.0), so there is no need for a separate download and installation. It all comes packaged and ready to run!

- Self extracting -- no need for Winzip or any other tools.
- Optional installation as a Windows Service.
- 'Start' and 'Stop' menu items, for both normal installation and Windows Service installation.
- Tested on Windows Vista.
- Detection of any other JIRA instances installed on the same machine.
- Port detection (detects if any other web servers are running on the machine and resolves conflicts).
- Uninstaller (not that you will be needing it).

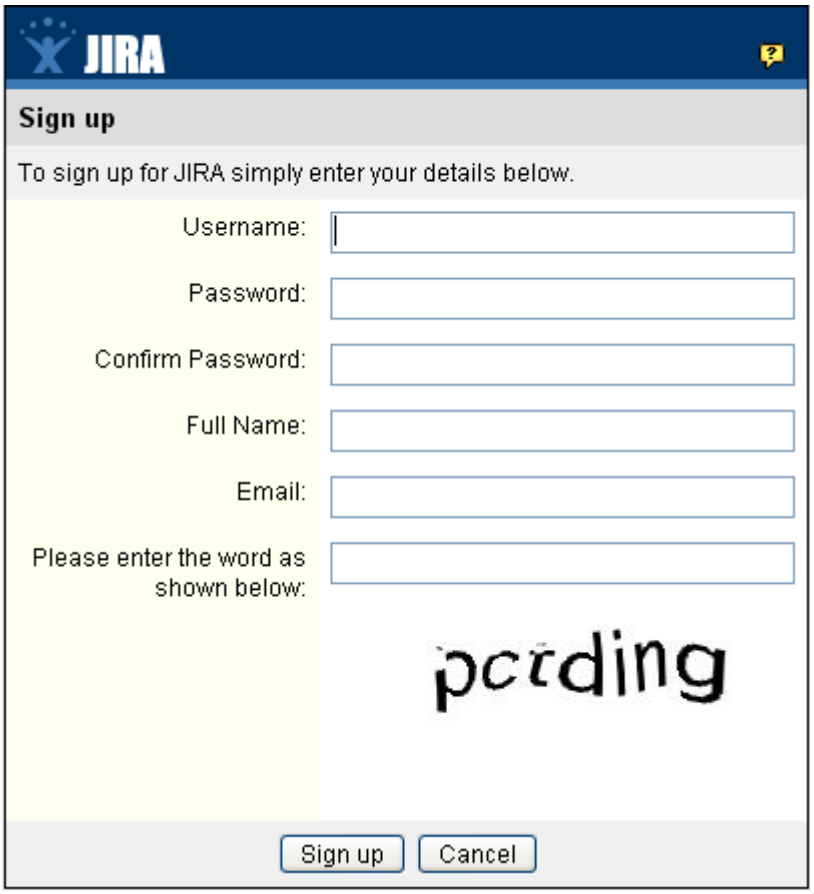
^Top

CAPTCHA for new account signup

If your JIRA server is accessible from outside your organisation's firewall, and you have enabled signup, then you may want to also enable

CAPTCHA.

CAPTCHA helps ensure that only real humans (and not automated spam systems) can sign themselves up to JIRA. When CAPTCHA is enabled, visitors will need to recognise a distorted picture of a word (e.g. "pctding" in the screenshot below), and must type the word into a text field. This is easy for humans to do, but very difficult for computers. We are hoping that this feature will help to fight evil JIRA spammers (see JRA-12293 for some of the background to this.)



We recommend anyone running a public JIRA instance (e.g. Codehaus, Apache, OpenSymphony) to enable this feature.

^Top

Integration with Crowd

JIRA can now be integrated with Atlassian Crowd, which is useful for organisations that have multiple user-repositories.

- How to integrate Crowd with JIRA

^Top

Improvements to the Bugzilla importer

JIRA's Bugzilla importer has been enhanced. When importing Bugzilla bugs and creating corresponding issues, JIRA will now:

- create Issue Links of type 'Duplicate' between issues that have been imported and marked as duplicates in Bugzilla. The 'Duplicate' link type will be automatically created if it doesn't exist.
- import Component Lead information.
- concatenate the 'URL' field (from Bugzilla) to the 'Environment' field in JIRA issues.

Many thanks to Vladimir Alexiev for his contributions.

^Top

DHTML-loading of Issue screens (Feature Preview)

To facilitate faster loading of issue screens (e.g. the "Edit Issue" screen and the "Resolve Issue" screen), we are working on a feature that will allow JIRA to re-load only those parts of the screen that have changed. We hope this will save a little of your valuable time, and improve your experience with JIRA.

This feature is shipped in JIRA 3.8, but as it has a few known problems (JRA-12348 and JRA-12349) it is disabled by default. The known

problems should not affect many users, so we encourage you to turn it on and provide any feedback by adding comments to this page. We would especially like to hear if you believe the feature is useful or if you find any problems that we are not aware of.

To enable the feature, please navigate to Administration -> General Configuration and enable the 'Dynamic HTML for issue screens' option.
^Top

Feedback for DHTML-loading of Issue screens

Please add any feedback you have about the 'DHTML-loading of Issue screens' in JIRA 3.8 as a comment to this page.

We would be very interested to know whether you think the feature is useful and hear about any problems that you find.

Currently we know about the following issues:

1. JRA-12348
2. JRA-12349

JIRA 3.8 Upgrade Guide

Upgrading from JIRA 3.7.4 to 3.8

Please follow the JIRA general upgrade instructions. Additionally, please note the following:

1. The 'Assign To' field name has been changed to 'Assignee' consistently across JIRA. This means that users need to be aware that the column heading in the Excel export has changed to 'Assignee' from 'Assign To'. Please be aware of this if for example you are exporting JIRA data to Excel and running macros on it. The field has been renamed for the following Issue Navigator Views:
    - Excel (all)
    - Word (all)
    - Full Content
2. The `issuecommentedited.vm` e-mail template for the new Issue Comment Edited event has been added to the `WEB-INF/classes/email-template-id-mappings.xml` file. The id of the e-mail template used for sending Filter Subscriptions has changed to 10000. **If** you have manually modified the `WEB-INF/classes/email-template-id-mappings.xml` file in the version of JIRA you are upgrading from, please do not simply copy the old file to JIRA 3.8. You will need to port your changes to the `WEB-INF/classes/email-template-id-mappings.xml` file that is shipped with JIRA 3.8. If you have not changed the `WEB-INF/classes/email-template-id-mappings.xml` file, you do not need to worry about this.
3. Two columns have been added to the `jiraaction` table to support editable comments.

Upgrading from JIRA 3.7.3 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.8 Database Schema Changes

The table below summarises the changes to the database schema. Please note, that if you have developed any custom utilities which query or modify the JIRA database directly (i.e. without using the JIRA API), please check whether the utilities need to be updated.

New Database Table Columns

The following database columns have been added to the existing `jiraaction` table to support editable comments:

TABLE NAME	NEW COLUMN NAME
jiraaction	updateauthor
jiraaction	updated

JIRA 3.8.1 Release Notes

JIRA 3.8.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.8.1 in Standard, Professional and Enterprise editions. This point release includes 23 bug fixes and improvements, notably:

- **Integrity Checker can cause data corruption** — - JRA-12491
- **SSH Connection to CVS in CVS plugin does not close stdout** — JRA-12480 particular thanks go to David Delbecq from the

Royal Meteorological Institute of Belgium for help finding this one.
- **OutOfMemoryErrors when reindexing if large numbers of custom fields and issues** — a thread-local cache for custom field values was expanding unbounded when reindexing all issues - JRA-12411

JIRA 3.8.1 can be downloaded here, and is of course free to all customers who purchased their JIRA licence or maintenance within the last 12 months.

If upgrading, please refer to the JIRA 3.8.1 Upgrade Guide.

Not using 3.8? Learn about all the new features you're missing out on!

JIRA 3.8.1 includes the following bug fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.8.1 Upgrade Guide

Upgrading from JIRA 3.8 to 3.8.1

Please follow the JIRA general upgrade instructions.

> **Charting Plugin must be upgraded to v1.3.5**
>
> Please note that the version of JFreeChart included in JIRA 3.8.1 is **not compatible** with older versions of the Charting Plugin. If you have the Charting Plugin installed, please make sure you upgrade it to version 1.3.5 or above.
>
> The updated JFreeChart 1.0.4 version is **not** backwards compatible with the previous 1.0.0pre2 version, so if you have any plugins that utilise JFreeChart, please make sure you test them before upgrading.

Upgrading from JIRA 3.7.4 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.6 Release Notes

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian is proud to announce the latest release of the award winning issue tracking and project mamagement software - **JIRA 3.6**. This release offers a range of new features and improvements throughout JIRA - from the introduction of custom events to wiki-style link aliasing - along with a number of bug fixes.

As always, this release can be downloaded from the JIRA Download Center - and dont forget to check out the upgrade guide if upgrading from a previous version!

Contents

- New Features
- Improvements
- Bug Fixes
- Upgrading

New Features

To see a full list of new features and improvements added - ask JIRA!

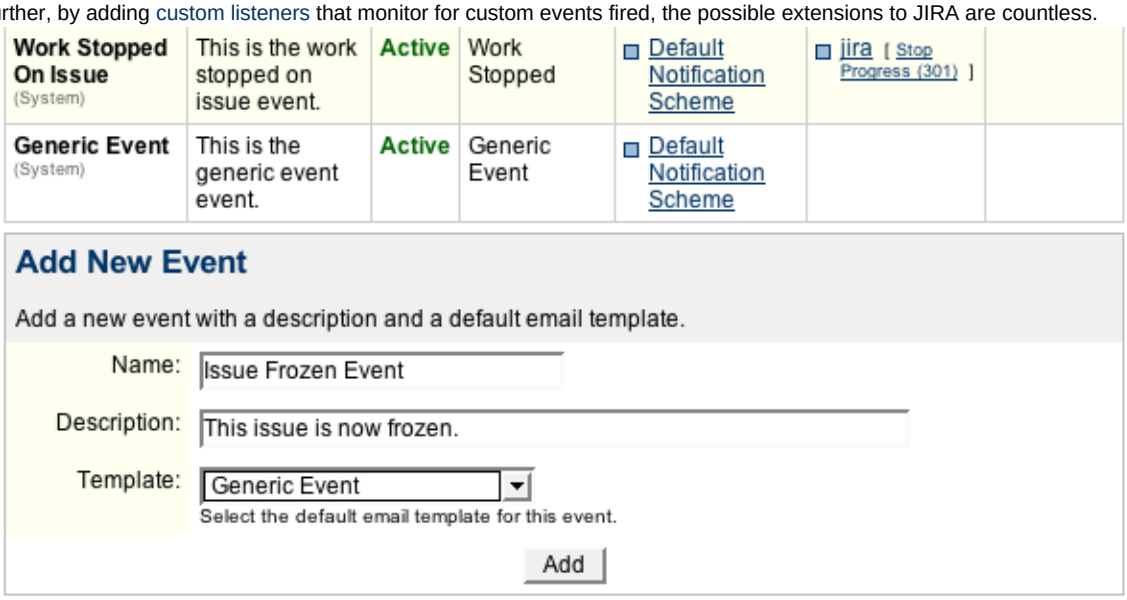
- Custom Events
- Group Picker Custom Field
- Per-Issue Group Notifications & Permissions
- Wiki-style Linking
- "I'm Feeling Lucky" Quick Search
- Collapsible Fields
- Nestable Conditions
- Charting Plugin Updates

Custom Events

JIRA uses an event-listener mechanism to alert the system that something has happened and allow it to perform an action based on that event. In both Professional and Enterprise editions, this release presents the ability to add custom events to the system - providing a pivotal extension point for notification and workflow schemes.

For instance, a custom event can be added to the system and associated with a workflow post-function - fired on completion of the event. A notification scheme can then be configured to email particular recipients once this event has been fired. With custom events, the notification and workflow schemes can be configured extensively to respond to specific custom events added to the system.

Further, by adding custom listeners that monitor for custom events fired, the possible extensions to JIRA are countless.

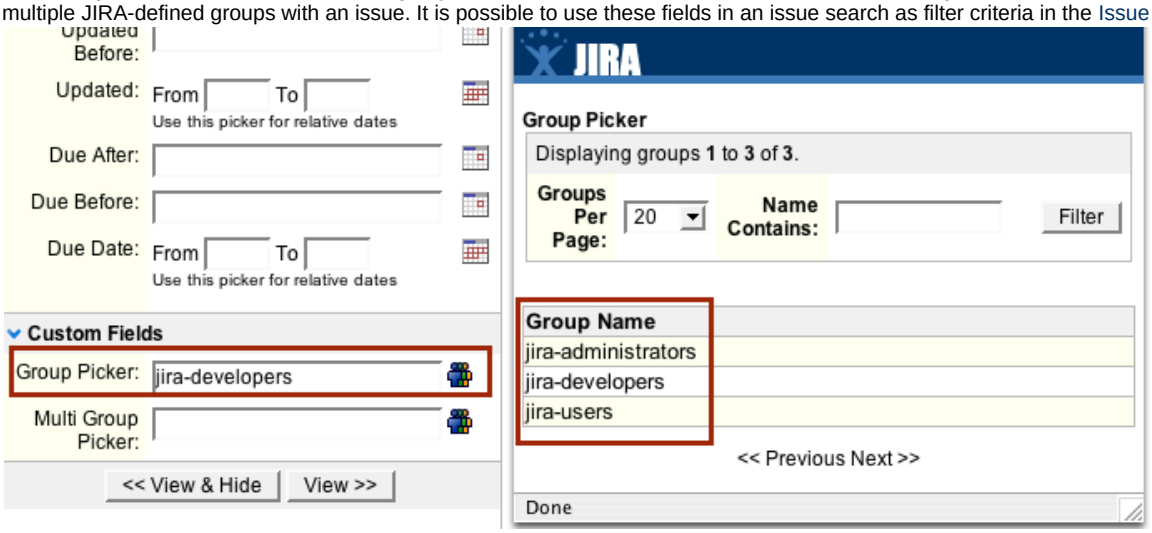


Group Picker Custom Field

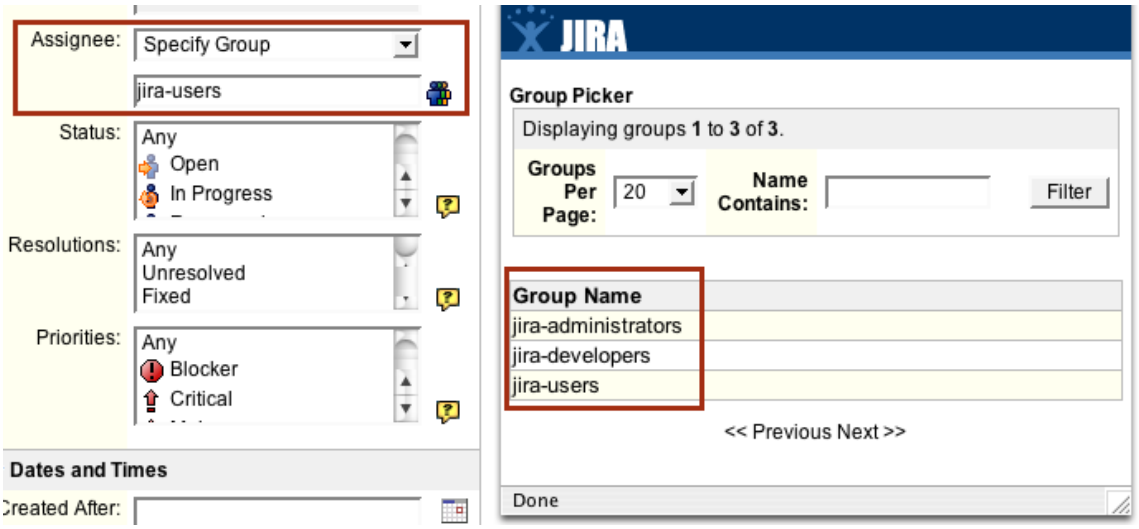
JIRA custom fields have become invaluable tools in extending the data associated with an issue. This release introduces two new custom fields into the fold:

- Group Picker
- Multi Group Picker

The **Group Picker** allows a JIRA-defined group to be associated with an issue while the **Multi Group Picker** allows the association of multiple JIRA-defined groups with an issue. It is possible to use these fields in an issue search as filter criteria in the Issue Navigator.



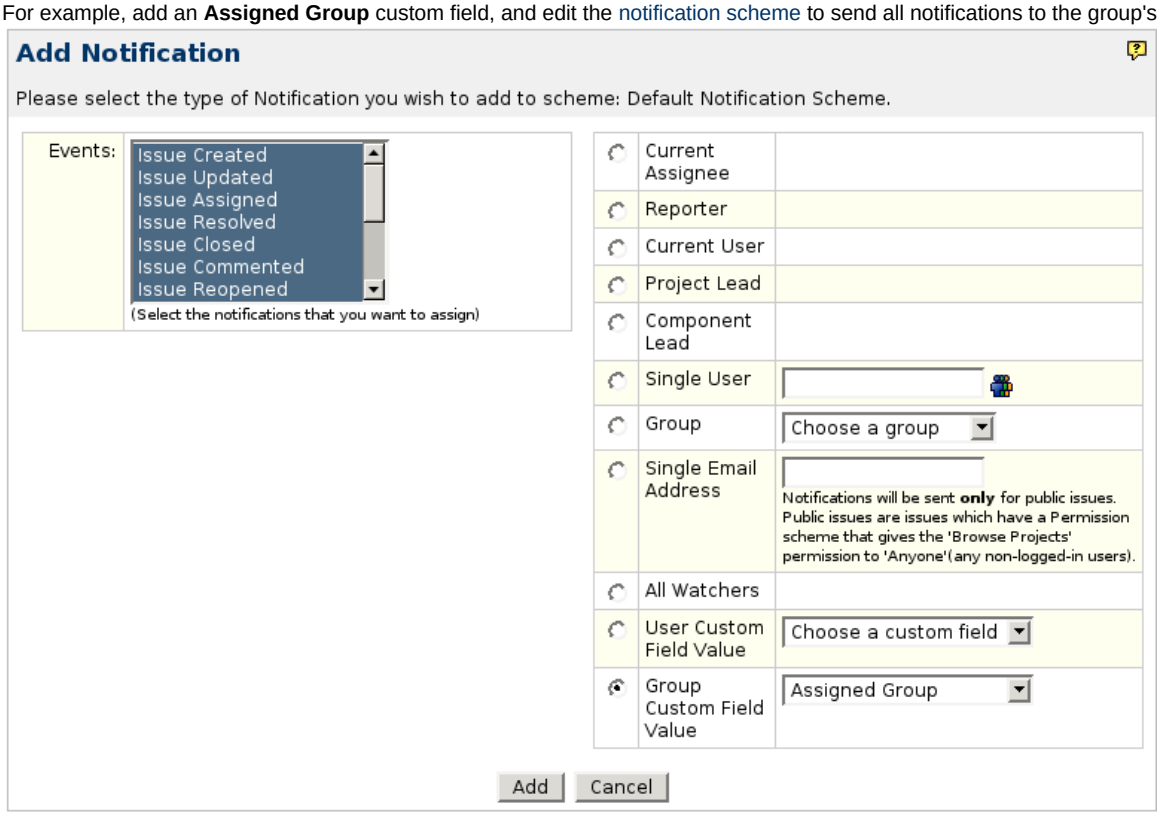
The assignee selector field in the Issue Navigator has also been updated with the Group Picker link - allowing the group to be selected from the available list.



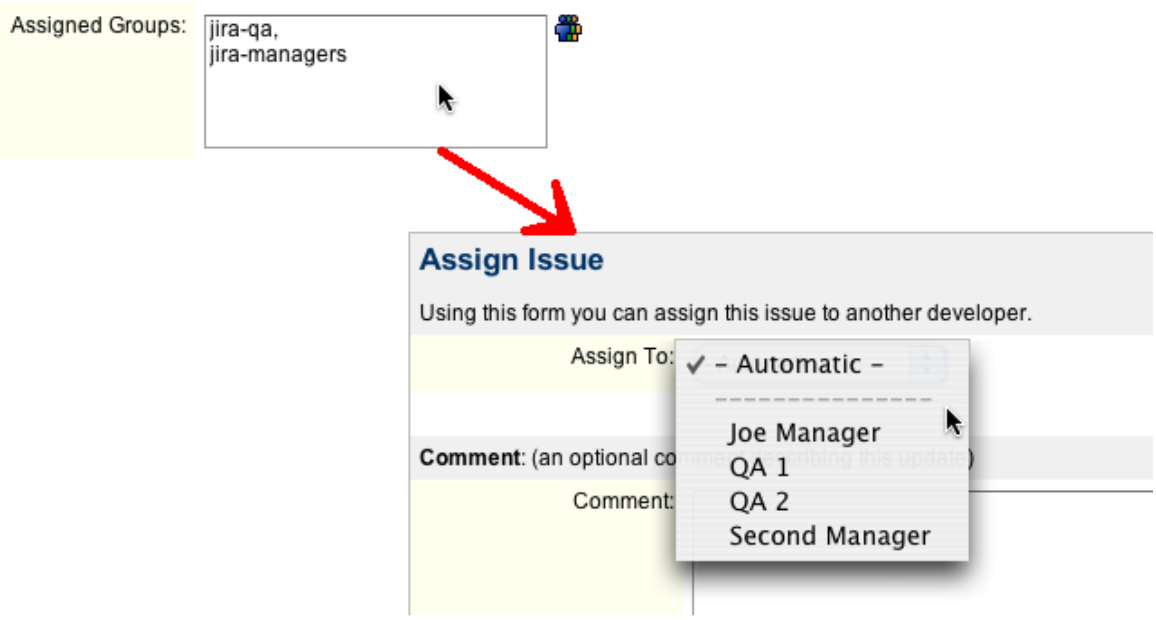
Per-issue Group Notifications and Permissions

Building on the group picker custom field, one can now send notifications to members of a group determined by a custom field.

For example, add an **Assigned Group** custom field, and edit the notification scheme to send all notifications to the group's members:



Now the "Assigned Group" on each issue will be notified of changes.
As with notifications, one can now grant a permission to members of a group(s) selected by a custom field. Continuing our **Assigned Group** example, by granting the **Assigned Group** the **Assignable** permission, the possible assignees of an issue will be the members the **Assigned Group** custom field group:



You can also restrict workflow operations to members of a custom field's group, with a custom field condition.

Preset group lists

Per-issue groups can also be chosen via select-lists, in addition to the group picker. Say you wish to restrict the **Assigned Group** to `jira-managers` or `jira-qa`. Simply create a select-list custom field with these two values, and add the select-list to the notification and permission schemes.

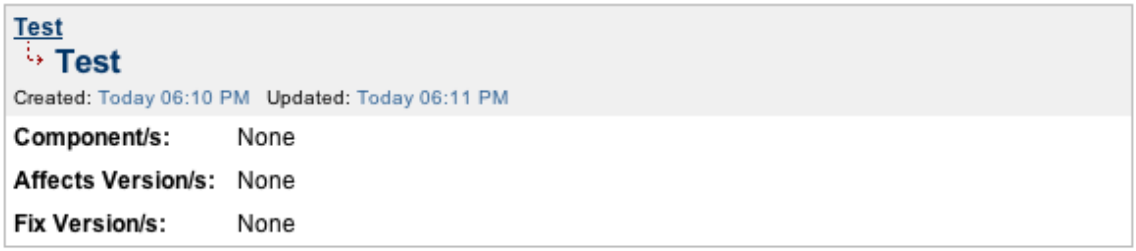
Wiki-Style Linking

Adding to the Wiki-style renderingfunctionality, users can now enrich their description, environment and comment entries with pertinent links to other JIRA data:

- User Aliasing
- Issue Aliasing

User Aliasing

Using the syntax *[~username]*, the username text will appear linked to the user profile of that user.



Issue Aliasing

Using the syntax *[some text here|TST-1]*, the text *some text here* will appear as a link to the specified issue. This text adopts the behaviour of a regular JIRA issue key link - if the issue is resolved, the text will appear with a strike-through.

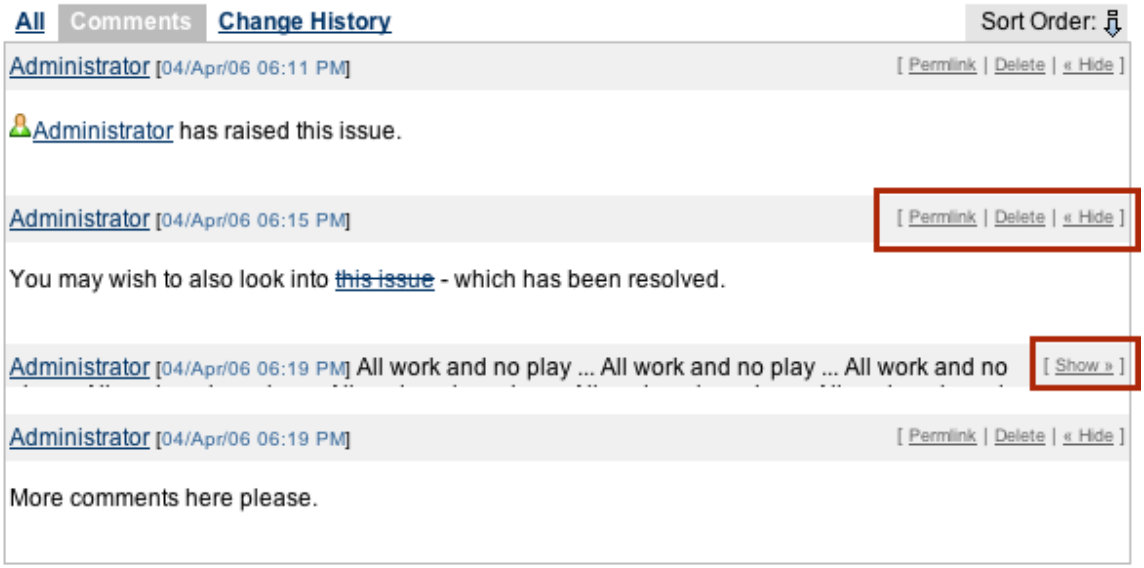
"I'm Feeling Lucky" Quick Search

Letting the user skip the results screen in the Issue Navigator of a search, the 'I'm Feeling Lucky' search presents the user with the first result associated with the search query. This seach can be engaged by pressing **Control + Enter** after submitting the search query to the 'Quick Search' query box.

Collapsible Fields

Enhancing the JIRA user interface, it is now possible to quickly configure the level of detail displayed while viewing an issue through collapsible and expandable issue fields.

The environment, description, individual comment fields and any textarea custom field can be *hidden* or *displayed* by simply clicking on the relevant link for a particular field - allowing fields with large amounts of data to be temporarily hidden while viewing other entries against that issue. JIRA only adds these options to fields that include a large amount of information.

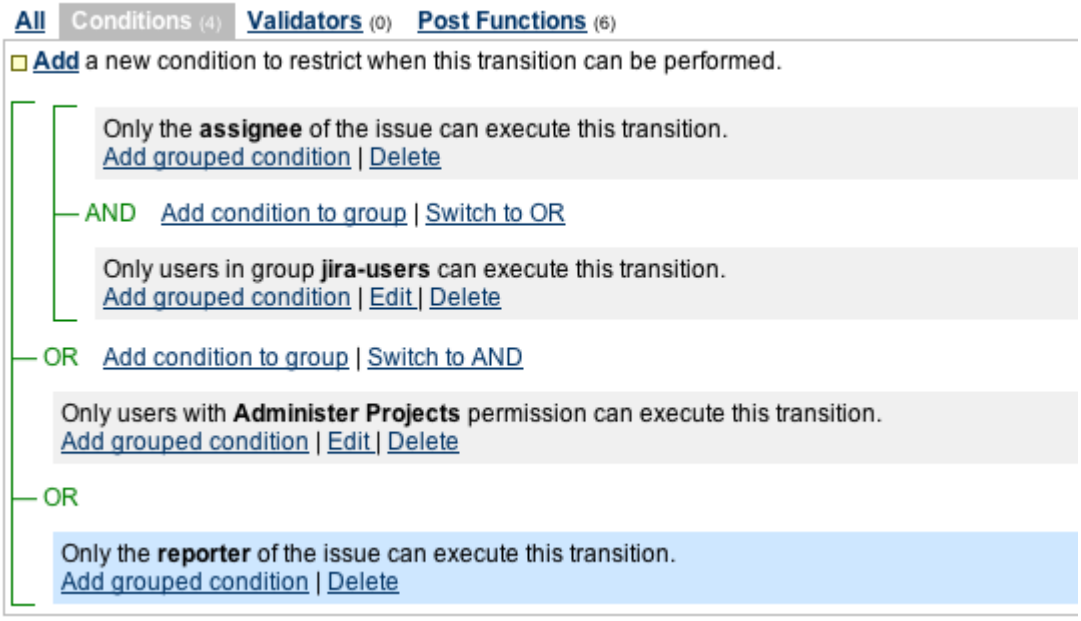


Nestable Conditions

Workflow conditions allow workflow designers the ability to restrict the availability of a workflow transition. The criteria of the condition must be met in order for the workflow transition to become available.

Workflow conditions can now be configured to combine criteria with boolean **OR** statements - allowing condition specification to match the workflow design more closely. For example, a condition can be constructed whereby it is only met if the user is the assignee of the issue **OR** the user is a member of the **jira-users** group.

Extending this concept, more complex condition criteria can be constructed with the ability to *nest* conditions in groups. Each group can be combined with other individual conditions or groups with the boolean **AND** or **OR** statements.



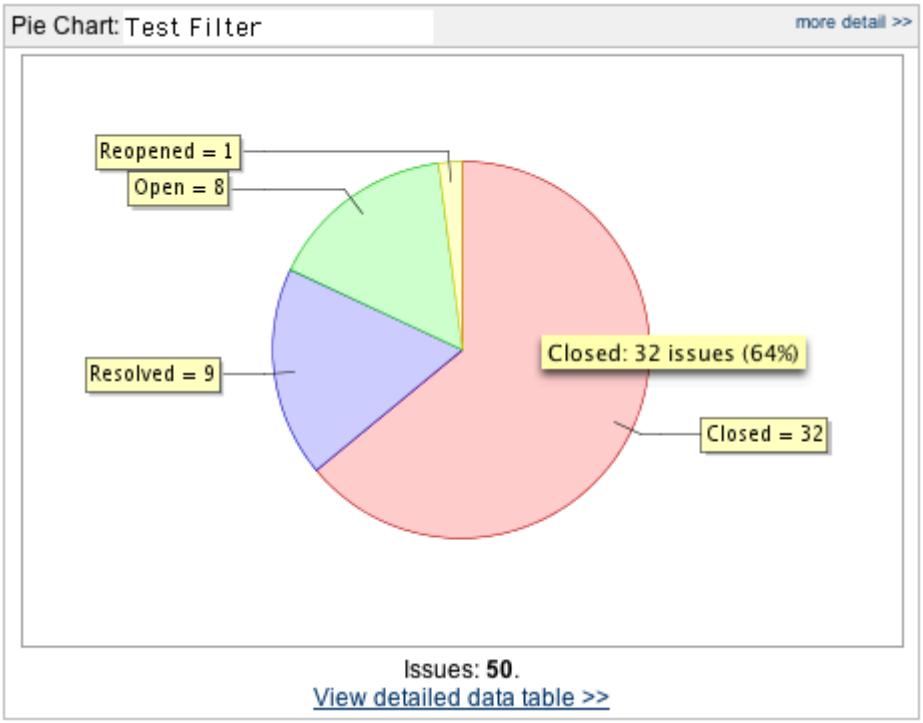
Charting Plugin Updates

The latest version of the Charting Plugin includes two new charting options:

- Pie Chart
- Average-Age Open Issues

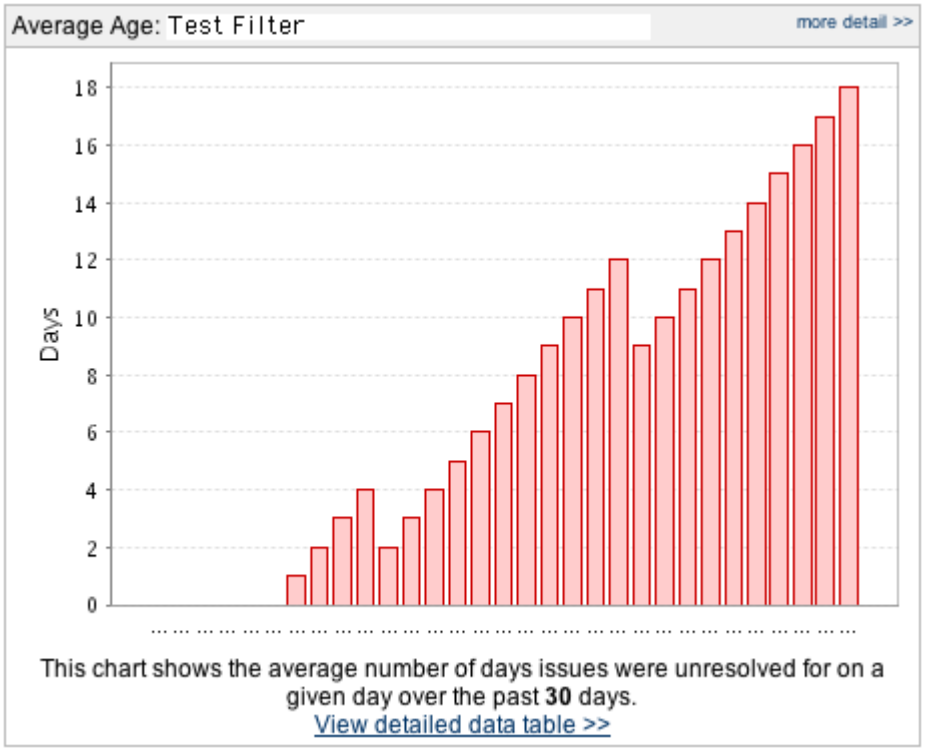
Pie Chart

Reports and portlets can be generated displaying data based on a statistic type (*e.g. Status, Priority, etc.*) of issues from a project or specified filter in pie-chart format.



Average-Age Open Issues

This chart displays the average-open-age of issues over a specified period with a configurable interval. This chart graphically conveys the trend for the average amount of time that issues remain unresolved.



Improvements

- Email Notification & Internationalization
- Performance
- Clone Issue Extensions
- Disable Notification for Bulk Operation
- Banner Visibility
- *... and many more ...*

Email Notification & Internationalisation

JIRA is shipped with over 15 language bundles - allowing the user to configure the language JIRA is displayed in globally and on a per-user basis. The internationalisation coverage now includes the email templates used in issue event notifications. Each template has been converted to allow full translations of the body of the email for all locales.

Taking JIRA into a truly global team environment, JIRA can now send individual email notifications to each recipient in *their user-profile selected language*. Hence, global team members located in regional sites around the world can be updated of issue events and updates with individually-tailored email notifications in their desired language.

This functionality becomes complete with the addition of properties files for the locales in use.

The translation process is greatly supported through the *much appreciated* efforts of the JIRA community. Many thanks to Gerd Gueldenast who has already provided a German translation for the email template properties - which is included in this release.

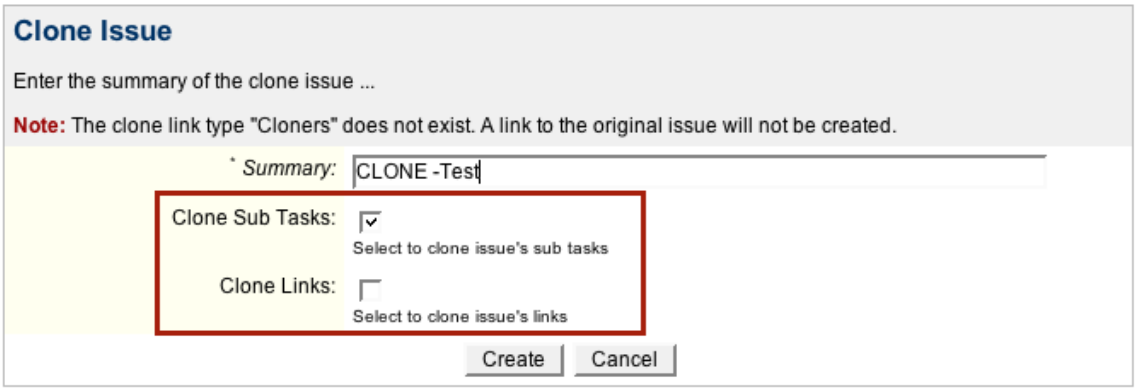
Performance

Improving overall response times while navigating through JIRA, this release introduces a caching servlet for all JavaScript and CSS resources. JIRA will cache these resources on the client system until the instance of JIRA is restarted. In-house testing and initial reports indicate that this change results in a significant performance improvement throughout JIRA.

Clone Issue Extensions

The **Clone Issue** functionality has been expanded with further configuration options. It is now possible to specify the level of detail cloned through:

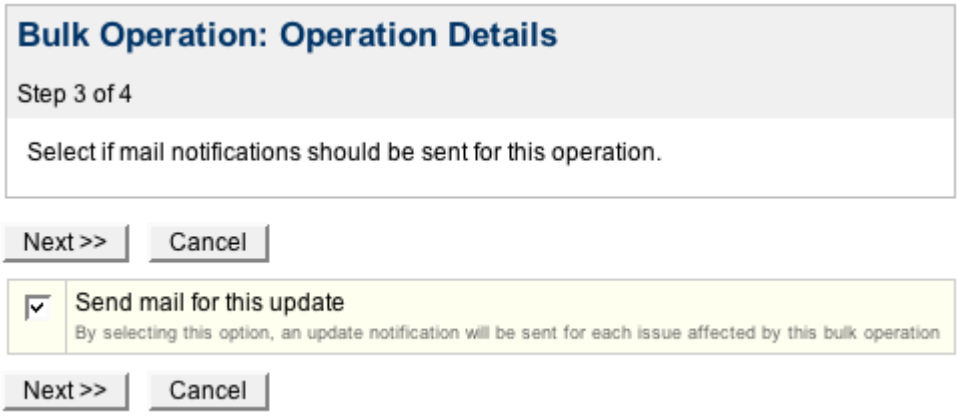
- Clone issue links - links between the clone issue and those linked to/from the original cloned issue will be recreated in the clone issue
- Clone sub-tasks - sub-tasks associated with the original cloned issue will be re-created for the clone issue



Disable Notification for Bulk Operation

JIRA administrators (and project administrators of the selected issues) can now configure whether notifications are sent for a bulk operation - avoiding mass emailing of all changes made during a bulk operation.

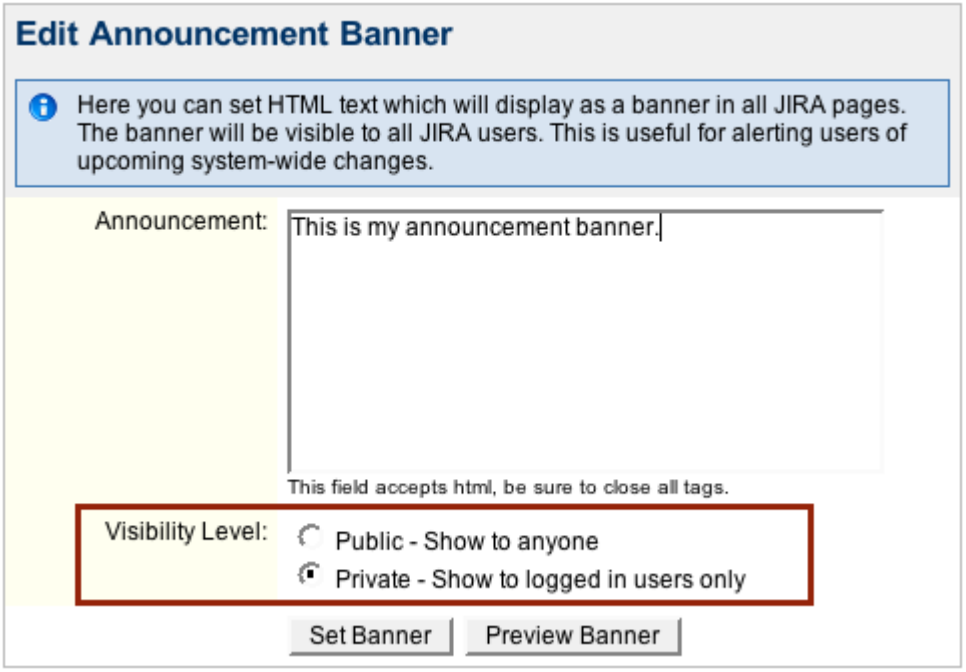
With this configurability, it is possible to complete updates on a collection of issues without generating unwanted email notification noise. The possible scenarios where this option can be applied are many - e.g. adding a new custom field and setting a default value, updating old issues ...



Banner Visibility

The announcement banner allows pertinent information to be displayed on all JIRA pages. This feature has been extended to allow configuration of the banner visibility level:

- **Public** - show the banner to anyone
- **Private** - show the banner to logged-in users only



... and many more ...

A number of other notable improvements included in this release ...

Mail Server & Bulk Precedence

The JIRA mail service can be configured to ignore emails with a *Precedence: bulk* header. This improvement eliminates potential issue/comment generation loops through the mail service from auto-generated emails.

Statable VersionPicker Field

JIRA provides a vast array of portlets that can populate your dashboard with instant access to issue information. The **Version Picker** custom field has been extended to allow inclusion in the 2-Dimensional Filter Statistics portlet - a portlet displaying the results of a search in table format with configurable axes.

Quick Sub-Task Creation Form

JIRA can now be configured so that the quick sub-task creation form is present on the *View Issue* screen at all times - even if the issue does not currently have any sub-tasks.

Issue Link Configuration

The issue link view can be configured to include various issue fields (e.g. issuetype, issuekey, etc.) - providing further details of the linked issue without having to navigate to that issue. Further, it is also possible to specify the sort order for this table.

Bug Fixes

This release includes a number of bug fixes - just ask JIRA to view the entire list!

Upgrading

In order to complete a successful upgrade, please refer to our Upgrade Guides. If you are upgrading from JIRA 3.5.3 please refer to the JIIRA 3.6 Upgrade Guide.

If you are upgrading from a pre-3.5.3 release, please refer to the relevant JIRA 3.x Upgrade Guides.

JIRA 3.6 Upgrade Guide

JIRA 3.6 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.x from JIRA 3.5.x. If upgrading from an older version of JIRA, please go to the complete list of Upgrade Guides, and read the notes for each version you are skipping during the upgrade.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Database Intensive Upgrade Task

To introduce the Custom events to JIRA, it was necessary to upgrade a large data set within JIRA's database for 3.5.x and earlier releases. Depending on the size of your JIRA data the upgrade task (number 150) might get your DBMS to do a lot of work which might take some time. The exact amount of time also depends on the processing power of the machine running JIRA's database.

Please be patient with the upgrade task and do not restart JIRA while the upgrade is in progress. The upgrade task will report on its progress to JIRA's log file as it upgrades your data.

The following is the sample output that the upgrade task will produce. As you can see the upgrade task took roughly 5 and a half minutes to modify over 660,000 records in the database.

```
11:14:09 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Phone Support
Workflow v.6'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Support Workflow
v.3'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Phone Support
Workflow v.7'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Test'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Copy of Support
Workflow'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Support Workflow
v.4'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Support Workflow'.
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150]
************************************************************
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating 660453 records in the
'NotificationInstance' table.
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150] This might take a long time. Please do
NOT stop JIRA.
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150]
************************************************************
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_CREATED'.
11:15:12 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_UPDATED'.
11:15:51 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_ASSIGNED'.
11:16:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_RESOLVED'.
11:16:46 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_CLOSED'.
11:16:57 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_COMMENTED'.
11:18:57 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_REOPENED'.
11:19:17 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_DELETED'.
11:19:26 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_MOVED'.
11:19:31 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_WORKLOGGED'.
11:19:37 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_WORKSTARTED'.
11:19:41 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_WORKSTOPPED'.
11:19:43 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_GENERICEVENT'.
11:21:23 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Update of 'NotificationInstance'
records finished.
```

Workflow Post Functions

Applies to	users with custom workflow XMLs saved on disk - external to JIRA

JIRA stores its workflows in the database. During the upgrade, these workflows will be upgraded automatically. However, if you have stored your workflows on disk (outside the database), you will need to follow these instructions to upgrade the workflows manually.

Previously, workflow post functions referenced the event to fire through a string value of the event name. All post functions now reference the event through a numeric ID value. As mentioned, all workflows stored within JIRA will be automatically updated. However, all workflows saved to disk - external to JIRA - should be updated manually as follows. The actual workflow XML file should be updated as follows:

For each workflow post function that accepts the event ID as an argument:

1. The value of the **name** attribute of the **arg** tag has to be changed from **eventType** to **eventTypeId**
2. The body of the **arg** tag has to change according to the following table:

Event Name	Event Type Id
created	1
updated	2
assigned	3
resolved	4
closed	5
commented	6
reopened	7
deleted	8
moved	9
worklogged	10
workstarted	11
workstopped	12
genericevent	13

By default, the only post functions that accept event IDs are *FireIssueEventFunctions.* Therefore, unless you have implemented your own custom post function that also deals with events, you will only need to update the arg tags for the *FireIssueEventFunctions* everywhere in the workflows.

For example, *FireIssueEventFunction* for create issue workflow transition looked like:

```
<function type="class">
    <arg
name="class.name">com.atlassian.jira.workflow.function.event.FireIssueEventFunction</arg>
     <arg name="eventType">created</arg>
</function>
```

and needs to be changed to:

```
<function type="class">
    <arg
name="class.name">com.atlassian.jira.workflow.function.event.FireIssueEventFunction</arg>
     <arg name="eventTypeId">1</arg>
</function>
```

Custom Events

Applies to	users who have **modified JIRA source code or added custom code to define new notification events**. Also of interest to users wishing to define new notification templates

Releases before JIRA 3.6 did not allow users create custom events. If you have modified the JIRA source to add custom events - please

follow these instructions.

If you have previously defined a custom event within JIRA - it is necessary to add appropriate entries to the following files:

- `system-event-types.xml`- used to install and upgrade all event types within the system to the new 3.6 event type object.
- `email-template-id-mappings.xml` - maps the event id to an assocaited velocity template file.

The `system-event-types.xml` file requires name and description details of the previously added custom event. For example, if the custom event type "Issue Frozen" was added to the system - the following entry should be added to the XML file:

```
<eventtype id="10000">
   <name>Issue Frozen</name>
   <description>This is the 'Issue Frozen' event type.</description>
   <notificationName>ISSUE_FROZEN</notificationName>
   <eventName>issuefrozen</eventName>
</eventtype>
```

The elements provide the following information:

- id - the new id for the event type. **All custom event types should be added from ID 10000 and above**
- notificationName - the original name for the event as found in the Notification table
- eventName - the origianl name for the event as found in workflows

The `email-template-id-mappings.xml` file requires an entry mapping the new custom event to an associated velocity email template. This mapping is used when a notification is sent for this event. Following from the above example, the following entry would be made:

```
<templatemapping id="10000">
   <name>Issue Frozen</name>
   <template>issuefrozen.vm</template>
</templatemapping>
```

The id should match that of the event as specified in the `system-event-types.xml` The `template` entity should reference the Velocity template to be used in email notifications of this event. A HTML and text version should be provided in the appropriate directory (html or text) at:

`<JIRA>/src/etc/java/templates/email/`

All custom event types added to the file *system-event-types.xml* should be added with an ID of 10000 and above

Custom Listeners

Applies to	users who have **added custom listeners** to JIRA.

For all users who have **added custom written** listeners to JIRA, it might be necessary to update the listener to follow the new JIRA 3.6 API.

There are two things to look out for:

1. signature change of the *workflowEvent* method
2. change of return type of *getIssue()* method on the *IssueEvent* object

The signature of the method *workflowEvent* in the *IssueEventListener* has changed from:

```
public void workflowEvent(int type, IssueEvent event);
```

to:

```
public void workflowEvent(IssueEvent event);
```

Note: the type parameter has been removed.

If you have implemented *IssueEventListener* directly or have extended *AbstractIssueEventListener* and have overridden the method

workflowEvent, you will need to change and recompile your listener before installing JIRA 3.6.

In JIRA 3.6, the event type ID can be retrieved by calling the following method on the *IssueEvent* object:

```
Long eventID = event.getId();
```

However, the returned value of the *getId()* method is different to the values of the type parameter that was passed to the *workflowEvent* method. The following table represents these differences:

Event Name	Old ID	New ID
created	0	1
updated	1	2
assigned	2	3
resolved	3	4
closed	4	5
commented	5	6
reopened	6	7
deleted	7	8
moved	8	9
worklogged	9	10
workstarted	10	11
workstopped	11	12
genericevent	-1	13

Also, the *getIssue()* method of the *IssueEvent* object has changed to return an Issue object instead of a *GenericValue* object representing an issue.

Users who have created and added custom listeners must update the listener to now operate with the *Issue* object. For example:

```
Issue issueObject = event.getIssue();
```

As a quick fix, you can modify your listener to use `event.getIssue().getGenericValue()`.

The event type ID constants are now only available from the class *EventType*. Any use of the original constants must be updated to use the *EventType* constants. For listeners that reference an event ID by its numeric value - it is necessary to ensure that the IDs now match those as defined in *EventType*.

Custom permission types

Applies to	users who have modified JIRA source to add new permission types (ie. in addition to the standard 'user', 'group', 'assignee' types).

The SecurityType interface, used to implement permission types ('single user', 'group' etc) has had a `getUsers()` method added. If you have implemented your own SecurityType you will need to implement this. See the source of current implementations (eg. `GroupCF`) for tips.

Plugin upgrades required

As usual, you should check whether the plugins you use are compatible with the new release. Generally, plugins (like the Subversion plugin or JIRA toolkit) need to be upgraded when JIRA is upgraded. See the list of plugins at:

http://confluence.atlassian.com/display/JIRAEXT/Home

JIRA 3.6.5 Release Notes

JIRA 3.6.5 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.6.5 in Standard, Professional and Enterprise editions. This point release includes 8 bug fixes and improvements, in particular improvements in performance and efficiency. It can be downloaded here.

Not using 3.6? Learn about all the **new features you're missing out on**!

If upgrading from an earlier version please read through the JIRA 3.6.5 Upgrade Guide.

JIRA 3.6.5 includes 8 bug fixes and improvements.

JIRA Issues (10 issues)					
Type	**Key**	**Summary**	**Priority**	**Status**	**Resolution**
	JRA-11267	Stacktrace: "java.lang.ClassCastException: java.lang.String" when viewing custom fields		Resolved	Fixed
	JRA-11063	"Add a new Issue Security scheme" link in SelectProjectIssueSecurityScheme is empty		Resolved	Fixed
	JRA-11036	Fix MailHandler attachment code - CPU inefficient		Resolved	Fixed
	JRA-11029	Unable to edit custom fields in workflow transition view in "Bulk change" mode		Resolved	Fixed
	JRA-11014	UnSynced map/cache in AbstractSchemeManager.cacheProjectSchemes		Resolved	Fixed
	JRA-10990	"Edit Configuration" on the "Default Configuration Scheme" for a "Custom Field" I created causes a crash (upgrading from 3.6.3 to 3.6.4) for "All Issue Types"		Resolved	Fixed
	JRA-10951	Mystery stacktraces: IllegalArgumentException in CustomFieldTypeModuleDescriptor.getEditHtml		Resolved	Fixed
	JRA-10907	Caching the user's projects improves the speed of Issue Navigator		Resolved	Fixed
	JRA-10223	Ensure transaction thread local gets cleared for every request and service run (potential deadlock fix)		Resolved	Fixed
	JRA-6744	Issue details pane does not stretch properly when reporter name is a long single word		Resolved	Fixed

JIRA 3.6.5 Upgrade guide

JIRA 3.6.5 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.5 from JIRA 3.6.4. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading to JIRA 3.6.5 from JIRA 3.6.4.

JIRA 3.6.4 Release Notes

JIRA 3.6.4 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.6.4 in Standard, Professional and Enterprise editions. This point release includes over 20 bug fixes and improvements, in particular improvements in performance, efficiency and security. It can be downloaded here.

Not using 3.6? Learn about all the **new features you're missing out on**!

If upgrading from an earlier version please read through the JIRA 3.6.4 Upgrade Guide.

JIRA 3.6.4 includes over 20 bug fixes and improvements.

JIRA Issues (25 issues)

Type	Key	Summary	Priority	Status	Resolution
	JRA-10601	XML-RPC to fix/improve String casting		Resolved	Fixed
	JRA-3363	"Default Locale" is not listed under "Installed Locales"		Resolved	Fixed
	JRA-10833	Ensure the OfBiz Iterator is closed before the transaction is commited (i.e. connection closed)		Resolved	Fixed
	JRA-10700	Link types drop-down on Create Link screen shows "jira_subtask" links		Resolved	Fixed
	JRA-10199	Need to make move issue in Professional edition show the issue type since we allow issue type schemes in Pro		Resolved	Fixed
	JRA-10762	Connection is Closed errors if the ThreadLocal connection gets closed before a commit or rollback.		Resolved	Fixed
	JRA-10637	No Manage Filters link in saved filter popup		Resolved	Fixed
	JRA-7831	Changing Workflow should not change Date Updated or provide option not to		Resolved	Fixed
	JRA-9418	setArchived and setReleased in RemoteVersion does nothing		Resolved	Fixed
	JRA-10785	Integrity Checker leaks a database connection if an exception occurs during a check		Resolved	Fixed
	JRA-3981	Change error message when user does not have global USE permission		Resolved	Fixed
	JRA-10662	Progress Status bar is broken in IE & Opera		Resolved	Fixed
	JRA-10595	SOAPUtils's transformToStringArray() always returns an empty String array		Resolved	Fixed
	JRA-10308	Giving Manage Watchers Permission to Reporter or Current Assignee allows anyone who can view the watchers list to edit it		Resolved	Fixed
	JRA-10535	Hide/show of Free Text Field Custom Field does not work properly in Issue Navigator		Resolved	Fixed
	JRA-10695	NullPointerException when sharing a filter		Resolved	Fixed
	JRA-5038	Porlets - cannot specify Portlet name without i18n string and associated property.		Resolved	Fixed
	JRA-11080	Version custom fields not available for use in statistics portlets (not statable)		Resolved	Fixed
	JRA-10421	NPE rendering work logged email with '0m'		Resolved	Fixed
	JRA-8405	Priority levels popup prompts for login		Resolved	Fixed
	JRA-10494	Bugzilla import looses descriptions of components		Resolved	Fixed
	JRA-7213	SOAP Interface - getAttachmentNames(...) always returns null		Resolved	Fixed
	JRA-10336	After changing the icon on a custom Issue Type, related custom fields disappeared from the different screens		Resolved	Fixed

	JRA-10468	Add comment of using JVM version 1.5_06-b05 in "Installing Java" section of Documentation.		Closed	Fixed
	JRA-10789	typo in admin projects page		Resolved	Fixed

JIRA 3.6.4 Upgrade guide

JIRA 3.6.4 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.4 from JIRA 3.6.3. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading to JIRA 3.6.4 from JIRA 3.6.3.

JIRA 3.6.3 Release Notes

JIRA 3.6.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.6.3 in Standard, Professional and Enterprise editions. This point release includes over 30 bug fixes and improvements, in particular improvements in performance, efficiency and security. It can be downloaded here.

If upgrading from an earlier version please read through the JIRA 3.6.3 Upgrade Guide.

JIRA 3.6.3 includes over 30 bug fixes and improvements.

JIRA Issues (36 issues)					
Type	**Key**	**Summary**	**Priority**	**Status**	**Resolution**
	JRA-10526	Support Date custom fields in Mantis 1.0.3+		Resolved	Fixed
	JRA-3349	More Information on files for the Version Control Tab		Closed	Fixed
	JRA-10609	Ability to sort 2DStats portlet by totals and limit results		Resolved	Fixed
	JRA-9090	Wiki Renderer does not HTML encode contents of unknown macros		Resolved	Fixed
	JRA-10625	Saved Filter is broken in 3.6.3 #159		Resolved	Fixed
	JRA-10105	jsessionid twice in URL causes Fix and Affects version links to break in issue view screen during session's first page view		Resolved	Fixed
	JRA-9296	Indexing intermittently fails with IndexException		Resolved	Fixed
	JRA-10276	Custom field is displayed during Bulk Edit and Bulk Transition when it does not have a context for some of the issues.		Resolved	Fixed
	JRA-10504	The CombinedCachingServlet can become a bottleneck under heavy load		Resolved	Fixed
	JRA-10542	Request parameters are not HTML encoded on the 500 page		Resolved	Fixed
	JRA-10317	Bundle JIRA Calendar plugin correctly.		Resolved	Fixed
	JRA-10590	Fix PERF-95 - address Perforce depots with 'gaps' in the commit numbers		Resolved	Fixed
	JRA-10237	Customfields belonging to specific projects are shown when those projects are selected		Resolved	Fixed

	JRA-10518	FieldScreenImpl.getTab(...) throws IndexOutOfBoundsException		Resolved	Fixed
	JRA-10253	Indexing the subtasks is inefficient		Resolved	Fixed
	JRA-10488	FilterStatisticsValuesGenerator retains references to managers through static final references		Resolved	Fixed
	JRA-10567	Investigate caching 'EditActions' on Bulk Workflow Transition 'edit fields' screen		Resolved	Fixed
	JRA-10552	Cancel button in Comment does not have tooltip		Resolved	Fixed
	JRA-10178	Message custom field produces change history entries		Resolved	Fixed
	JRA-9336	NPE on viewing saved filters if the user is not logged in		Resolved	Fixed
	JRA-10304	Fix handling of null users present in database		Resolved	Fixed
	JRA-9933	Digital signature as an attachment		Resolved	Fixed
	JRA-10487	Upgrade to latest Atlassian Renderer		Resolved	Fixed
	JRA-8794	Unable to CreateIssue or Comment as another user via SOAP (eg. set reporter)		Resolved	Fixed
	JRA-10341	Adding New Version does not show up in Open Issues or via SOAP until Restart		Resolved	Fixed
	JRA-10263	Description is not shown in full content view of Issue Navigator		Resolved	Fixed
	JRA-10394	NullPointerException when searching on Cascading Select		Resolved	Fixed
	JRA-10505	Second dropdown list of Cascading Select custom field does not get updated during Bulk Workflow Transition		Resolved	Fixed
	JRA-10061	Access to schemeItems in FieldScreenSchemeImpl needs to be synchronized		Resolved	Fixed
	JRA-10434	ClassCastException in com/atlassian/jira/issue/comparator/IssueKeyComparator		Resolved	Fixed
	JRA-10392	Full Content (HTML) View doesn't show description		Resolved	Fixed
	JRA-10272	"Restore Data from XML" doesn't seem to flush SOAP cache		Resolved	Fixed
	JRA-10289	Permlink not translated in e-mails		Resolved	Fixed
	JRA-10335	Status names are not translated in the tooltip on the Browse Project page		Resolved	Fixed
	JRA-10499	In Firefox, the Home page bar-graph links to Issue Navigator fail		Resolved	Fixed
	JRA-10450	View issue details (Word export) should contain issue status		Resolved	Fixed

JIRA 3.6.3 Upgrade guide

JIRA 3.6.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.3 from JIRA 3.6.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.6.3 from JIRA 3.6.2.

JIRA 3.6.2 Release Notes

JIRA 3.6.2 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.6.2 in Standard, Professional and Enterprise editions. This point release includes over 40 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version please read through the JIRA 3.6.2 Upgrade Guide.

JIRA 3.6.2 includes over 40 bug fixes and improvements.

JIRA Issues (47 issues)					
Type	**Key**	**Summary**	**Priority**	**Status**	**Resolution**
	JRA-10167	Bulk Edit allows you to edit issue type even if the workflows are different		Resolved	Fixed
	JRA-10086	create upgrade task to clean out erroneous statuses in workflow transitions with SubTaskBlockingCondition		Resolved	Fixed
	JRA-10109	Second select box of cascading select custom field not populated correctly during Bulk Edit		Resolved	Fixed
	JRA-10065	Cascading select custom field doesn't pop up in the "Move Issue" screen if the field has a context in source and destination projects.		Resolved	Fixed
	JRA-9864	OutOfMemoryErrors when running Confluence and Jira on the same server		Resolved	Fixed
	JRA-9791	Error unarchiving Jira on Mac OS X		Resolved	Fixed
	JRA-10049	Write functional tests for subversion plugin and automate them in the nightly build		Resolved	Fixed
	JRA-10213	Version custom fields will munge their data when going through a move or bulk move		Resolved	Fixed
	JRA-9883	Deleting a User doesn't remove them from the Component Lead		Resolved	Fixed
	JRA-10101	Review 3.6.1 Bugs		Resolved	Fixed
	JRA-9946	The projects won't appear in browse project view after Bugzilla import		Resolved	Fixed
	JRA-3969	Services should be disabled during reindexes		Resolved	Fixed
	JRA-9812	Project cache refreshed incorrectly after creating project by remote API		Resolved	Fixed
	JRA-10007	Docs: Add documentation on DB connection pool size		Closed	Fixed
	JRA-9377	Unable to set default value in Multi User Picker field		Resolved	Fixed
	JRA-9944	Trackbacks - send outgoing pings to all issues is not reflected in trackback view screen.		Resolved	Fixed
	JRA-9711	Excel export layout became worse with the new JIRA version		Resolved	Fixed
	JRA-9928	Status names with apostrophes generate exceptions on *Browse Projects* page		Resolved	Fixed
	JRA-10116	JavaScript error on everypage with IE 7		Resolved	Fixed

	JRA-10055	Attachment creation failure causes message handler to loop over message indefinitely.		Resolved	Fixed
	JRA-10006	Tomcat Config: Increase *maxActive* database pool connection limit		Resolved	Fixed
	JRA-10076	Screenshot attachment applet cause closed session on Websphere 6.0		Resolved	Fixed
	JRA-9724	Session Timeout/Logout with tabbed dashboard broken		Resolved	Fixed
	JRA-10204	Remove I'm feeling luck from quick search as no-one uses it, and it causes problems on IE 7		Resolved	Fixed
	JRA-10215	CustomFieldManager: getCustomFieldObjects passes issue ID rather than project ID		Resolved	Fixed
	JRA-10148	Subversion Plugin: Repository URLs with no realtive repository path not working		Resolved	Fixed
	JRA-7952	Unable to locate documentation reference		Closed	Fixed
	JRA-10084	Update the Issue Security Scheme Documentation		Closed	Fixed
	JRA-9934	Sub-Task blocking condition is not displayed correctly in Transition Condition screen after it has been added.		Resolved	Fixed
	JRA-10083	Exporting an issue to word displays descrption twice		Resolved	Fixed
	JRA-10212	TestUpgradeTask_Build56 should fetch the e-mail address from external-link.properties rather than hard code the value		Resolved	Fixed
	JRA-9961	Session Timeout caused in Bulk Operations		Resolved	Fixed
	JRA-9324	Groups in group browser encoded with system encoding		Resolved	Fixed
	JRA-10052	Create a script that calculates the number of classes that use GenericValues		Resolved	Fixed
	JRA-10113	Upgrade to latest atlassian-extras		Resolved	Fixed
	JRA-10133	move patched jar to build server and distribute to ImaHima		Resolved	Fixed
	JRA-10087	Add doc that *no count* setting should not be checked for MS SQL Server		Resolved	Fixed
	JRA-10176	Update docs to mention that the drop down list restricting comment and work log view only contains user's groups		Closed	Fixed
	JRA-10030	Adjust layout formatting for HTML left column on Issue Details page		Resolved	Fixed
	JRA-10111	Remove JiraLuceneFieldCache cache		Resolved	Fixed
	JRA-9938	JIRA Docs - Trackbacks - Technical Info section displayed twice.		Closed	Fixed
	JRA-10103	Type in Danish Email Translation		Resolved	Fixed
	JRA-10112	Update Screenshot of the Setup Wizard part in the docs		Resolved	Fixed
	JRA-10082	Translation error in notifications for German		Resolved	Fixed
	JRA-9719	Update notifications doco		Closed	Fixed

	JRA-10142	Typo in an error message		Resolved	Fixed
	JRA-10144	Need to synchonize a hashmap in the ofbiz memoryHelper that we use for unit tests		Resolved	Fixed

JIRA 3.6.2 Upgrade guide

JIRA 3.6.2 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.2 from JIRA 3.6.1. If upgrading from an older version of JIRA, please go to the complete list of Upgrade Guides, and read the notes for each version you are skipping during the upgrade.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Maximum Active Databased Connections

Applies to	JIRA Standalone users

In version of JIRA before 3.6.2, the maximum number of database connections was limited to 8 by default. If JIRA was used by more than 8 concurrent users or under very haeavy usages, the users could experience delays or JIRA could hang.

In JIRA 3.6.2 the default number of maximum active database connections has been increased to 20. When upgrading to JIRA 3.6.2, please ensure that your database will allow JIRA to establish 20 connections, or decrease this number to desired value. To adjust the number of connections change the value of the `maxActive` attribute of the `jdbc/JiraDS` resource in `config/server.xml` file. JIRA has to be restarted to apply the change.

JIRA 3.6.1 Release Notes

JIRA 3.6.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.6.1 in Standard, Professional and Enterprise editions. This point release includes over 30 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version please read through the JIRA 3.6.1 Upgrade Guide.

JIRA 3.6.1 includes over 30 bug fixes and improvements.

JIRA Issues (32 issues)					
Type	**Key**	**Summary**	**Priority**	**Status**	**Resolution**
	JRA-9974	Please enable localization of Constants Help pop-up page		Resolved	Fixed
	JRA-8954	Ensure JIRA unit (and functional tests?) pass on machines with locate set to other than en_* e.g. en_US or en_AU		Resolved	Fixed
	JRA-9905	Contain HSQL warning in the Admin portal		Resolved	Fixed
	JRA-9820	Danish Property Filtes for 3.6		Resolved	Fixed
	JRA-9932	Bug in Find issues - sub tasks field causes error		Resolved	Fixed
	JRA-9967	Invalid workflow action with 'Assignee only' condition and AssignTo field on transition screen		Resolved	Fixed
	JRA-9903	JIRA doesn't build from source		Resolved	Fixed
	JRA-9971	Update workflow PostFunction does not show all users		Resolved	Fixed
	JRA-9806	NullPointerException when commenting on an issue through edit screen		Resolved	Fixed

	JRA-9550	When editing workflow transition with no transition view, transition view is set (seemingly) randomly in editor screen.		Resolved	Fixed
	JRA-10022	admin.jsp throws Exception in SunOne Application Server 7		Resolved	Fixed
	JRA-9993	NullPointerException when component has no lead		Resolved	Fixed
	JRA-12899	FixVersions parameter is being ignored in the CreateIssue tag		Resolved	Fixed
	JRA-9811	jira:CreateIssue via Jelly does not allow specifying multiple components		Resolved	Fixed
	JRA-9889	Version Picker Custom Field can't be CSV imported with value or custom field ID		Resolved	Fixed
	JRA-9975	update firebird entitymodel		Resolved	Fixed
	JRA-10027	Synchronize the getTemplateFilename method from TemplateManager		Resolved	Fixed
	JRA-9894	IndexOutOfBoundsException in Average Age Report and Pie Chart Report		Resolved	Fixed
	JRA-9922	ConcurrentModificationException if you move two portlets on the dasboard at the same time		Resolved	Fixed
	JRA-9942	Issue Links in column view not escaped		Resolved	Fixed
	JRA-9973	Update Traditional Chinese Pack for 3.6.x		Resolved	Fixed
	JRA-10037	Set default email format to text		Resolved	Fixed
	JRA-9962	Create link to Services page on Email Handler doc		Resolved	Fixed
	JRA-9813	Add log location to 500 page		Resolved	Fixed
	JRA-9887	Merge this fix into the 3.6 branch		Resolved	Fixed
	JRA-9851	Image thumbnails displayed for (lowercase) png images but not displayed for (uppercase) PNG images		Resolved	Fixed
	JRA-4945	Clicking the Log In link creates ever longer URLs		Resolved	Fixed
	JRA-9969	Minor email-translation issue		Resolved	Fixed
	JRA-9957	Need to make a clean way of deploying the jira dev kit		Resolved	Fixed
	JRA-9839	Misspelling in SimpleWorkflowManager error - "woth" should be "with"		Closed	Fixed
	JRA-9854	Cannot find a key for a text to translate		Resolved	Fixed
	JRA-9991	JIRA help link for "version management" is incorrect		Resolved	Fixed

JIRA 3.6.1 Upgrade Guide

JIRA 3.6.1 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.1 from JIRA 3.6. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is

available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.6.1 from JIRA 3.6.

JIRA 3.5 Release Notes

JIRA 3.5 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems, recipient of the Deloitte Fast 500 Award, is proud to announce the latest release of *the* issue tracking and project management application - JIRA 3.5 (download it here).

The latest release includes over 50 powerful new features and improvements along with over 50 bug fixes.

Upgrade Information

In order to complete a successful upgrade, please refer to our Upgrade Guides. If you are upgrading from JIRA 3.4.3 please refer to the JIRA 3.5 Upgrade Guide.

If you are upgrading from a pre-3.4.3 release, please refer to all JIRA 3.x Upgrade Guides.

Contents

- New Features
- Improvements
- Bug Fixes

New Features

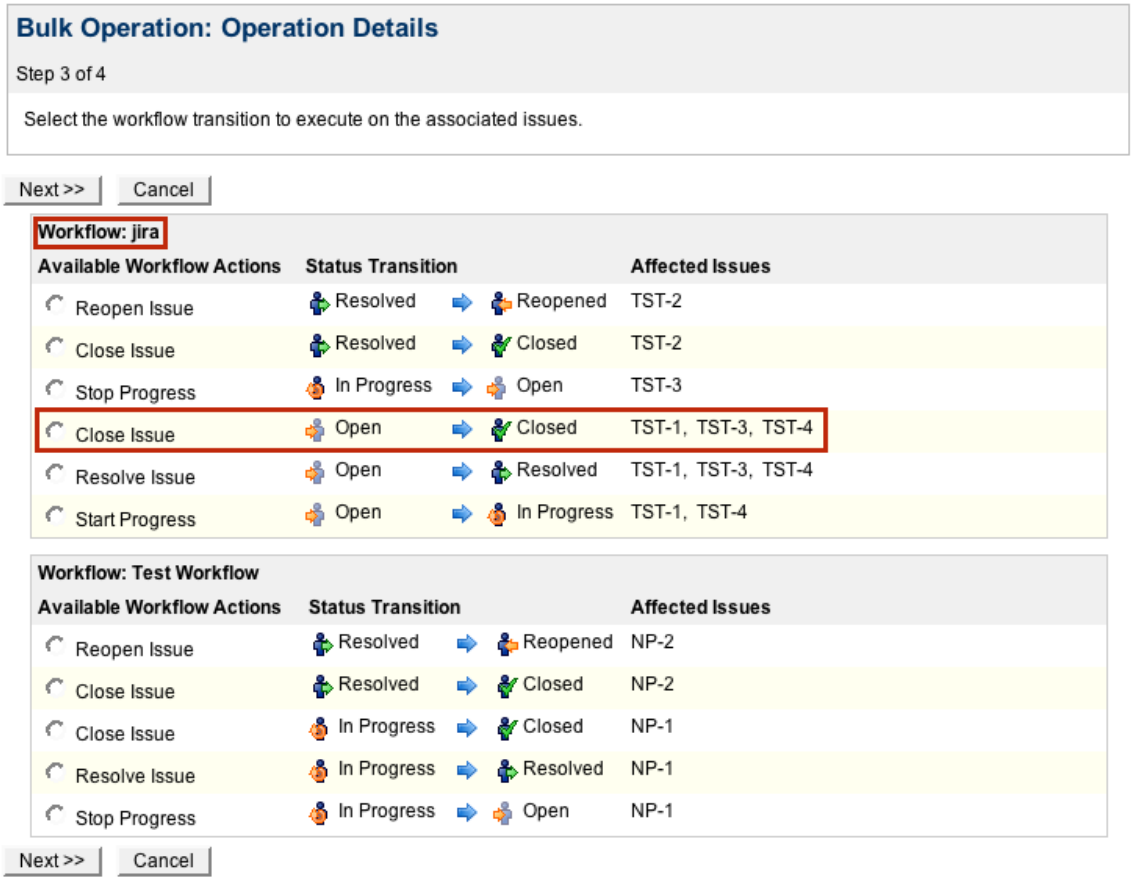
JIRA 3.5 includes an impressive array of new features and improvements - some of which are noted below. To view the entire new feature and improvement list - ask JIRA!

- Bulk Workflow Transition
- FogBugz Importer
- Charting Plugin
- MS Word Export
- JIRA Page Linker Plugin
- Component Lead Notification Type

Bulk Workflow Transition

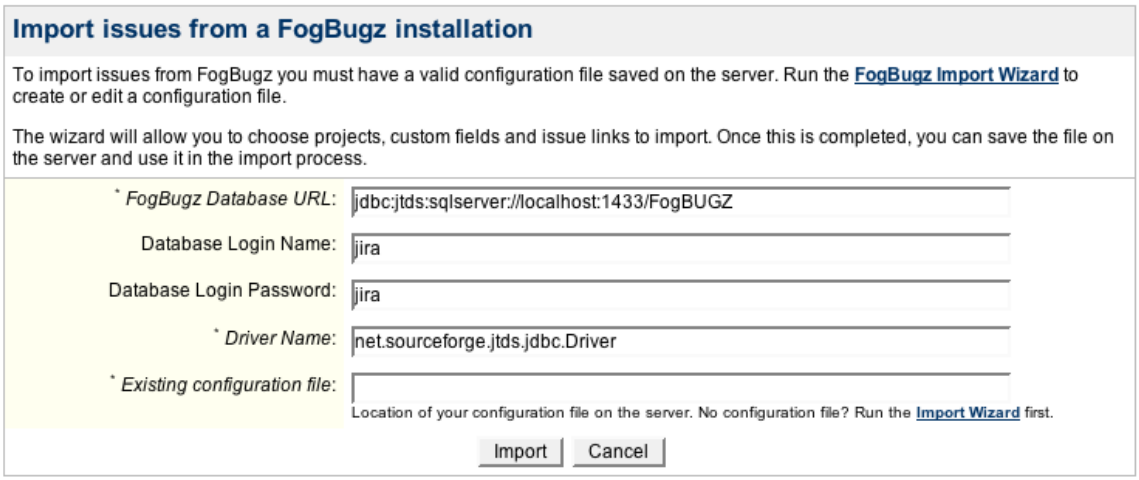
JIRA 3.5 extends the bulk operation capabilities with the addition of **Bulk Workflow Transition** - allowing a selected collection of issues to be advanced through the associated workflow.

The **Bulk Workflow Transition** process is as simple as if transitioning one issue. Once the collection of issues is retrieved through the issue navigator, it is possible to progress the issues through the associated workflow. The associated screen for the transition is displayed to the user - allowing all related fields to be edited as normal. All edits (including comments) are applied to each issue and each issue is advanced to the associated workflow status. All field configurations, workflow conditions, validators and post functions are respected throughout the process.



FogBugz Importer

Users of the FogBugz issue tracking system can now easily import their FogBugz data into JIRA through the new configurable import wizard. The import wizard allows the user specify which projects, custom fields and issue links are to be imported.

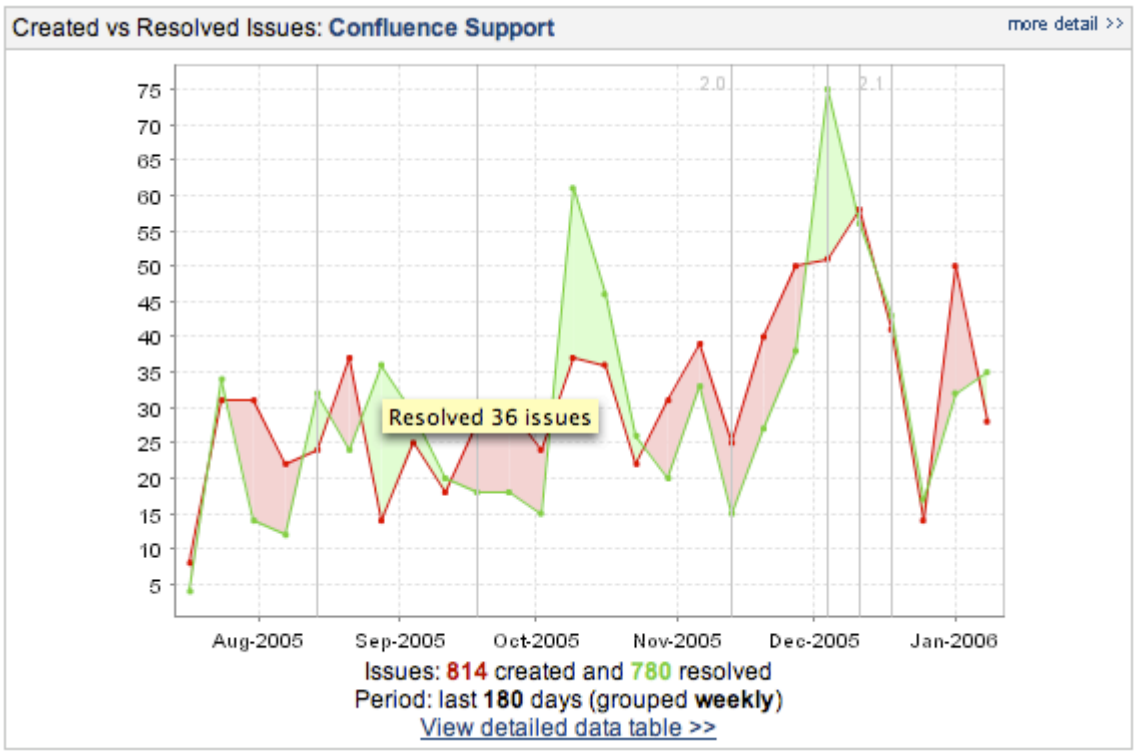


Charting Plugin

The ability to visually represent JIRA data in a graphical form is a key requirement for many JIRA users. With the **Charting Plugin**, users can translate their data into graphs and charts.

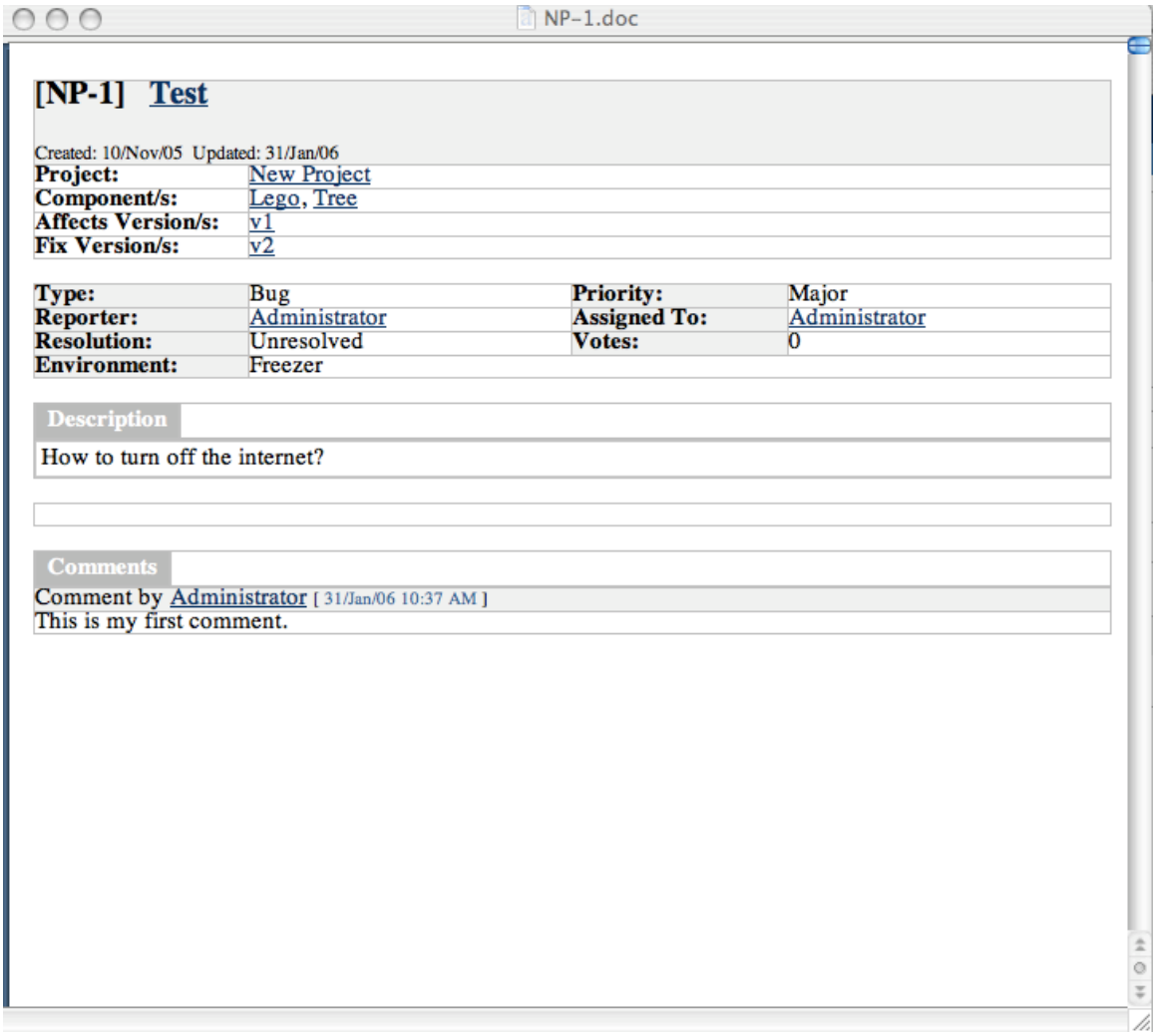
Allowing for more meaningful presentations with greater impact, graphs and charts are more easily absorbed than plain numerical data. Over time, users can also plot trend graphs - with the prospect of extrapolating and interpreting the graphical data to predict future trends and workloads.

The plugin can be downloaded here.



MS Word Export

It is now easier to create Word document presentations, with the ability to export JIRA data to Microsoft Office Word format. It is possible to export an individual issue or the 'Full Content View' of any search to a Word document.


NP-1.doc

[NP-1] Test

Created: 10/Nov/05 Updated: 31/Jan/06

Project:	New Project		
Component/s:	Lego, Tree		
Affects Version/s:	v1		
Fix Version/s:	v2		
Type:	Bug	**Priority:**	Major
Reporter:	Administrator	**Assigned To:**	Administrator
Resolution:	Unresolved	**Votes:**	0
Environment:	Freezer		

Description

How to turn off the internet?

Comments

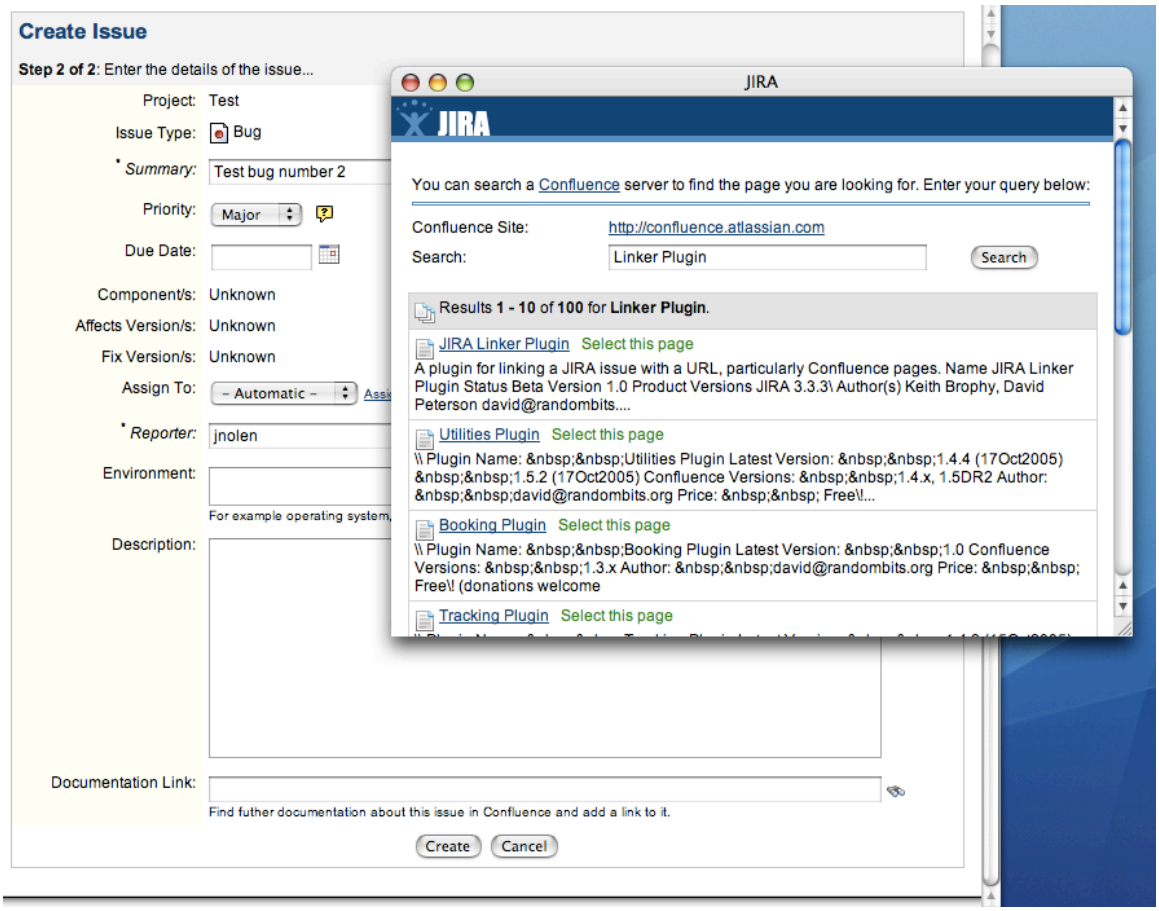
Comment by Administrator [31/Jan/06 10:37 AM]
This is my first comment.

JIRA Page Linker Plugin

Providing further integration between JIRA and Confluence, the **JIRA Page Linker Plugin** provides a custom field plugin for linking a JIRA issue with a Confluence URL.

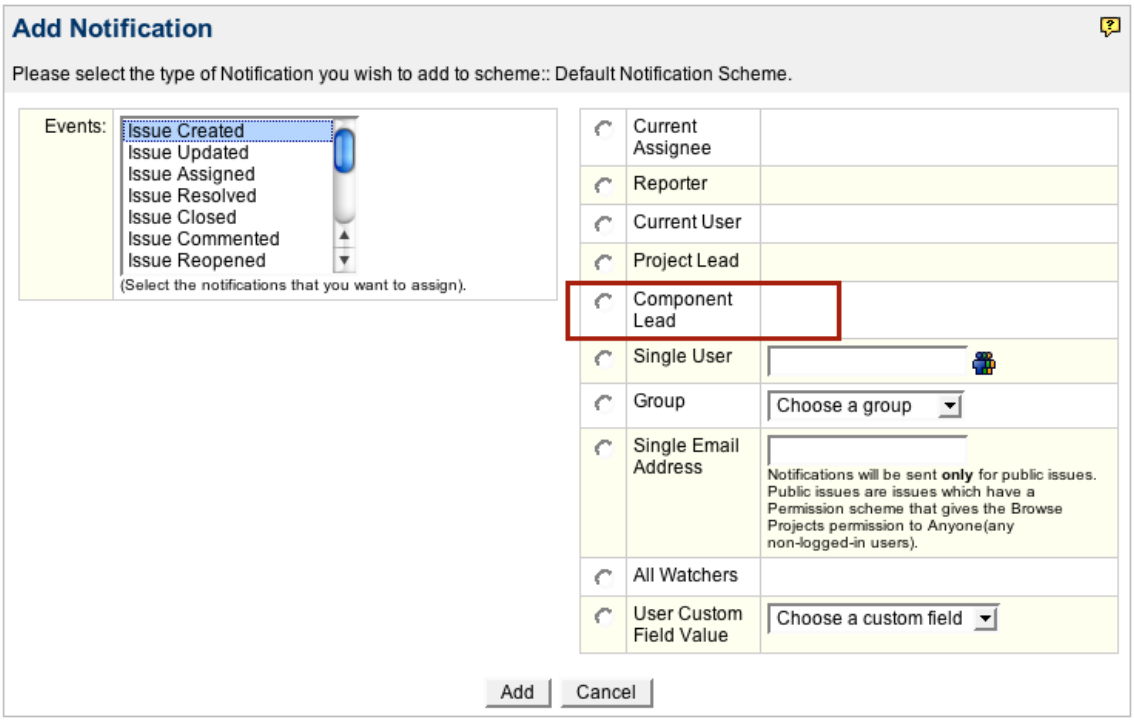
While you are creating or editing a JIRA issue, you can bring up a popup window that will allow you to quickly search a Confluence site for pages you would like to link to this issue. Clicking a link in the popup window will add the page to your new issue.

The JIRA Page Linker plugin can be downloaded here.



Component Lead Notification Type

JIRA 3.5 Enterprise introduces the notification type of **Component Lead**. With this option, notification schemes can be simplified through the specification of the **Component Lead** as a recipient of issue update emails. For each event specified in the scheme, the **Component Lead** will receive an email update - ensuring that the correct users are alerted.

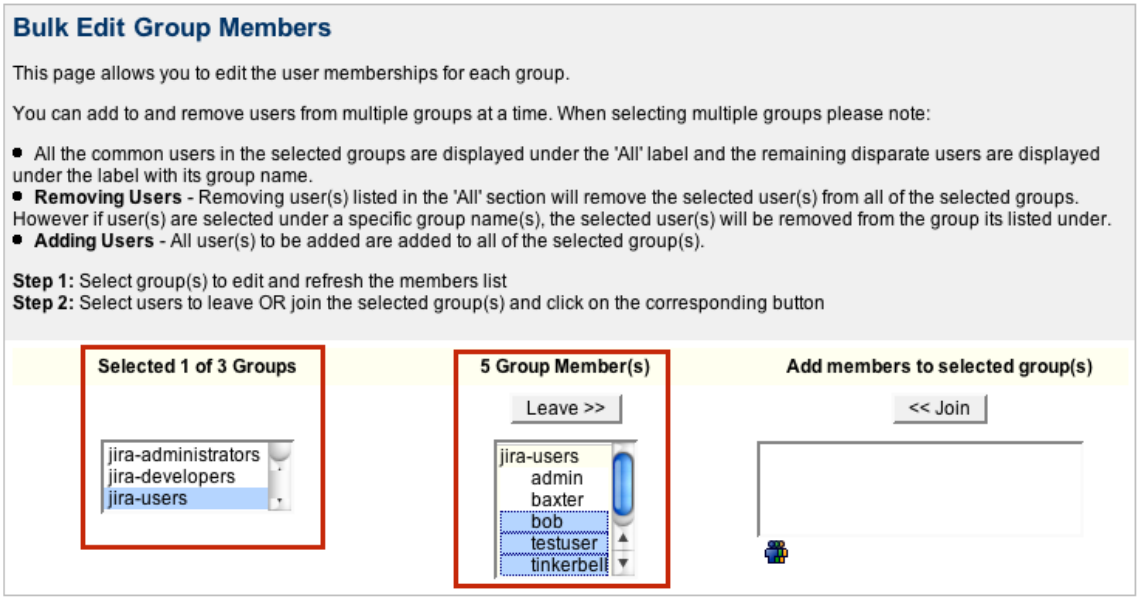


Improvements

- Bulk Assignment of Users to Groups
- Jelly Improvements
- SOAP Improvements
- Configure CSV Delimiter
- Mantis Custom Field Import
- Plugin System Improvements
- Configure Email Address Format
- Internationalisation
- Multi-Select Version Picker Custom Fields in Filters
- Delete Trackbacks
- Configured Navigator Columns in Email Filter Subscription
- Convert Bugzilla Link to JIRA Link

Bulk Assignment of Users to Groups

Group management has been greatly simplified with the addition of **Bulk Assignment of Users to Groups** - allowing multiple users to be added or removed to a group at once.



Jelly Improvements

The Jelly tag **RunSearchRequest** now accepts a filter ID - the ID of the filter to be executed. This tag allows one to execute any saved Search Request and then use its results in any JIRA Jelly script.

For instance, one could define a filter identifying all old/inactive issues, and write a Jelly script to move them to an 'Inactive' state (see the Jelly docs for examples). This Jelly script can be scheduled to run periodically with the Jelly service.

SOAP improvements

This release includes various SOAP improvements - including the ability to retrieve via SOAP:

- a list of general configuration properties - i.e. determine if attachments, time tracking, voting, unassigned issues, sub-tasks, issue linking or watching is turned on/off
- an issue by its ID
- a match count for a search filter - useful to determine how results should be displayed - e.g. full or partial list of results

Configure CSV Delimiter

Some versions of MS Office Excel (e.g. German) save CSV files with separating values using '**;**' instead of '**,**'. The CSV Import Wizard has been improved to allow the delimiter to be specified for a particular import file.

Mantis Custom Field Import

Mantis 0.18+ has a basic custom field implementation. The JIRA Mantis Importer can now import any defined custom fields.

Plugin System Improvements

The ability to create more powerful plugins is now possible through the following improvements:

- Reference downloadable external resources. Additional static files such as images, Javascript or CSS can be served through **Downloadable Plugin Resources**. Further details available here.
- The plugin configuration screen has been improved with the ability to include a checkbox parameter. Further details available here.
- Servlet Plugin module enables users to deploy Java servlets as part of a plugin. Further details available here.

Maybe these improvements will be of use to the entrants of the Codegeist Competition?

Configure Email Address Format

Previously, it was necessary to stop JIRA and edit a properties file to modify the email address format. JIRA 3.5 now allows this modification to be made through the web interface - without the need to stop the server.

Internationalisation

The **Administration** section of the JIRA User Interface is now fully internationalized. This makes it possible to translate and present the entire JIRA web interface in a particular language, once the appropriate set of language files are included.

Further details on the translation process and how to get involved are available here. The continued support of those users who volunteer their translation skills is greatly appreciated.

Multi-Select Version Picker Custom Field

A new multi-select **Version Picker Custom Field** is now available - allowing this field to store multiple versions related to the associated project. It is also possible to use this field within a search through the Issue Navigator.

Delete Trackbacks

Trackback management has been improved to allow users delete specific trackbacks.

Configured Navigator Columns in Email Filter Subscription

In JIRA Enterprise edition, it is possible to configure which issue fields are displayed to the user for each saved filter. Previously, this configuration was only used throughout the web interface. JIRA 3.5 Enterprise now respects the chosen fields when sending e-mail notifications to subscribers of a filter.

Convert Bugzilla text links to JIRA Issue links

On import, original Bugzilla text links will be linked to the new JIRA issues. This improvement helps ease the migration to JIRA for users who still use the Bugzilla bug IDs.

Bug Fixes

This release includes over 50 bug fixes. To view the entire bug fix list - ask JIRA!

JIRA 3.5 Upgrade Guide

JIRA 3.5 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5 (release notes) from JIRA 3.4.3. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

JIRA 3.5 Jira Service extension

- If you have implemented a custom JIRA service you need to be aware of the following API change.

In JIRA 3.5 the `getName()` and `setName(String name)` methods was added to the `com.atlassian.jira.service.JiraService` interface. This method should return and set the name of the service respectively. The name of the service can be used to identify a service uniquely. (Fixed made due to JRA-8352 bug)

Therefore, if you have implemented this interface, you will need to implement these methods and recompile your service(s) before deploying it into JIRA 3.5. If you have extended a JIRA class instead, e.g. `com.atlassian.jira.service.AbstractService` or `com.atlassian.jira.service.JiraServiceContainer` you do not need to modify your custom services.

Introduction of global Bulk Change permission

JIRA 3.5 introduces the global **Bulk Change** permission. This permission governs the ability to execute the bulk change operations:

- Workflow Transition
- Edit
- Move
- Delete

An upgrade task has been added to grant the new **Bulk Change** permission to all groups with the global **JIRA Users** permission.

The JIRA documentation includes further details on this new permission.

The decision to grant the **Bulk Change** permission should be considered carefully - the permission permits a user to modify a collection of accessible issues at once. For example, in JIRA installations configured to run in 'Public' mode (anybody can sign up and create issues), a user could comment on all accessible issues with the **Bulk Change** and **Add Comments** permission. Undoing such modifications may not be possible through the JIRA UI and may require changes made directly against the database.

CustomFieldPersister changes

CustomFieldPersister is used to store custom field values to database. The methods of this class has been refactored to remove the redundant parameter, `defaultValueMarker`. For example, the create values method went from:

```
void createValues(CustomField field, Long issueId, String defaultValueMarker,
PersistenceFieldType persistenceFieldType, Collection values, String parentKey);
```

to:

```
void createValues(CustomField field, Long issueId, PersistenceFieldType persistenceFieldType,
Collection values, String parentKey);
```

You will need to update and recompile any `CustomFieldType` that you wrote to use this new interface.

VersionCFType Changes

This affects plugin writers who uses the version custom field `VersionCFType`. The change is that previously the Transport Object type was a single `Version` object, but it is now a collection that contains a single `Version` object.

This was done to handle an improved version custom field which can be a mutli-select version custom field as well

JIRA 3.5.3 Release Notes

JIRA 3.5.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.5.3 in Standard, Professional and Enterprise editions. This point release includes over 20 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version please read through the JIRA 3.5.3 Upgrade Guide.

JIRA 3.5.3 includes over 20 bug fixes and improvements.

JIRA Issues (23 issues)					
Type	**Key**	**Summary**	**Priority**	**Status**	**Resolution**
	JRA-9691	jira.projectkey.warning and jira.projectkey.description properties not used		Resolved	Fixed
	JRA-9689	Jira says JAVA_HOME contains spaces even if it doesn't		Resolved	Fixed
	JRA-9670	Integrity Checker fails when correcting the default dashboard		Resolved	Fixed
		When an issue type is not specified the CSV importer grabs the first it sees, but this			

	JRA-9666	may not be valid because of issue types per project		Resolved	Fixed
	JRA-9643	Update docs to reflect the renaming of 'Developer Workload'		Closed	Fixed
	JRA-9640	Add to OS section to Requirments page		Resolved	Fixed
	JRA-9613	Found typo in a file		Resolved	Fixed
	JRA-9586	Bugzilla doesn't import priorities		Resolved	Fixed
	JRA-9582	UpgradeTask 132 fails if "Assign Issue Screen" does not have a description		Resolved	Fixed
	JRA-9578	initiating user detail not properly shown in usercreated email		Resolved	Fixed
	JRA-9575	Permission-Bug in Manage Watcher List		Resolved	Fixed
	JRA-9562	@123,456@ in Wiki renderer results in a text field		Resolved	Fixed
	JRA-9548	Custom field values for multi selects not returned in SOAP		Resolved	Fixed
	JRA-9544	file "\includes\snippets\listbyname.jsp" is missing taglib identifier		Resolved	Fixed
	JRA-9505	Delete component displays erratic message		Resolved	Fixed
	JRA-9476	Workflow cache not updated on project creation.		Resolved	Fixed
	JRA-9465	Update global permissions of the documentation to include teh Bulk Change permission		Closed	Fixed
	JRA-9452	Last two builds JIRA won't allow setting an issue to unassigned as a Post function in workflow transitions		Resolved	Fixed
	JRA-9435	Add to Upgrade Guide advice that IssueEvent changelog can now be null		Resolved	Fixed
	JRA-9430	Add documentation that specifys that text emails for subscriptions do not behave the same way as html emails		Resolved	Fixed
	JRA-9399	Bulk Move - New security level field values are shown from originating project, not the target project		Resolved	Fixed
	JRA-9394	Security Bug: Deletion of a comment made to jira-developers is visible to all jira-users when viewing change history		Resolved	Fixed
	JRA-9285	Project portlet doesnt remember the [hide] [show>>] settings upon refresh		Resolved	Fixed

JIRA 3.5.3 Upgrade Guide

JIRA 3.5.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5.3 from JIRA 3.5.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.5.3 from JIRA 3.5.2.

JIRA 3.5.2 Release Notes

JIRA 3.5.2 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.5.2 in Standard, Professional and Enterprise editions. This point release includes over 20 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version please read through the JIRA 3.5.2 Upgrade Guide.

JIRA 3.5.2 includes over 20 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.5.2 Upgrade Guide

JIRA 3.5.2 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5.2 from JIRA 3.5.1. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Issue Event Changelog Can Now Be Null

- If you have implemented a custom JIRA Issue Event Listener you need to be aware of the following API change.

In JIRA 3.5.2, the IssueEvent object thrown as a result of an edit operation, **may now return null** from a `getChangeLog()` call. The case where this happens is when a user chooses to edit an issue but only leaves a comment and makes no other changes to the issue. Prior to 3.5.2 no event was fired in this case and this was identified as a bug (JRA-9415) and has since been fixed. Check any calls to getChangeLog() for null.

JIRA 3.5.1 Release Notes

JIRA 3.5.1 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.5.1 in Standard, Professional and Enterprise editions. This point release includes over 20 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version please read through the JIRA 3.5.1 Upgrade Guide.

JIRA 3.5.1 includes over 20 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.5.1 Upgrade Guide

JIRA 3.5.1 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5.1 from JIRA 3.5. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.5 to JIRA 3.5.1.

JIRA 3.4 and 3.4.1 Release Notes

JIRA 3.4 Release Notes

Atlassian Software Systems, Australia's fastest growing software company, is proud to announce the latest release of *the* issue tracking and project management application - **JIRA 3.4** (download it here). Just over 3 weeks since the final 3.3.x release, **JIRA 3.4** includes some **50**

powerful new features and improvements along with over **70** bug fixes.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Upgrade Information

In order to complete a successful upgrade, please refer to our Upgrade Guides. If you are upgrading from JIRA 3.3.3 please refer to the following document:
http://confluence.atlassian.com/display/JIRA/JIRA+3.4+and+3.4.1+Upgrade+Guide

If you are upgrading from a pre-3.3.3 release please read the following guides also:
http://confluence.atlassian.com/display/JIRA/All+JIRA+3.x+Upgrade+Guides

Features and Improvements:

- Issue Types Per Project
- Renderers
- Clone Portlets
- Issue Operation Plugin
- RSS Improvements
- RPC Searching
- Dynamic Header Banner
- Internationalisation
- Bulk Migration by Issue Type
- Change Parent Issue of Sub-Task
- Multi-User Custom Field
- Trackback Configuration

Contents

1. New Features
2. Improvements
3. Bug Fixes
4. Notes

New Features

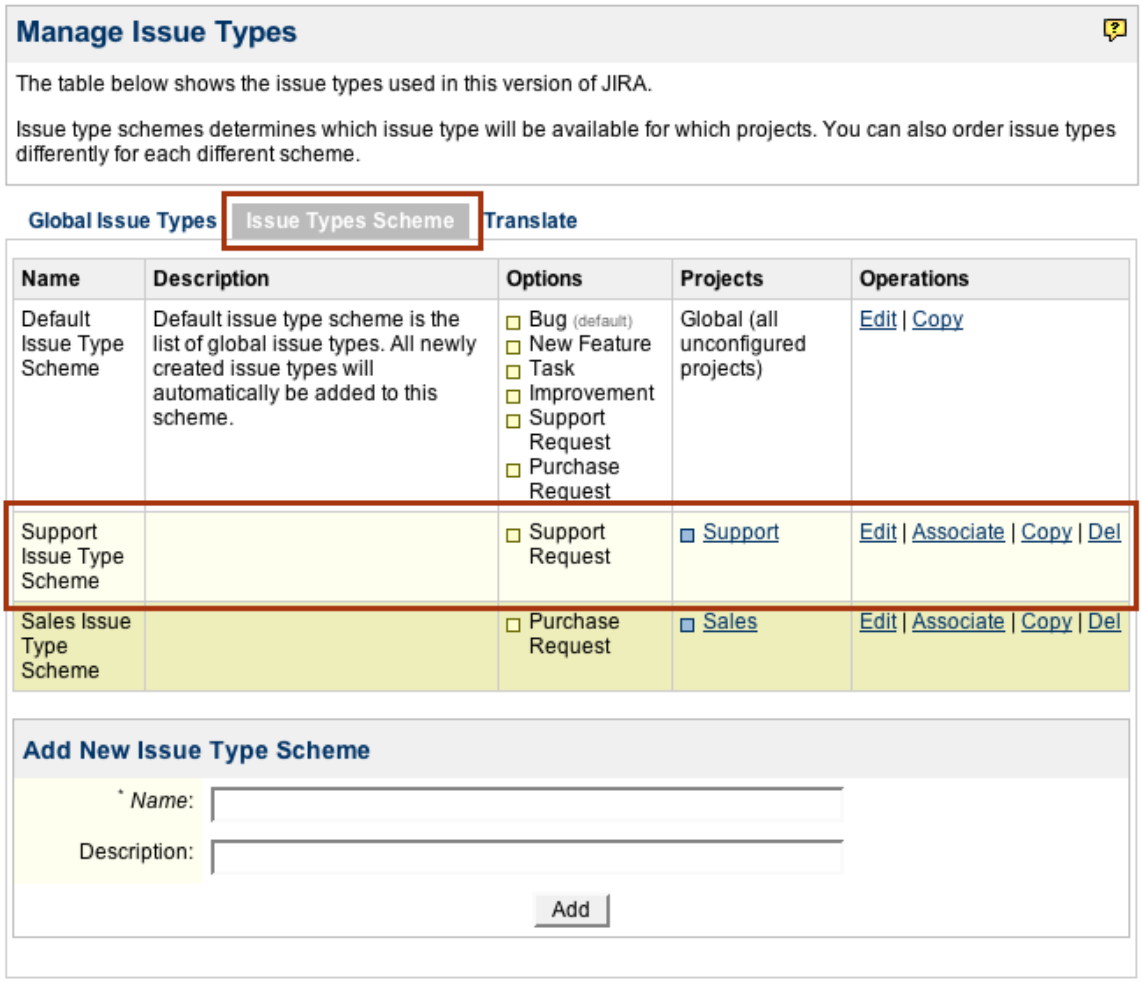
This release includes over 50 new features and improvements - see the full list of features & improvements online

Issue Types Per Project

The **most highly requested** feature in JIRA - with over **200** votes - you can now configure **Issue Types Per Project**. Issue types can be defined for a specific project - adding contextual relevance to issue type selection when creating/editing/moving an issue.

This feature provides the ability to better define the context of a project with only relevant issue types available - resulting in a more intuitive user interface, and enables greater control over the users' issue selection process.

For example, the **Sales** project may require a **Purchase Request** issue type, while this issue type would not apply to the context of the **Support** project. Further, the issue type **Support Request** is relevant within the **Support** project but would not apply to the context of the **Sales** project.



Renderers

This release also introduces **Renderers** within JIRA text-based fields such as description and comments - allowing a greater range of expression within these fields.

This release ships with two renderers, the default text renderer, which preserves the functionality available in previous JIRA releases, and the Atlassian wiki renderer, which brings the power of the Confluence wiki engine to JIRA. In the Enterprise edition of JIRA, renderers can be configured on a per field, per project/issue type level, allowing a flexible combination of text and wiki markup. In the Standard and Professional editions renderers can be configured on a per field basis. Renderers are implemented as pure JIRA plugins, meaning that any renderer can be easily added to or removed from use within JIRA, including any custom renderers that may be developed.

Further details on the renderer functionality is available in the documentation:

http://www.atlassian.com/software/jira/docs/v3.4/renderers_overview.html

Description

Header 1

Header 4

```
public String exampleCode(String test)
{
   return "this is an example";
}
```

inline attachments

green blue red

TST-31

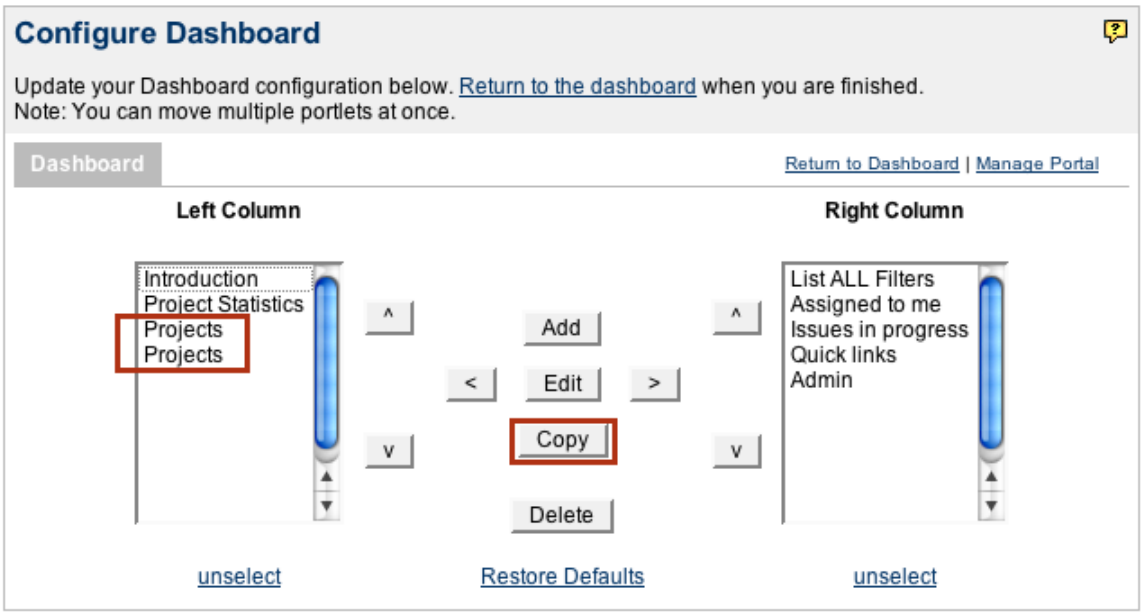
link to google

1. item 1
2. item 2

Day	Working Hours
Mon (Day 1)	8 - 5
Tue (Day 2)	8 - 5
Wed (Day 3)	8 - 5
Thu (Day 4)	8 - 5
Fri (Day 5)	8 - 5

Clone Portlets

Dashboard administration is simplified with the ability to clone a portlet.

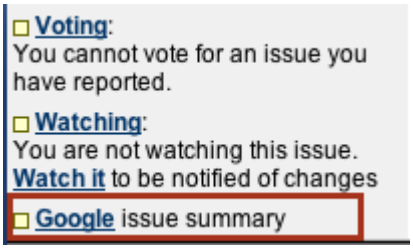


Issue Operation Plugin

This new system plugin allows new issue operations (links and HTML) to be added to the operations panel of the view issue screen. With a simple configuration process, it is possible to add quick access to commonly used links to every issue.

For example, the following operation will execute a **Google** search on the issue summary.

Information about how to create your own Issue Operations can be found in the docs online.

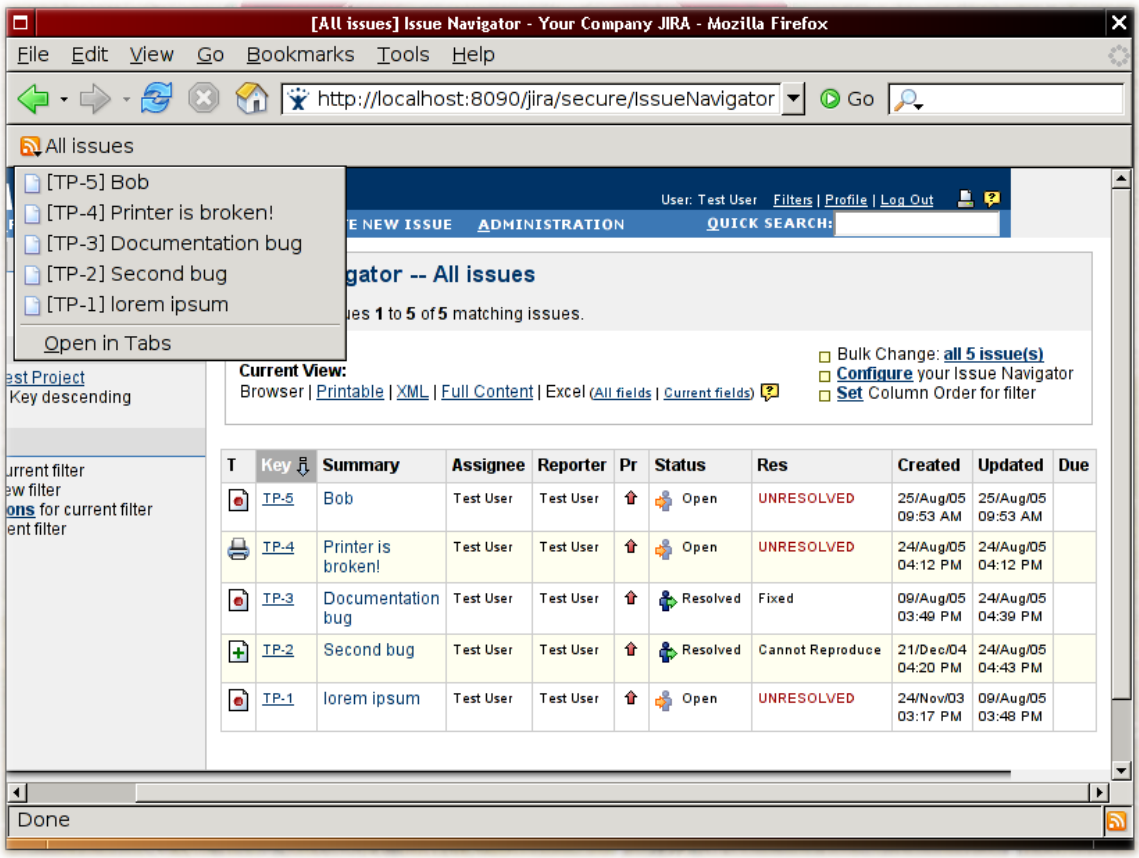


Improvements

RSS Support Improvements

JIRA has been improved to take advantage of browser support for RSS feeds. A result of any search can be accessed quickly and directly by adding 'live bookmarks'.

Basic authentication support for RSS feeds from JIRA has also been added in this release - allowing the option of avoiding the transmission of usernames and passwords across the wire in clear text format.



RPC Searching

The RPC searching functionality has been improved - results are returned quicker and more efficiently with fewer database calls.

Dynamic Announcement Banner

With thanks to Nick Minutello, the announcement banner can now be dynamically configured through the JIRA UI.

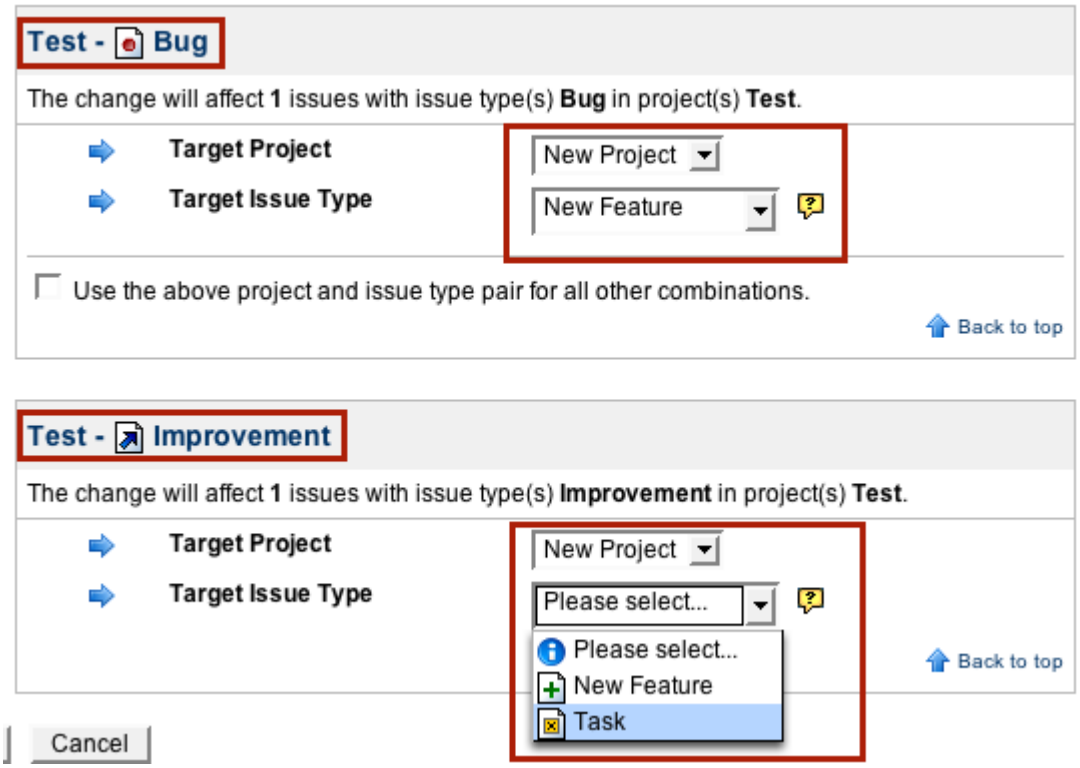
The announcement banner is displayed on all JIRA pages and is useful for alerting users to important information - for example, scheduled server maintenance, approaching project deadline, etc. This information can be updated immediately without having to restart JIRA.

Internationalisation

With the introduction of the **Czech** language pack, JIRA is now available in **16** different languages. This release also includes an updated **Traditional Chinese** language pack. Once again, the Atlassian team would like to thank our users who have contributed to the translation process - this work is much appreciated.

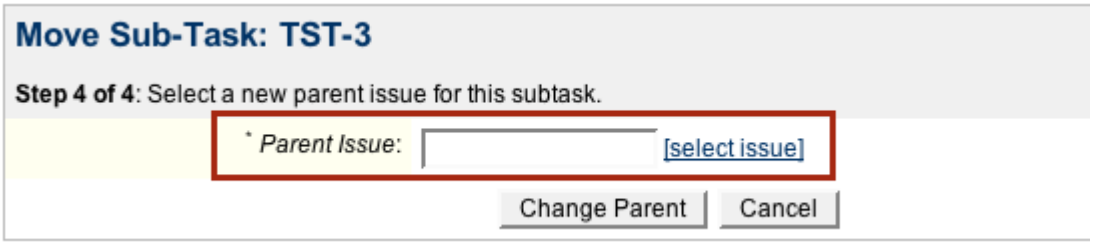
Bulk Migration by Issue Type

Previously, it was only possible to bulk migrate a collection of issues to a single project and issue type. Now, different issue types can be migrated to different projects and issue types - allowing you to specify exactly how each issue should be moved. With this fine-grained configuration capability, bulk migration operations are now easier and quicker.



Change Parent of Sub-Task

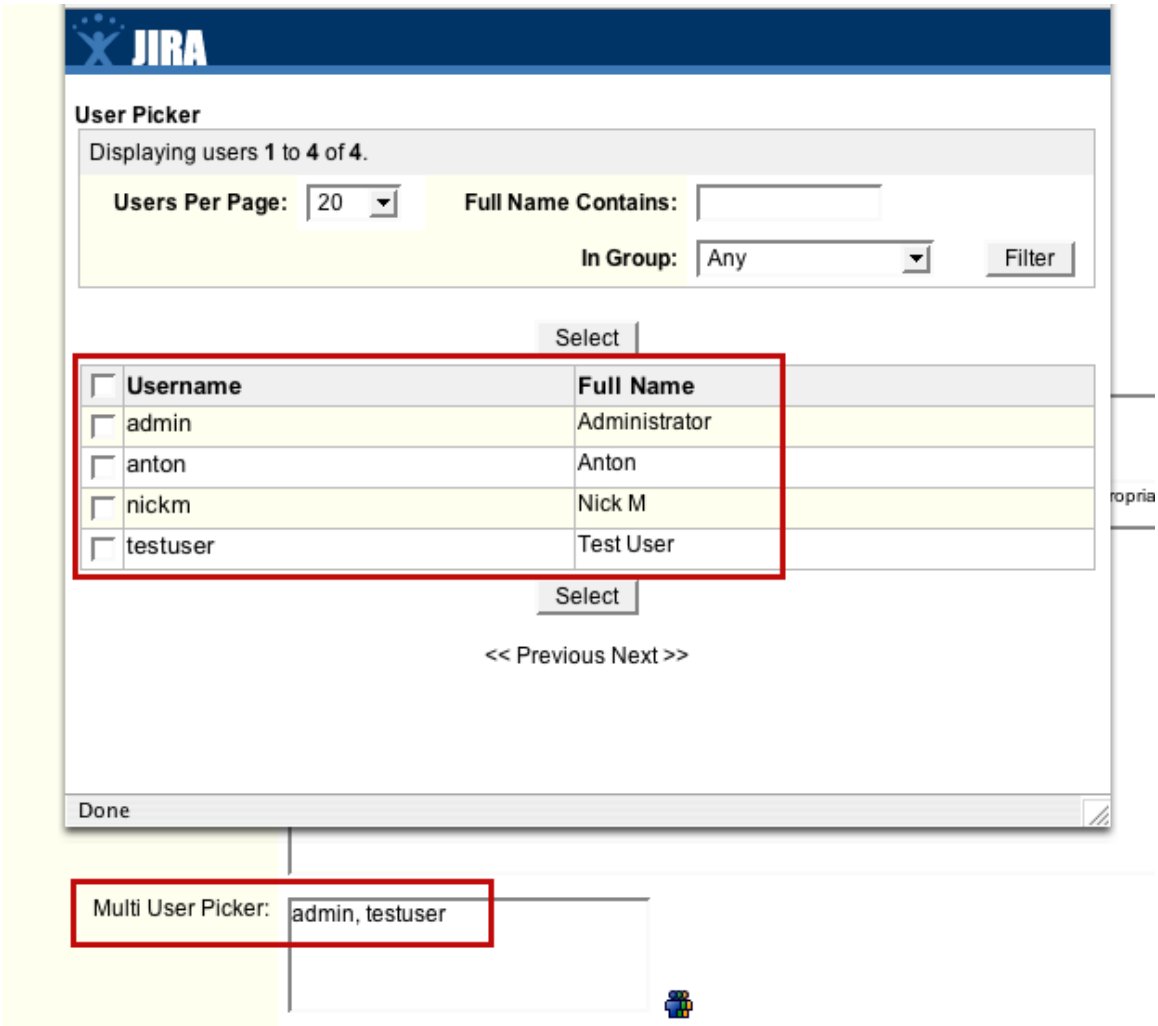
The parent issue of a sub-task can now be changed - allowing a sub-task to be 'moved' from one parent issue to another.



Multi-user Custom Field

It is now possible to select multiple users through the multi-user custom field.

This improvement has also been extended to the configuration of security level, permission and notification schemes. This means that by selecting users for the Multi-user field it is possible to control who will have access to the an issue and who will be notified when the issue is updated. Please note that using Multi-user custom field for permissions and security levels is only supported by the Enterprise Edition.



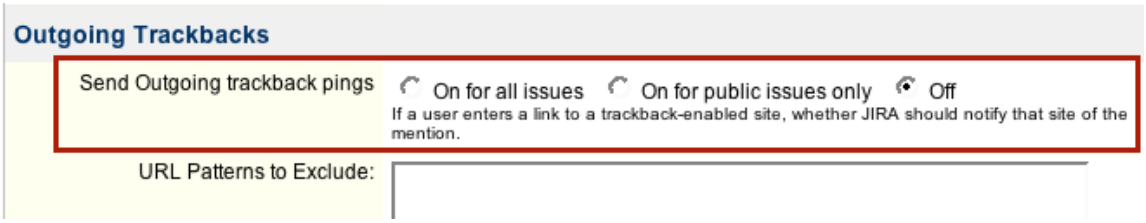
Trackback Configuration

Trackback configuration has been extended with three possible modes for **Outgoing Trackbacks**:

- **Off for All Issues**
- **On for Public Issues Only**
- **On for All Issues**

By using the **On for Public Issues Only** setting, trackbacks from issues protected by a security or permission scheme (i.e. issues not visible by non-logged in users) will not be sent - the external sites referenced in these secure issues will not be notified of the referral.

The operation of **Incoming Trackbacks** has also been modified such that 'secure' issues can receive trackback notifications. The issues will remain secure based on their security settings, but users who have access to the issues will be able to see the trackback references that issues have received.



Notable Features & Improvements

Some other notable improvements included in this release:

- New Jelly tag allowing a field (including custom field) to be placed onto a Screen
- Bugzilla importer preserves inter-issue dependencies
- Bugzilla importer converts Bugzilla text links to JIRA links
- Search user custom fields using group membership

Bug Fixes

This release includes over 70 bug fixes - the full list of bug fixes online

Notes

Javascript Caching

In some instances, the browser may read from a cached version of the Javascript files used throughout JIRA. This will result in certain elements appearing incorrectly - e.g. the issue type selection list may appear with repeated icons. This problem can be corrected by forcing a reload on the page in question - i.e. pressing 'CTRL-R' or 'F5'.

JIRA 3.4.1 Release Notes

JIRA 3.4.1 is released today in Standard, Professional and Enterprise editions. This point release fixes JIRA 3.4's incompatibility with MySQL and MS SQL Server and includes bug fixes and improvements which appear below.

If upgrading from JIRA 3.3.3 or 3.4 please read through JIRA 3.4 and 3.4.1 Upgrade Guide. If upgrading from an earlier version of JIRA please read through all the Upgrade Guides for all versions that you are skipping during the upgrade.

JIRA 3.4.1 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.
Error rendering macro 'jiraissues' : java.lang.NullPointerException

JIRA 3.4 and 3.4.1 Upgrade Guide

JIRA 3.4 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.4 from JIRA 3.3.3. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

> Two major new features of JIRA 3.4, wiki renderer previews, and issue types per project require that javascript be enabled to make use of their full functionality. You will still be able to use all the core features of JIRA with javascript disabled.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

1. Please do **not** copy `jira-application.properties` file from your old JIRA installation. Edit the file that is shipped with JIRA 3.4 and make needed changes. New properties have been added to this file so if you simply copy the old file across the following error would occur JRA-8645.
2. If you have written any CustomFieldType or CustomFieldSearcher plugins please refer to Upgrading Custom Field Types in JIRA 3.4
3. The default user preferences are now configured in the jira-application.properties file and are configurable through the admin section of JIRA. Any properties in the old file preferences-default.xml will no longer effect JIRA configuration.
4. Please note that to configure issue types per project you must have JavaScript turned on in your web browser.
5. If you are using MySQL please do not use Connector/J 3.1.11 JDBC Driver as it has the following bug. Connector/J 3.1.10 and earlier work fine.

JIRA 3.4.1 Upgrade Guide

This section contains specific information you need to know when upgrading to JIRA 3.4.1 from JIRA 3.4. If upgrading from JIRA 3.3.3 please read the previous section as well. If upgrading from an older version than JIRA 3.3.3, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

1. Please do **not** copy `jira-application.properties` file from your old JIRA installation. Edit the file that is shipped with JIRA 3.4 and make needed changes. New properties have been added to this file so if you simply copy the old file across the following error would occur JRA-8645.
2. If you have written a CustomFieldType that implements the `com.atlassian.jira.issue.customfields.CustomFieldType` interface directly rather than extending one of the Abstract classes that ship with JIRA please read Upgrading Custom Field Types in JIRA 3.4.1.
3. If you have written an Custom Field Searcher please have a look at Upgrading Custom Field Types in JIRA 3.4.1.
4. JIRA 3.4 and 3.4.1 do not generate an Issue Assigned event. The Issue Updated event is generated instead. In previous versions of JIRA the Issue Assigned event was generated when issues are assigned using the "Assign" operation on the View Issue page. This means that even when the "Assign" operation is used JIRA will send notifications to parties listed under the Issue Updated event. The patch to correct this behaviour is available at JRA-8533.

Upgrading Custom Field Types in JIRA 3.4

Unknown macro: {version-warn}

JIRA 3.4 and higher

Changes to the custom field types code

For JIRA 3.4, there has been further upgrades to the custom fields code. If you have written your own Custom Field Types of Custom Field Searchers, you'll want to read this document. We recommend that you ensure that your custom field types compile against the latest 3.4 build and verify this under a testing environment before putting it into production.

You can download the latest JIRA Development Kit from its confluence space.

CustomFieldType Interface changes CustomFieldConfig and CustomFieldConfigItemType changed to FieldConfig and FieldConfigItemType

Note that the references to `CustomFieldConfig` and `CustomFieldConfigItemType` has become `FieldConfig` and `FieldConfigItemType` respectively. This change is to bring CustomFields and SystemFields closer so that, in the future, fields can be configured in uniform way.

```
com.atlassian.jira.issue.customfields.config.CustomFieldConfig;
com.atlassian.jira.issue.customfields.config.CustomFieldConfigItemType;
```

has become

```
com.atlassian.jira.issue.fields.config.FieldConfig;
com.atlassian.jira.issue.fields.config.FieldConfigItemType;
```

A straight replacement should do the trick.

isRenderable()

With the addition of renderers in JIRA 3.4 there has been a minor addition to the CustomFieldType interface. There is a now a method of the signature:

```
public boolean isRenderable();
```

There is a default implementation of this method in the AbstractCustomFieldType class that returns false. If you are extending any of the existing base classes, AbstractCustomFieldType, TextCFType, SelectCFType, then there are no changes needed to upgrade your CustomField Types.

If you are implementing the CustomFieldType interface then you will need to implement the isRenderable method in your class. To retain the functionallity that you always had the method should return false. With the introduction of renderable fields in JIRA 3.4 if you return true in the isRenderable method then you will be able to assign the custom field a renderer. This only really makes sense for text fields where you can use the power of the wiki syntax.

getChangelogString()

To better support logging of change history from custom fields there is now a method of the signature:

```
public String getChangelogString(CustomField field, Object value);
```

This method allows a custom field type to provide a string that can give a custom fields value meaning. The two examples that ship with JIRA are the ProjectPicker and VersionPicker custom fields. Both of these fields store the database id of the value (e.g. My Project is id 10000, so it stores 10000). The ProjectCFType object will return the value 'My Project' from this new method so that we can register in the change item bean that the string is 'My Project' and that the value is '10000'. The AbstractCustomFieldType object returns null for this method by default. If you are extending any of the existing base classes, AbstractCustomFieldType, TextCFType, SelectCFType, then there are no changes needed to upgrade your CustomField Types.

Upgrading Custom Field Types in JIRA 3.4.1

Upgrading Custom Field Types in JIRA 3.4.1

valuesEquals method addition

If you have implemented a CustomFieldType by directly implementing the `com.atlassian.jira.issue.customfields.CustomFieldType` interface instead of extending one of the Abstract classes that ship with JIRA, you will need to add the following method to your implementation:

```
/**
 * Used to compare old field value to the new field value when the issue is being updated
 * and work out whether a change item should be generated.
 * @param v1 current value
 * @param v2 new value
 * @return true if the change item should be generated, false otherwise
 */
boolean valuesEqual(Object v1, Object v2);
```

This means that you will need to implement this method and recompile your class before deploying it into JIRA 3.4.1.

The method has been introduced to resolve bug JRA-8480 and allows the CustomFieldType to have control over change items that JIRA generates when an issue is updated.

The `com.atlassian.jira.issue.customfields.impl.AbstractCustomFieldType` implements the method as follows:

```
if (v1 == v2)
{
    return true;
}

if (v1 == null || v2 == null)
{
    return false;
}

return v1.equals(v2);
```

which is compatible with the behaviour of previous JIRA releases. Therefore if your CustomFieldType extends AbstractCustomFieldType, or any of its subclasses, you do not need to add this method, unless you would like to take advantage of it.

CustomFieldSearchers - AbstractCustomFieldSearcher constructor change

If you have implemented any CustomFieldSearchers that extend the com.atlassian.jira.issue.customfields.searchers.AbstractCustomFieldSearcher class then you will have to modify your CustomFieldSearcher. The constructor of the AbstractCustomFieldSearcher class has changed from being a no-arg constructor to taking an instance of com.atlassian.jira.web.bean.FieldVisibilityBean. You must pass the FieldVisibilityBean to the super classes constructor. You will need to include code that will look something like this:

```
public MultiProjectSearcher(FieldVisibilityBean fieldVisibilityBean)
{
   super(fieldVisibilityBean);
}
```

JIRA 3.4.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.4.3 Release Notes

In the tradition of worthwhile updates, JIRA 3.4.3 is released today in Standard, Professional and Enterprise editions. This point release includes over 40 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version than JIRA 3.4.2 please read through the Upgrade Guides for all versions that your are skipping during the upgrade. If upgrading from JIRA 3.4.2 please read the 3.4.3 Upgrade Guide before continuing.

JIRA 3.4.3 includes over 40 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.4.3 Upgrade Guide

JIRA 3.4.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.4.3 from JIRA 3.4.2. If upgrading from an older version

of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.4.2 to JIRA 3.4.3.

JIRA 3.4.2 Release Notes

JIRA 3.4.2 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

In the tradition of worthwhile updates, JIRA 3.4.2 is released today in Standard, Professional and Enterprise editions. This point release includes over 35 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version than JIRA 3.4.1 please read through the Upgrade Guides for all versions that your are skipping during the upgrade. If upgrading from JIRA 3.4.1 please read the 3.4.2 Upgrade Guide before continuing.

JIRA 3.4.2 includes over 35 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.4.2 Upgrade Guide

JIRA 3.4.2 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.4.2 from JIRA 3.4.1. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.4.1 to JIRA 3.4.2.

JIRA 3.4.1 Release Notes

JIRA 3.4.1 Release Notes

JIRA 3.4.1 is released today in Standard, Professional and Enterprise editions. This point release fixes JIRA 3.4's incompatibility with MySQL and MS SQL Server and includes bug fixes and improvements which appear below.

If upgrading from JIRA 3.3.3 or 3.4 please read through JIRA 3.4 and 3.4.1 Upgrade Guide. If upgrading from an earlier version of JIRA please read through all the Upgrade Guides for all versions that you are skipping during the upgrade.

JIRA 3.4.1 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.
Error rendering macro 'jiraissues' : java.lang.NullPointerException

JIRA 3.3 Release Notes

JIRA 3.3 Release Notes

Atlassian Software Systems is proud to announce the latest release of *the* issue tracking and project management application - **JIRA 3.3** - download it here. Only 10 weeks since the last release, **JIRA 3.3** includes some of the most requested features along with a host of bug fixes and improvements.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Upgrade Information

In order to complete a successful upgrade, please refer to the following guides:

- JIRA 3.2 Upgrade Guide
- JIRA 3.3 Upgrade Guide

It is necessary to follow both guides if upgrading from a **pre 3.2.x** version. It is only necessary to follow the JIRA 3.3 Upgrade Guide when upgrading from a **3.2.x** version.

Features and Improvements:

- Multiple Project Filters
- Bulk Move
- User Custom Field Notification Target
- Extended Search Capabilities
- SOAP Enhancements
- Performance Improvements
- New Translations

Contents

1. New Features
2. Improvements
3. Bug Fixes

New Features

This release of JIRA includes some of the most requested features as logged at http://jira.atlassian.com - the full list can be viewed at JIRA 3.3 New Features & Improvements.

Multiple Project Filters

The Issue Navigator has been completely overhauled to pave the way for future extended search functionality and enhancements. Addressing one of the most popular feature requests (with over 50 votes), this release includes the ability to execute a **search across multiple projects**.

This feature increases the searching capabilities available to the user with the ability to aggregate a collection of issues from across multiple projects. Users can now design a search query that directly addresses their requirements for searching throughout JIRA.

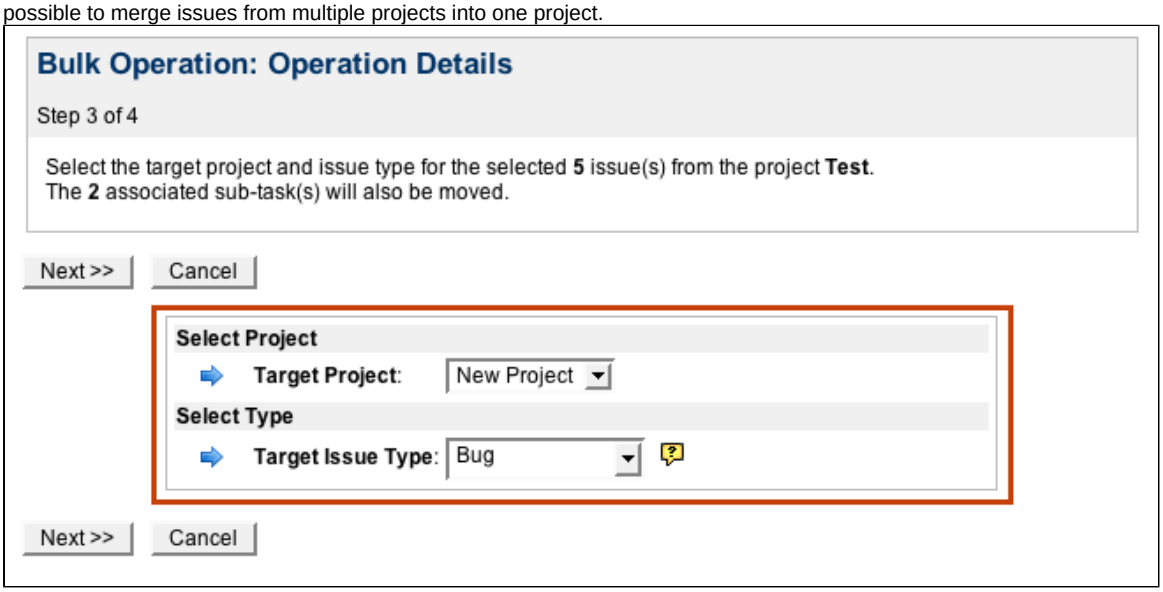
The applications of this improved search capability are countless ... for example:

- Retrieve all issues opened in the last week from selected projects - add these results to an RSS feed or view them on the JIRA dashboard
- Add a statistics portlet to the JIRA dashboard with results from a multi-project filter
- Subscribe to a multi-project filter containing issues assigned to your team members - with issues retrieved only from the projects you manage!multiplesearch.png|align=center,border=1!

Bulk Move

Extending the range of bulk operations available in JIRA, it is now possible to move multiple issues at once.

The Bulk Move operation allows a collection of issues (from multiple projects and consisting of multiple issue types) to be moved to another project and/or issue type. The ability to update multiple issues in this manner gives the user even more power - for instance, it is now possible to merge issues from multiple projects into one project.



User Custom Field as Notification Target

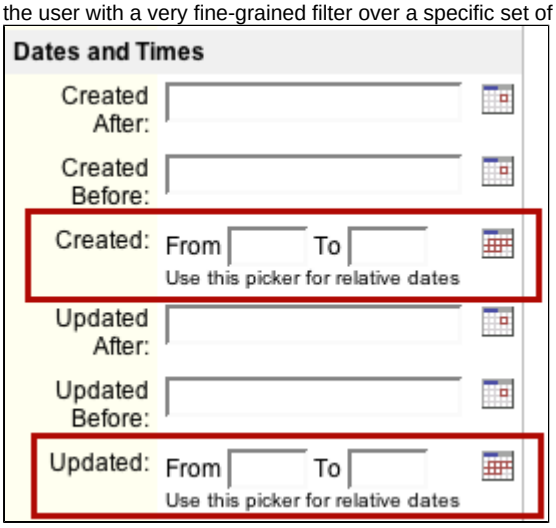
It is now possible to specify an additional notification target by selecting a user in a '**CC**' custom field. This provides the issue creator/editor the ability to add another user to the notification recipient list for a specific issue. Many thanks to Chris Wood at MetOcean Engineers for his work on this feature!

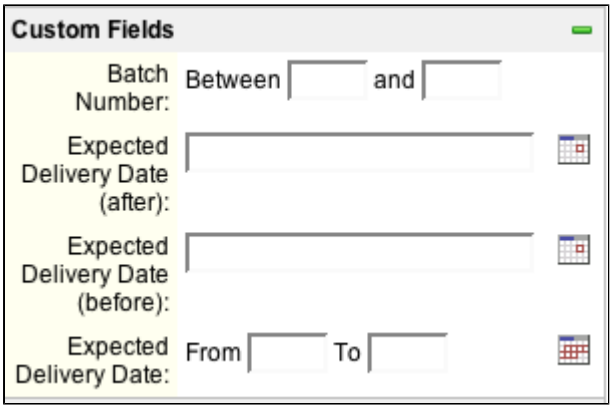
Improvements

This release of JIRA includes some significant improvements in available functionality and overall system performance.

Extended Search Capabilities

More precise search filters can be created by specifying a date range in relation to the system fields '**Created**' and '**Updated**' and the custom field '**Date Time**'. A number range can also be specified for the '**Number**' custom field. The ability to specify a range in this manner provides the user with a very fine-grained filter over a specific set of issues.





SOAP Enhancements

JIRA's SOAP capabilities are growing more complete with each release. For JIRA 3.3, you can now progress an issue through a workflow remotely; find out what actions are available for an issue, update the fields and progress through the workflow. It's now easier than ever to seamlessly integrate JIRA with external tools.

Extended XML-RPC Functionality

Bringing the XML-RPC functionality inline with the current SOAP plugin, it is now possible to update an issue & run a search request.

Performance Improvements

Continually striving to advance system efficiency, this release includes a number of notable performance improvements:

- **Issue Navigator** performance has been improved with fewer database access calls - displaying filter results quicker
- **Workflow Activation** is now less memory intensive - the operation completes without pulling all issues into memory to make the new workflow association
- **Version Management** operations have been streamlined to complete quicker
- The **Link Issue** pop-up displays quicker
- The **Permission Schemes** and **Notification Schemes** management pages have been refined to return scheme information quicker
- **Date Range** searches are executed more efficiently

New Translations

Along with updates to the **Traditional Chinese** and **German** translations, JIRA is now available to use in **Italian** and **Slovakian**. Once again, we would like to thank all those who have contributed to the translation process - JIRA is now available in **14** different languages.

JIRA Standalone

JIRA Standalone now ships with Tomcat 5.5 - allowing users to use JIRA out of the box with the latest version of the popular application server.

> **Previous Standalone Installations**
>
> Due to incompatibilities in the format of the **server.xml** file between versions of Tomcat - please **do not copy** the **server.xml** file from previous standalone installations to the new standalone installation.

Bug Fixes

This release includes over 90 bug fixes - the best way to see them is them is to ask JIRA - Issues Resolved for JIRA 3.3.

JIRA 3.3 Upgrade Guide

JIRA 3.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.3 from JIRA 3.2.x. If upgrading from an older version of JIRA, please go to the complete list of Upgrade Guides, and read the notes for each version you are skipping during the upgrade.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Known incompatibilities

3.3.x is not a good release for IBM shops:

1. JIRA 3.3.x may not work on **Websphere 5.0.x and 5.1.x** due to JRA-7699
2. When using DB2, JIRA may hang when deleting projects or performing workflow operations. See the full problem description (and possible workaround) in the documentation

Websphere or DB2 users, please stick with 3.2.x or move on to 3.4.x or higher, where these problems have been resolved.

Notes on upgrading

1. Due to web browser caches, changes to JIRA's Issue Navigator might appear corrupted or unstyled. Please refresh your browser's cache (press *Shift+Reload* on the Find Issue's page) for the changes to appear correctly.
2. JIRA's **issue cache** size will be automatically set to 0 during the upgrade, as it is no longer needed due to performance improvements in JIRA (JRA-7166)
3. If you have written any CustomFieldType or CustomFieldSearcher plugins please refer to this document
4. Users with **outgoing** trackback pings enabled (not the default) may wish to disable this until JRA-7589 is fixed, to avoid the risk of the mail queue hanging.
5. If you have bookmarks or deal with hard coded links to the issue navigator, you should read about the changed issue navigator parameters
6. If you are **using JIRA Standalone**, please do *not* simply copy your old `conf/server.xml` file to the new installation of JIRA. Please read this document.
7. If upgrading JIRA in an external Tomcat installation, be sure to delete the `work/` temporary directory before restarting JIRA, to clear cached JSPs from the old JIRA.

Parameter changes in Issue Navigator

We've made significant backend changes to the issue navigator in 3.3. This resulted in some parameters being changed and are deprecated.

What does this affect?

This affects only direct links to the issue navigator that's been saved outside JIRA. e.g. a bookmark to an RSS feed, a Confluence page with the search parameters hard coded. This will not affect saved filters in JIRA, or portlets shipped with JIRA.

What has changed?

For 3.3 parameters that have changed are:

1. resolutionIds -> resolution
2. priorityIds -> priority
3. statusIds -> status
4. createBefore -> create:before
5. createAfter -> create:after
6. createPrevious -> create:previous
7. updateBefore -> update:before
8. updateAfter -> update:after
9. updatePrevious -> update:previous
10. duedateBefore -> duedate:before
11. duedateAfter -> duedate:after

12. duedatePrevious -> duedate:previous
13. duedateNext -> duedate:next

Also the values for the createNext and updateNext parameters has been modified.

What this means for me? What do I need to do?

We've put in place mechanisms that makes the issue navigator backwards compatible, so you won't actually notice any difference using links with deprecated parameters. However, it's **strongly recommended** that you re-bookmark any affected links to JIRA. We can't guarantee that this will be in place forever and it's better if you update it as a soon as possible.

For system administrators, all searches using the deprecated parameters will be logged to the server with the client's URL and IP address. You should try to chase up the user so that there's no major problems down the track.

If you find these warnings impossibly annoying, you can update your `log4j.properties` to disable them by adding the line below to your log4j.properties. However, we **strongly discourage** you from doing this. The warnings are there so that they can be identified and stop any problems further down the track.

```
log4j.category.com.atlassian.jira.util.retro = ERROR, console
```

Upgrading custom CustomFieldTypes in JIRA 3.3

Applies to JIRA 3.3 and higher

Changes to the custom field types code

For JIRA 3.3, there has been further upgrades to the custom fields code. If you have written your own Custom Field Types of Custom Field Searchers, you'll want to read this document. We recommend that you ensure that your custom field types compile against the latest 3.3 build and verify this under a testing environment before putting it into production.

You can download the latest JIRA Development Kit from its confluence space.

CustomFieldType Interface changes

We removed all references to GenericValue objects in the CustomFieldType and replaced it with Issue. The issue object provides powerful accessors to the issue's parameters such as affects versions and components. If you still need the old GenericValue object, you can call `getGenericValue` on the issue object.

We've also added a new method to the interface.

```
/**
     * Returns a list of indexers that will be used for the field. This will over-ride the
anonymous searcher specified
     * by {@link AbstractCustomFieldSearcher#getRelatedIndexers()} and {@link
AbstractCustomFieldSearcher#index(Document,
     * CustomField, Object)}
     *
     * @return List of instantiated and initialised {@link FieldIndexer} objects. Null if no
related indexers.
     */
    List getRelatedIndexers(CustomField customField);
```

Through this method you can over-ride how this custom field will get indexed.

CustomFieldSearcher Interface over-haul

The CustomFieldSearcher interface now extends the new IssueSearcher interface. This allows you to write more flexible searchers as well as easily extend and reuse code from the default system searchers.

If you haven't developed your own custom searchers, then there's nothing you need to do. If you do have custom searchers but they extend AbstractCustomFieldSearcher then you shouldn't have to make any changes to the Java code as we have tried to keep this class backwards compatible. However, you should still recompile to make sure.

You will need to update all velocity pages used by the searcher. For the edit pages you'll need to use the new header (rather than the controlHeaders)

```
#searcherEditHeader ($customField.id $customField.name)

...

#searcherEditFooter ($customField.id $customField.description)
```

and for the view templates:

```
#searcherHeader ($customField)

...

#searcherFooter ($customField)
```

This change is in order to give the search templates greater flexibility (i.e. allow different rendering behaviour of the searcher from the standard edit screens.

Upgrading to JIRA 3.3 Standalone

Upgrading to JIRA 3.3 Standalone

JIRA Standalone now ships with Tomcat 5.5 - allowing users to use JIRA out of the box with the latest version of the popular application server.

Previous version of JIRA shipped with an older version of Tomcat, for example, JIRA 3.2.x shipped with Tomcat 4.1.29. The configuration files for Tomcat 5.5 are **not compatible** with files from older Tomcat releases.

This means that you cannot simply copy the `conf/server.xml` from your old JIRA Standalone installation. You will need to use the `conf/server.xml` shipped with JIRA 3.3 Standalone as a starting point, and then customise it for your database as described in our documentation.

JIRA 3.3.3 Release Notes

JIRA 3.3.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

In the tradition of worthwhile updates, JIRA 3.3.3 is released today in Standard, Professional and Enterprise editions. This point release includes over 70 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version than JIRA 3.3.2 please read through the Upgrade Guides for all versions that your are skipping during the upgrade. If upgrading from JIRA 3.3.2 please read the 3.3.3 Upgrade Guide before continuing.

JIRA 3.3.3 includes over 70 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.3.3 Upgrade Guide

JIRA 3.3.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.3.3 from JIRA 3.3.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

1. the release notes templates, releasenotes-html.vm and releasenotes-text.vm are no longer provided with an Issue GenericValue, they are now provided an Issue object. If you have customized these templates you will need to modify your velocity syntax to access the Object methods instead of accessing the values via the GenericValue.

JIRA 3.3.2 Release Notes

JIRA 3.3.2 Release Notes

In the tradition of worthwhile updates, JIRA 3.3.2 is released today in Standard, Professional and Enterprise editions. This point release

includes over 40 bug fixes and improvements. It can be downloaded here.

If upgrading from an earlier version than JIRA 3.3.1 please read through the Upgrade Guides for all versions that your are skipping during the upgrade. If upgrading from JIRA 3.3.1 there is no specific Upgrade Guide for JIRA 3.3.2.

JIRA 3.3.2 includes over 40 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.3.1 Release Notes

JIRA 3.3.1 Release Notes

In the tradition of worthwhile updates, JIRA 3.3.1 is released today in Standard, Professional and Enterprise editions. This point release includes over 30 bug fixes and improvements. It can be downloaded here. See the JIRA 3.3 Upgrade Guide and JIRA 3.3.1 Upgrade Guide before upgrading.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.3.1 includes over 30 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.3.1 Upgrade Guide

JIRA 3.3.1 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.3.1 from JIRA 3.3.

If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below:

1. If you have implemented a cutom Issue Tab Panel plugin you need to be aware of this API change.

If you are upgrading to JIRA 3.3.1 from a previous version, due to web browser caches, changes to JIRA's Issue Navigator might appear corrupted or unstyled. Please refresh your browser's cache (press Shift+Reload on the Find Issue's page) for the changes to appear correctly.

JIRA 3.3.1 Issue Tab Panel extension

Unknown macro: {version-warn}

JIRA 3.3.1

JIRA 3.3.1 Issue Tab Panel extension

You only need to worry about this page if you have a implemented a custom [Issue Tab Panel] JIRA plugin

Before JIRA 3.3.1 every `IssueAction` object that a `IssueTabPanel` returned would appear on the "All" tab on the View Issue page. Therefore an Issue Tab Panel could not return an IssueAction with a simple informational message, without poluting the "All" tab with this message. (This caused the JRA-7822 bug).

In JIRA 3.3.1 the `isDisplayActionAllTab()` method was added to the `com.atlassian.jira.issue.action.IssueAction` interface. This method should return "true" if the `IssueAction` should appear on the "All" tab, and "false" otherwise.

Therefore, if you have implemented this interface directly in your plugin, you will need to implement this method and recompile your plugin before deploying it into JIRA 3.3.1. If you have extended a JIRA class instead, e.g. `com.atlassian.jira.issue.action.AbstractIssueAction` or `com.atlassian.jira.issue.action.AbstractGVIssueAction` you do not need to recompile your plugin.

JIRA 3.2 Release Notes

JIRA 3.2 Release Notes

Atlassian Software Systems is proud to announce the latest release of *the* issue tracking and project management application - **JIRA 3.2** (

download). Following the tradition of frequent and worthwhile upgrades, **JIRA 3.2** once again raises the bar in the professional issue tracking arena with a host of new features, improvements and bug fixes. Included in this release:

- Customisable fields for transitions
- Tabbed field screens
- Contextual custom fields
- Extended Bulk Edit Capabilities
- Improved internationalisation
- Improved performance
- ...and much more!

JIRA just got a whole lot better!

Upgrading to JIRA 3.2?

Please refer to the following documentation regarding the upgrade process to JIRA 3.2 from previous versions - JIRA 3.2 Upgrade Guide

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Contents

1. New Features
2. Improvements
3. Bug Fixes

New Features

JIRA 3.2 Documentation

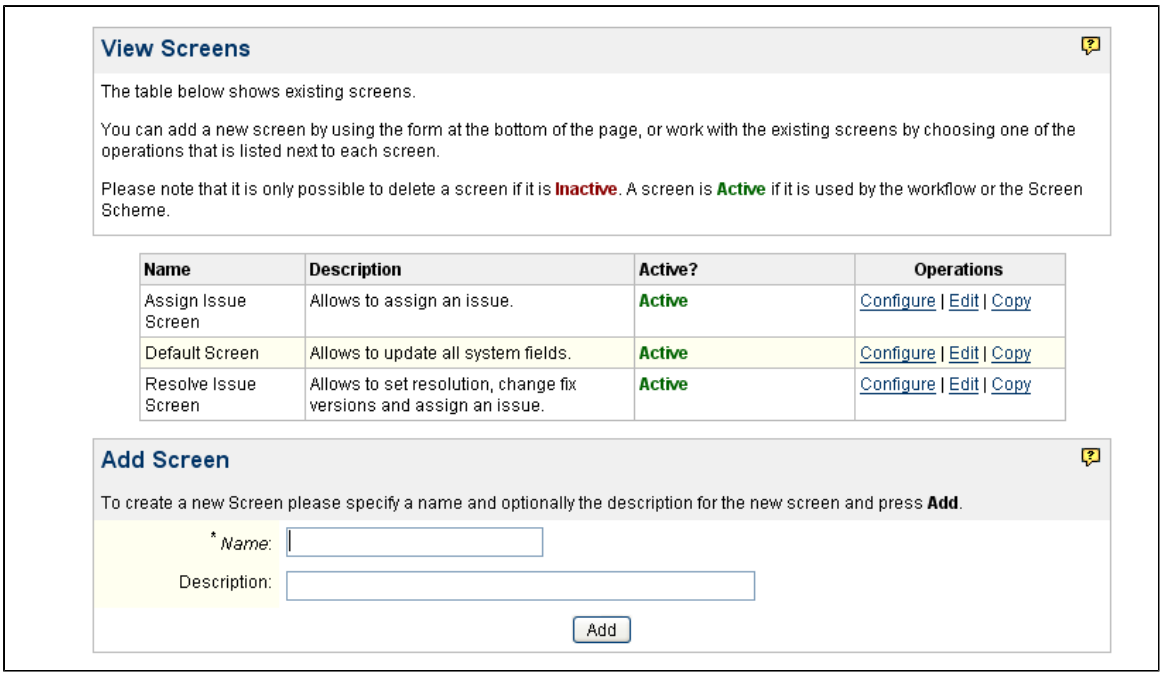
Further information on all these new features can be found in the JIRA 3.2 documentation. A full list of the new features and improvements in JIRA 3.2 can be found here.

Field Screens

JIRA 3.2 allows configuration of field position and visibility for each issue operation and in Professional and Enterprise editions for each workflow transition screen.

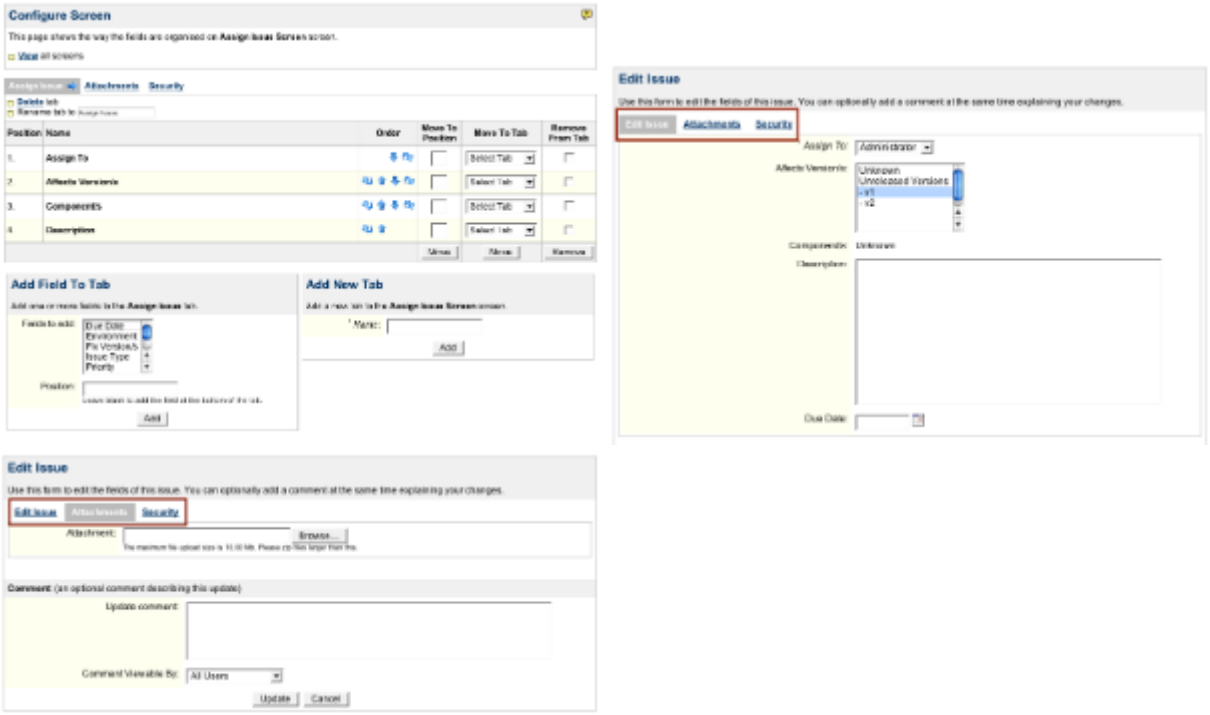
Each page on which you view and enter data for an issue is now a customisable "field screen". This provides for a flexible, more intuitive interface, with the ability to configure exactly which fields are presented for each operation. Each screen contains only those fields that directly relate to the operation being performed - while fields that do not relate to the operation can be hidden.

For example, it is possible to configure the *'Create Issue'* screen to display the *'Assignee'* field, while hiding this field in the *'Edit Issue'* screen. It is also possible to configure JIRA such that certain fields (including custom fields) appear only on specific workflow transitions: for example, if you have a *'QA Contact'* custom field, you can now configure JIRA to ensure that it is shown only on the *'Close Issue'* transition.



Field Screen Tabs

With JIRA Enterprise edition, you aren't even limited to a single field screen. Each page can have its fields divided amongst a group of tabs, creating a less cluttered environment as less used fields (e.g. attachments) can be placed in separate tabs.

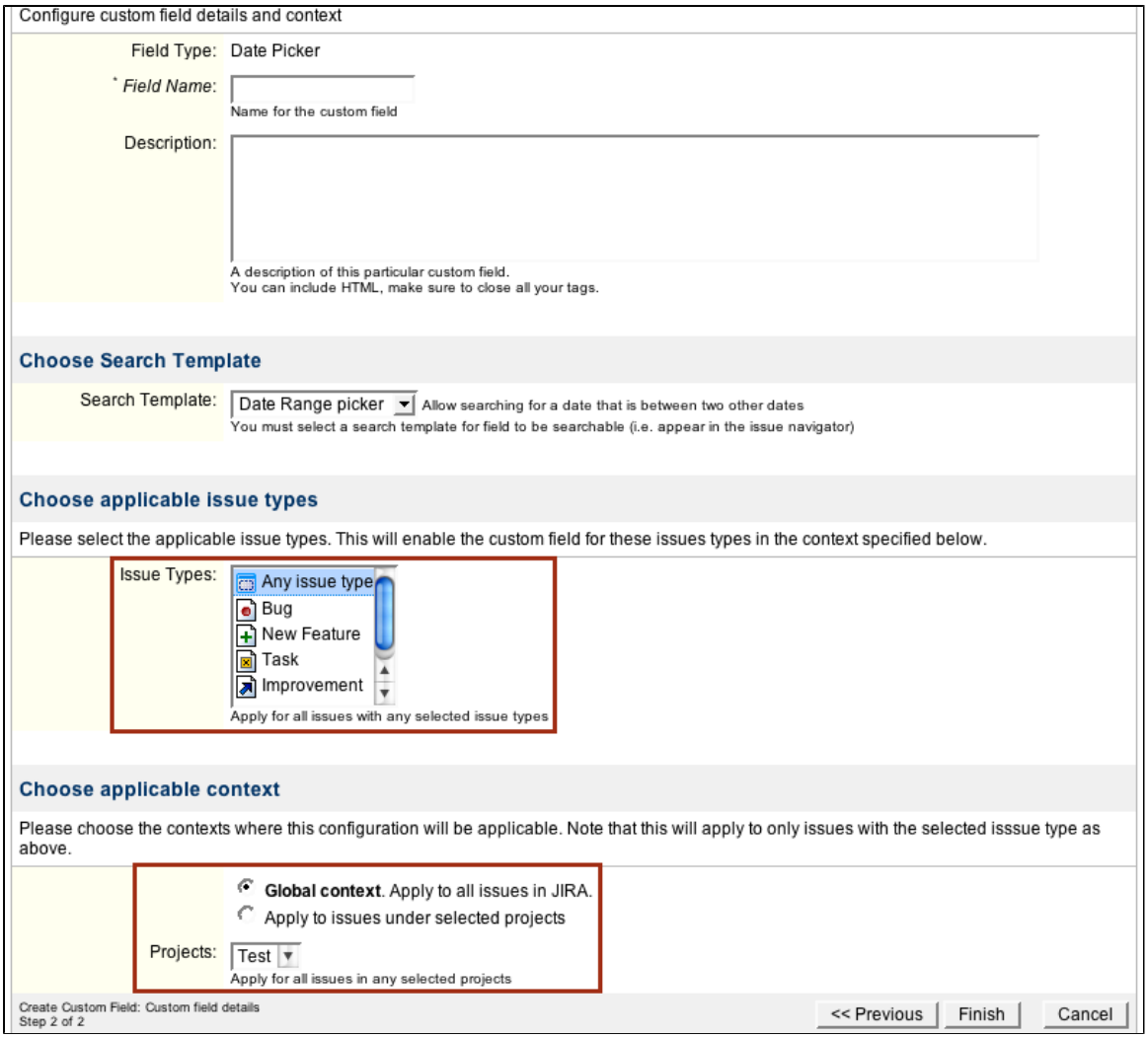


Contextual Custom Fields

Custom fields have been enhanced even further to allow greater flexibility and potential for customisation. Custom Fields are no longer limited to a single project or issue type, but can be shared between multiple issue types and multiple projects. Gone is the need to create the same custom field over and over again. Custom fields can be defined within a specified context or set of issue types, allowing them to be defined exactly where you need them.

Custom fields related to several projects and/or issue types can be configured to appear for those projects and/or issue types. For example, a custom field *'Customer ID'* could be associated with the *'Sales'* and *'Customer Info'* projects, while a custom field *'Operating System'* could be associated with the *'Bug'* and *'Improvement'* issue types in the *'Support'* project.

With configuration contexts, the same custom fields can still be configured differently for each project: for example having different default values.



Extended Bulk Edit

Further extending the bulk edit functionality, it is now possible to bulk edit more fields - a much requested feature. The list of fields that can be modified has been extended to include:

- Due Date
- Reporter
- Issue Security Level
- Issue Type

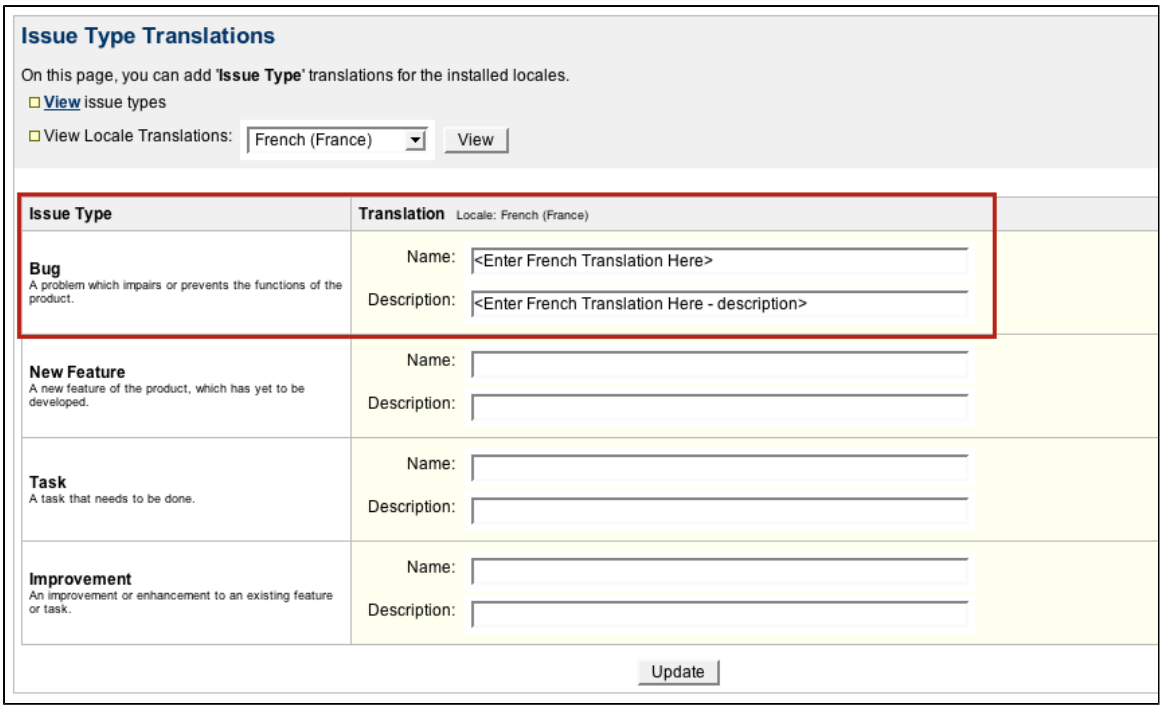
The bulk edit of assignee also works across multiple projects - allowing the selection of only valid assignees across all projects.

Internationalisation - Viva la JIRA!

Issue Constant Translations

Continuing our commitment to a fully internationalisable issue manager, JIRA 3.2 provides the ability to translate all issue settings - i.e. Priorities, Statuses, Issue Types and Resolutions. This allows for even more complete translations to present JIRA to each user in their own chosen language.

Hence, users who have selected the French locale will be presented with the French translations for the issue settings, while the Spanish users will be presented with the Spanish translations. The translations are presented throughout the JIRA interface - in issue creation/editing/viewing, reports, portlets, etc.



New Language Pack

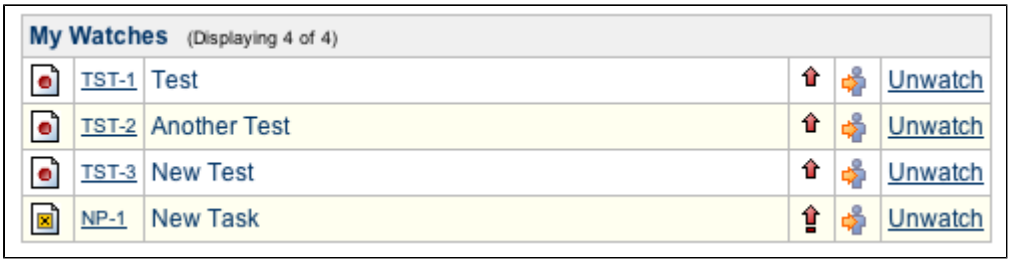
With the addition of Traditional Chinese, JIRA's user interface can be displayed in 12 languages.

Subversion - Multiple Repositories Support

The most popular JIRA Subversion plugin feature request has been addressed with support for multiple repositories now included in the latest plugin release. This allows users to map multiple JIRA projects to multiple Subversion repositories.

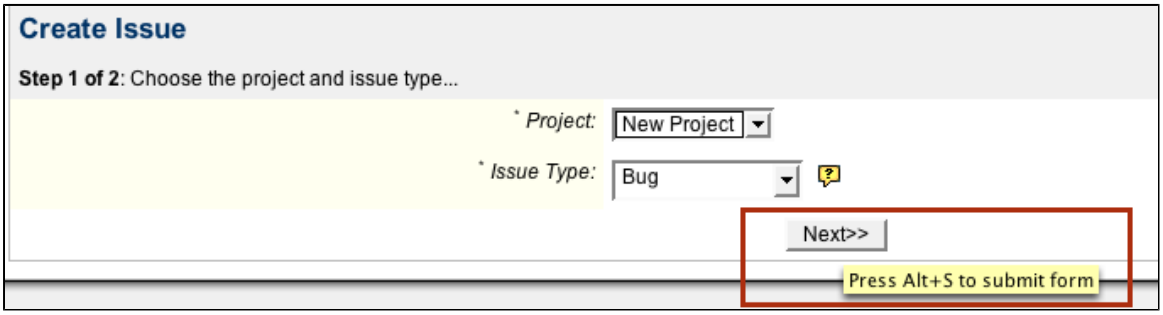
New Portlets

JIRA 3.2 introduces the **Voted Issues** and **Watched Issues** portlets - respectively showing the issues voted for and watched by the current user. These portlets can be added to the dashboard in the normal manner - providing quick access to the data you need.



Shortcuts

Navigating through JIRA is now quicker than ever with the addition of a number of shortcuts. Every form - from issue creation to editing a version - in JIRA can be submitted with the shortcut **ALT+S**. Form submission can be canceled using the shortcut **ALT+`** (ALT + backquote).



For more information on keyboard shortcuts please see our documentation.

Profile Email

In order to protect against spammers harvesting email addresses from JIRA, this release includes the functionality to hide the email addresses located on the user profile page. It is possible to configure the display of this information as follows:

- Show
- Hide
- Mask (e.g. *'user@example.com'* becomes *'user at example dot com'*)
- Show to logged in users only

Improvements

Performance - JIRA goes to boot camp!

One of the main goals of this release was to improve on overall performance of JIRA and there are major improvements in memory usage and speed. Utilising the Lucene search engine and some clever coding, some of the notable enhancements include:

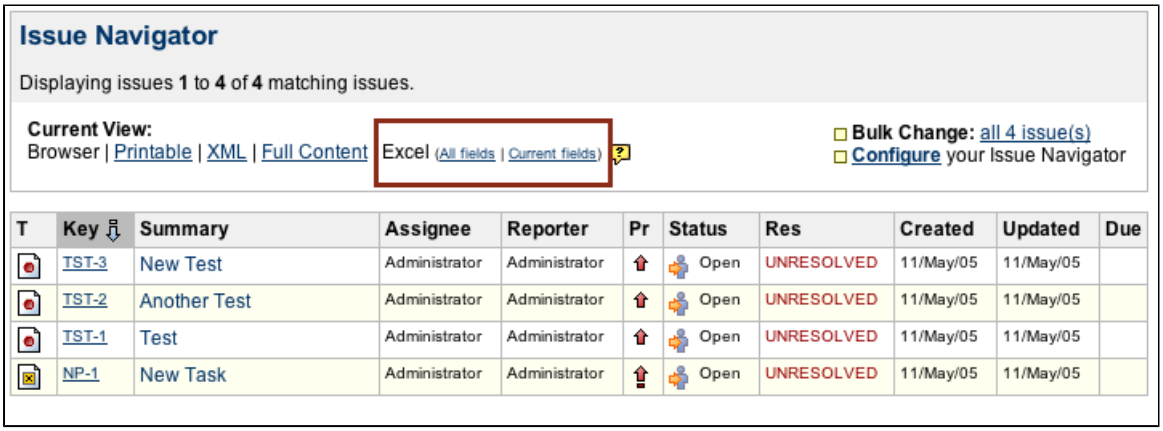
- Issue Navigator
  - **Much** quicker searching and Sorting (usually under 1s)
- Browse Project
  - Quicker **Project**, **Popular Issues** and **Open Issues** reports benefit with *up to **10x** improvement*
- Portlets
  - **Project** portlet is now ***2x** faster*
  - **Assigned to Me** portlet is *up to **10x** faster* and more efficient
- Importing and Indexing
  - Restoring JIRA backups or upgrading to a new version of JIRA is now ***2x** faster*.

Smart Query - let JIRA find it for you!

The 'Smart Query' functionality within JIRA has been further extended to provide easy access to commonly requested searches. For example, entering the query *'unresolved'* will direct the user to a list of unresolved issues. Further details on this feature can be found here.

EXCEL View

Many users requested that the EXCEL view display the same columns as displayed in the Issue Navigator view ... now it is possible to display *all* columns or just those visible in the the issue navigator.



Email Notification

Many users requested the ability to change the format of the ***From*** email address within JIRA notifications. The format is now fully configurable - with the ability to include or exclude the user name, email address and email hostname: for example, the format can be set to display as follows 'John Doe (ATLASSIAN) <jira@atlassian.com>'.

JDK 1.5 Compatibility

JIRA 3.2 is now fully compatible with JDK 1.5!

Bug Fixes

This release includes nearly 200 bug fixes - the best way to see them is them is to ask JIRA - JIRA 3.2 Bug Fixes.

Upgrading to JIRA 3.2

Please refer to the following documentation regarding the upgrade process to JIRA 3.2 from previous versions - JIRA 3.2 Upgrade Guide

JIRA 3.2 Upgrade Guide

This page contains information you need to know when upgrading to JIRA 3.2. The general upgrade instructions can be found here.

1. If you have written any Custom Field Type plugins please refer to this document
2. If you have created any Workflow plugins (custom Validators or Post Functions) please read this document.
3. If you have any custom file based workflows (workflows *not* created through JIRA's Workflow Editor) please read this document.
4. If you wish issues that are associated with the default system workflow and are closed to be bulk editable - please read this.

Notifications now respect permissions

In 3.2, JIRA respects the permission scheme and security levels when sending notifications (see JRA-5743. People who won't be able to see an update online won't get a notification email.

This has one important effect: if you have a project where:

- the notification scheme specifies that a raw email address (eg. developers@mycompany.com) should be notified, and
- 'Browse' permission has not been granted to 'Anyone' (eg. it is granted to 'jira-users'
  then that email address ('developers@mycompany.com' in our example) won't be mailed. As JIRA cannot verify that the recipient(s) of the email address have the 'browse' permission, it makes the conservative assumption that they are not.

This can be fixed by creating a user (eg. 'developers') for the email address, making it a member of a group that has 'Browse' permission, and adding it as a recipient of notifications. The raw email address should then be removed from the notification scheme, as it serves no purpose.

Notifications no longer sent to raw email addresses if anonymous browsing disabled

In 3.2, JIRA respects the permission scheme and security levels when sending notifications (see JRA-5743. People who won't be able to see an update online won't get a notification email.

This has one important effect: if you have a project where:

- the notification scheme specifies that a raw email address (eg. developers@mycompany.com) should be notified, and
- 'Browse' permission has not been granted to 'Anyone' (eg. it is granted to 'jira-users'
  then that email address ('developers@mycompany.com' in our example) won't be mailed. As JIRA cannot verify that the recipient(s) of the email address have the 'browse' permission, it makes the conservative assumption that they are not.

This can be fixed by creating a user (eg. 'developers') for the email address, making it a member of a group that has 'Browse' permission, and adding it as a recipient of notifications. The raw email address should then be removed from the notification scheme, as it serves no purpose.

Restricting Edit based on Issue Status

Restricting Edit based on Issue Status

JIRA 3.2 ensures that issues cannot be bulk edited if they cannot be edited normally due to their workflow status. The default workflow restricts issues in the *Closed* status from being edited.

Restricting Edits

A new property - **jira.issue.editable** - has been added that allows the administrator to specify which statuses/steps within a workflow are editable. The administrator can set the flag to be false for a given status - any issue assocaited with the workflow in this status will not be editable or bulk editable.

An example can be found in the default workflow - the **Closed** status has the property key set to false - restricting the ability to bulk edit any issue in this status.

In order to change this behaviour, it is necessary to copy and edit the default workflow and associate your issues with the new workflow.

Upgrading custom CustomFieldTypes in JIRA 3.2

Changes to the custom field types code

For JIRA 3.2, there has been incremental upgrades to the custom fields code. If you have written your own Custom Field Types, you'll want to read this document. We recommend that you ensure that your custom field types still compile against the latest 3.2 build and verify this under a testing environment before putting it into production. It is vital that at the time of the data upgrade, all of your custom field types are functioning correctly, or else they may not be upgraded correctly.

- CustomFieldType Interface changes
- Velocity template changes
- Abstract class changes
- CustomFieldType deal with CustomFieldConfig and not CustomField objects

CustomFieldType Interface changes

There has been some new methods added to the CustomFieldType interface. You may need to implement them if you didn't extend one of the existing concrete classes or AbstractCustomFieldType. The main change here is that FieldValidationException are now thrown instead of CustomFieldValidationException and the addition of two methods.

```
/**
     * Return the String value object from the CustomFieldParams. The object may be a single
String (e.g. TextCFType,
     * List of Strings (e.g. MultiSelectCFType) or CustomFieldParams of Strings (e.g.
CascadingSelectCFType)
     *
     * @param parameters - CustomFieldParams containing String values
     * @return String value object from the CustomFieldParams
     */
    public Object getStringValueFromCustomFieldParams(CustomFieldParams parameters);

    /**
     * Returns a List of {@link CustomFieldConfigItemType} objects.
     * This opens up possibilities for configurable custom fields
     *
     * @return List of {@link CustomFieldConfigItemType}
     */
    public List getConfigurationItemTypes();
```

Velocity Template Changes

The parameters passed to velocity templates for view and editing custom fields have also significantly changed. To update your velcoity templates, you should change your headers to be:

```
#controlHeader ($action $customField.id $customField.name $fieldLayoutItem.required
$displayParameters.noHeader)
...
#controlFooter ($action $fieldLayoutItem.fieldDescription $displayParameters.noHeader)
```

Abstract class changes

There has also been changes to the abstract CustomfFieldTypes with an addition of a GenericConfigManager dependency. You'll need to add this to all constructors of classes which extends them. You can use GenericConfigManager as an arbitrary data store.

CustomFieldType deal with CustomFieldConfig and not CustomField objects

Custom fields can now be configured differently per context. Thus the CustomFieldType now gets passed the CustomFieldConfig object (which is an instance of the configuration set) rather than just the CustomField themselves.

Upgrading Workflow Plugins for JIRA 3.2

If you have written a workflow plugin (Validator or Post Function) for JIRA 3.0 or 3.1 you might have to modify it to make it work for JIRA 3.2. If you are getting ClassCastExceptions while transitioning issues through workflow after upgrading to JIRA 3.2 there is a high chance that this document is what you are looking for.

Workflow Conditions still use GenericValues for issues, so if you have written a custom Workflow Condition it should work with JIRA 3.2

Regular Workflow Transition

For a regular workflow transition in JIRA 3.1 the `transientVars` map contained a GenericValue object that represented an issue. The `transientVars` Map also contained the fields that were changed during the transition (if any), namely Fix Versions, Assignee and Resolution.

In JIRA 3.2 an Issue object has been created, and all the changes that have been made to the issue can be obtained from the Issue object:

```
Issue issue = (Issue) transientVars.get("issue");
Map modifiedFields = issue.getModifiedFields();
```

Please note that in version of JIRA 3.1 and earlier the `transientVars` map contained a `GenericValue` object with the key "issue". If your code does something like:

```
GenericValue issue = (GenericValue) transientVars.get("issue");
```

This will cause a ClassCastException in JIRA 3.2. You need to cast the object to `Issue` instead of `GenericValue`. If you need to get the `GenericValue` of the issue, you can do that by calling:

```
GenericValue issueGV = issue.getGenericValue();
```

As mentioned earlier, the `modifiedFields` map contains all the fields that have been updated during the workflow transition. The keys of the `modifiedFields` map are ids of fields (please see `com.atlassian.jira.issue.IssueFieldConstants`) that have been modified, and the values of the `modifiedFields` map are `ModifiedValue` objects. A ModifiedValue object represents an updated field. The object stores the old and the new value of the field for the issue. You can use this object like so

```
Map modifiedFields = issue.getModifiedFields();
for (Iterator iterator = modifiedFields.keySet().iterator(); iterator.hasNext();)
{
    String fieldId = (String) iterator.next();
    ModifiedValue modifiedValue = (ModifiedValue) modifiedFields.get(fieldId);
    // Old value of the field
    Object oldValue = modifiedValue.getOldValue();
    // New Value of the field
    Object newValue = modifiedValue.getNewValue();

}
```

Please note, that the `comment` and `commentLevel` are still recorded in the `modifiedFields` map.

Initial Workflow Transition

An initial workflow transition is the transition that creates an issue.

In JIRA 3.1 and earlier the `transientVars` map contained the field values that should be used to create an issue. In JIRA 3.2 the value of the fields are recorded in the Issue object rather than in the `transientVars` map. You can use various getter methods to retrieve values of these fields. For example, if you need to get issue's description, instead of doing:

```
String description = (String) transientVars.get(IssueFieldConstants.DESCRIPTION);
```

do this:

```
Issue issue = (Issue) transientVars.get("issue");
String description = issue.getDescription()
```

Using Oracle 10g drivers to solve the 4000 character limitation

As you might be aware Oracle has a 4000 character limitation on VARCHAR2 fields. Which causes quite a few headaches when dealing with custom workflows or working with issues that have long descriptions, comments or custom field values.

Fortunately Oracle have worked around the VARCHAR2 limitation in their latest Oracle 10g JDBC driver. This fix (described online here) works with Oracle 9 and 10. We strongly recommend using Oracle 10g drivers and the setup described below if you are using Oracle 9i or 10g.

Even though Oracle suggests that Oracle 10g drivers work with Oracle 8i, users have reported problems with this configuration.

Please follow the Upgrading JIRA Safely instructions, keeping the following in mind:

Use Oracle 10g driver

This configuration will work **only** with Oracle 10g drivers. Therefore, from Oracle's site download the ojdbc14.jar (or applicable) JDBC driver, and copy it to your app server's lib/ directory (eg. common/lib/ for Tomcat). Remove the old JDBC jar used previously.

Create a new database for JIRA 3.2

Please create a **new** database for JIRA 3.2 and configure JIRA's data source to use it. Please do *not* point JIRA 3.2 at your old database.

Set the SetBigStringTryClob flag

When configuring the data source for the new database set the SetBigStringTryClob flag to true. The way this must be done depends on your application server

JIRA Standalone, Tomcat 4 and 5.0:

Edit conf/server.xml (Tomcat 4) or conf/Catalina/localhost/jira.xml (Tomcat 5.0), locate the section where the 'jdbc/JiraDS' DataSource is set up, and add:

```
<parameter>
  <name>connectionProperties</name>
  <value>SetBigStringTryClob=true</value>
</parameter>
```

For instance, in JIRA Standalone one would then have:

```
<Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"/>
    <ResourceParams name="jdbc/JiraDS">
      <parameter>
        <name>driverClassName</name>
        <value>oracle.jdbc.driver.OracleDriver</value>
      </parameter>
      <parameter>
        <name>url</name>
        <value>jdbc:oracle:thin:@<database host machine>:<port>:<SID></value>
      </parameter>
      <parameter>
        <name>username</name>
        <value>...</value>
      </parameter>
      <parameter>
        <name>password</name>
        <value>...</value>
      </parameter>

<parameter>
  <name>connectionProperties</name>
  <value>SetBigStringTryClob=true</value>
</parameter>

      <parameter>
        <name>factory</name>
        <value>org.apache.commons.dbcp.BasicDataSourceFactory</value>
      </parameter>
    </ResourceParams>
```

Tomcat 5.5

In Tomcat 5.5, the format for the added section would be:

```
<Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"
    username="jirauser"
    password="jirauser"
    driverClassName="oracle.jdbc.driver.OracleDriver"
    url="jdbc:oracle:thin:@localhost:1521:jiradb"

connectionProperties="SetBigStringTryClob=true">

</Resource>
```

Orion / OC4J

For Orion/OC4J, edit config/data-sources.xml, and add the property as a nested tag:

```
<data-source
    class="<datasource driver class>"
    name="<name>"
    location="<location>"
    xa-location="<xa-location>"
    ejb-location="<ejb-location>"
    url="<url>"
    connection-driver="<driver>"
    username="<login>"
    password="<password>"
    inactivity-timeout="30"
>
  <property name="SetBigStringTryClob" value="true" />
</data-source>
```

Other app servers

Consult the relevant JIRA app server guide and the app server documentation to find how to add the property.

Use oracle10g field-type-name

Please specify **oracle10g** (*not* oracle) as the `field-type-name` when editing `WEB-INF/classes/entityengine.xml`

After the data is re-imported and JIRA upgrades the data the 4000 character limitation should disappear.

Workflows using default "Closed" status

Prior to JIRA 3.2, workflow steps using the default "Closed" JIRA status had the special behaviour that disabled certain operations such as "Edit". Since 3.2, this behaviour has been made available to **all** workflow steps, regardless to status. You can disable issue editing on a workflow step by adding a meta attribute "jira.issue.editable=false" to the relevant steps.

For backwards compatibility, all workflow steps with the "Closed" step that is stored in the database have been upgraded to have this new meta attribute. So for the majority of users, JIRA will have already done the upgrading for you. For those with workflows that are stored in the file system, you will have upgrade manually (if you want the closed status to disallow edits). Instructions are below:

Upgrading file based workflows

You only need to do this upgrade if you have created workflows as XML documents on disk and if you want the step to disallow edits. For most JIRA installations workflows are created using the workflow editor. These workflows will be upgraded automatically.

First, find the workflow step with the status id of 6 (closed)

```
<meta name="jira.status.id">6</meta>
```

Simply add the following to the step:

```
<meta name="jira.issue.editable">false</meta>
```

Restart JIRA and all the new changes should take effect.

3.2 performance benchmarks

Here are some quick benchmarks to illustrate the performance improvements in JIRA 3.2 vs. 3.1.1.

Return to the 3.2 release notes

Sample JIRA data details

Issues	14862
Comments/changes	38294
Users	9163 (most inactive)

Performance comparison

Operation	3.1.1	3.2
Full reindex	429s	287s
Search	37s	1.1s

Search returning first 25 of 14,862 issues, average over multiple runs.

System

Software: JIRA 3.1.1 Enterprise; all issues cached.
System: 2.6Ghz Pentium 4
Database: MySQL 4.1.9

JIRA 3.2.3 Release Notes

JIRA 3.2.3 Release Notes

In the tradition of frequent and worthwhile updates, JIRA 3.2.3 is released today in Standard, Professional and Enterprise editions. This point release includes 16 bug fixes and improvements. It can be downloaded here. See the JIRA 3.2 Upgrade Guide before upgrading.

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.2.3 includes 16 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.2.2 Release Notes

JIRA 3.2.2 Release Notes

In the tradition of frequent and worthwhile updates, JIRA 3.2.2 is released today in Standard, Professional and Enterprise editions. This point release includes over 30 bug fixes and improvements. It can be downloaded here. See the JIRA 3.2 Upgrade Guide before upgrading.

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

JIRA 3.2.2 includes over 30 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.2.1 Release Notes

JIRA 3.2.1 Release Notes

In the tradition of frequent and worthwhile updates, JIRA 3.2.1 is released today in Standard, Professional and Enterprise editions. This point release includes over 50 bug fixes and improvements. It can be downloaded here. See the JIRA 3.2 Upgrade Guide before upgrading.

JIRA 3.2.1 includes over 50 bug fixes and improvements.

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.0 Release Notes

Atlassian is proud to present the next major evolution in JIRA - JIRA 3.0! These are the release notes for the final release of JIRA 3.0. There are addendum pages for 3.0.1, 3.0.2 and 3.0.3.

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Existing customers who wish to upgrade, or new users who wish to try out JIRA 3.0 for 30 days can download either the standalone or WAR distributions from the Atlassian website.

What's new in JIRA 3.0?

JIRA 3.0 is the second major JIRA overhaul (after 2.x), and the largest release in Atlassian's short history. As always, all existing customers who are still within the 12 months maintenance period can upgrade for free, thanks to the provision for one year of free upgrades in your license.

You will find JIRA 3.0 contains many significant new features as well as the many, many enhancements, bug fixes and things we just couldn't resist throwing in at the last minute. By our calculations, over 400 issues were collectively resolved for JIRA 3.0!

New JIRA editions

Previously, JIRA came in 2 flavours - Professional and Enterprise. JIRA 3.0 marks the introduction of a new family member, JIRA Standard edition, which fits below the Professional edition. For more details on what features are in which editions - see JIRA 3.0 Feature Comparison.

Note: all existing JIRA Professional licenses who are still within the 12 months maintenance period are able to upgrade to JIRA 3.0 Professional at no cost - so for example if you are a JIRA 2.x Professional customer, you can upgrade to JIRA 3.0 Professional and will get all the Professional features.

Upgrading from previous versions

Upgrading JIRA should be pretty easy, see the JIRA 3.0 Upgrade Notes, and the guide to upgrading JIRA.

Documentation

The JIRA 3.0 documentation is online here.

Contents

1. New Features
2. Improvements

New Features

JIRA 3.0 contains a lot of major new features - more than in any other single release of JIRA. Let's look at a few of the most important ones:

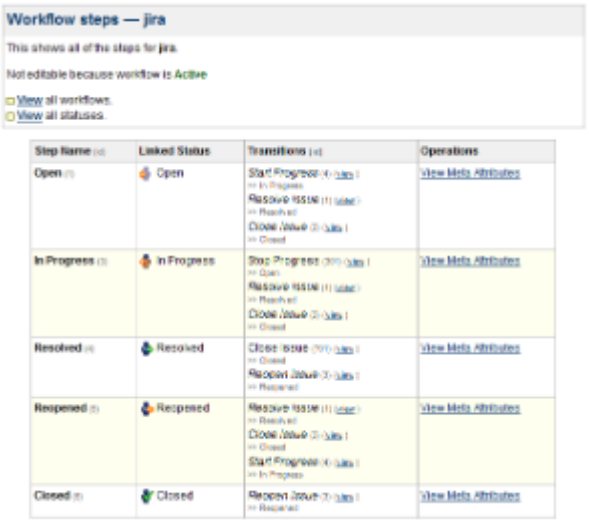
Workflow Editor & Configurable Workflow

The most anticipated new features in JIRA 3.0 are the workflow editing features. JIRA now contains a workflow editor (see screenshot) that allows you to view and edit workflows through the web interface, in the Professional and Enterprise editions. The workflow controls the set of steps which an issue moves through in order to move to a resolution.

Editor features:

- Workflows can be created with arbitrary steps (linked to statuses) and transitions between them.
- Statuses can be added and removed through the web interface.
- Conditions, functions, validators are now JIRA plugins (see below for details of the new plugin system):
    - Conditions allow you to govern when transitions can be executed and by whom (for example the transition from "Open" to "Closed" can only be executed by a user with the "Close Issue" permission).
    - Functions control exactly what happens after a workflow transition is executed (for example an event is fired, an issue field

is changed or an email is sent).
- Validators ensure that the data provided for a transition is correct.



Multiple Workflows - Enterprise users can configure multiple workflows for their JIRA installation, assigning a default workflow to each project, and then overriding that workflow for particular issue types if required (via a new workflow scheme). This powerful feature allows an issue to progress through a uniquely tailored workflow - one specifically designed for the life-cycle of that issue.

Sub-tasks

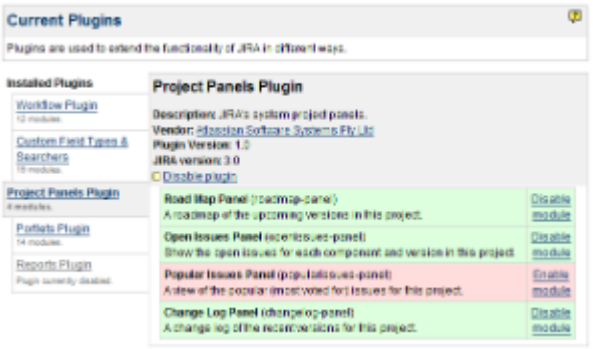
Issues are like people, they come in all shapes and sizes - large and small. Sometimes a given issue will be solved in multiple 'stages', often by different people. This is the genesis behind the sub-tasks feature.

Sub-tasks allow you to break an issue up into multiple tasks, each of which is a full issue in itself (with its own individual workflows). As you can see from the screenshot here, sub-tasks are very rapid to create and manipulate almost never requiring you to leave the View Issue screen.

Unlike other issues, sub-tasks also have a defined order and the issue contains an indicator of overall sub-task progress (see arrows and red/green indicators on screenshot).

As with all issue types, Enterprise users can override the default project workflow association and specify a custom workflow for each sub-task type.

Plugin System



JIRA now has a plugin system. This means that developers can build plugins, which extend the functionality of JIRA in different ways, plugged in at multiple points within the interface.

Detailed information for developers interested in building their own plugins will be coming soon, but as a summary: each plugin is one or more classes and a simple XML plugin descriptor. Often plugins will also contain Velocity templates to render portions of the UI as necessary. A brief guide to JIRA plugins is available here.

Each plugin consists of one or more plugin modules, each of a specific type (see below). Both plugins and individual modules can be disabled or enabled through the administration interface (see screenshot).

The 12 current module types shipping with JIRA 3.0 are:

- Reports - define a report with the information collected to run it and the resulting HTML.
- Portlets - define portlets and the parameters they accept when configured.
- Custom Field Types - define new types of custom fields (for example a "User Picker") including their rendering templates.
- Custom Field Searchers - provide index and search capabilities to custom field types.
- Issue Tab Panels - add new panels to the View Issue screen (like existing "Comments" and "Version Control" panels).
- Project Tab Panels - add new panels to the Browse Project screen (like existing "Roadmap" and "Popular Issues" panels).
- SOAP RPC Endpoints - define new SOAP web services for JIRA.
- XML-RPC Endpoints - define new XML-RPC web services for JIRA.
- Components - implement new components within JIRA (or override existing components) that are provided to other plugin modules (via PicoContainer).
- Workflow Conditions / Functions / Validators - define extra conditions, functions and validators to be used in your JIRA workflow.

Upcoming Plugins

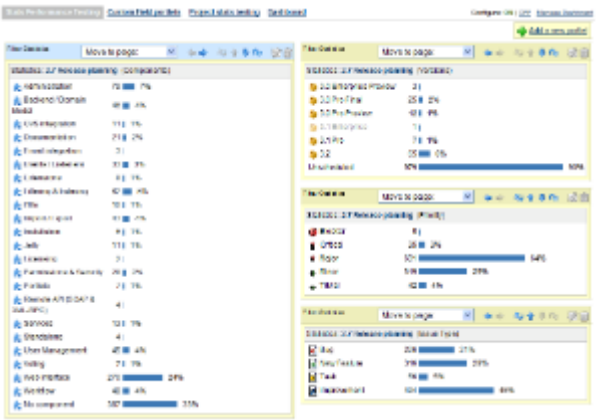
The popular requests for Subversion and Perforce support within JIRA will be addressed with the forthcoming release of two new plugins. While providing support for these extensions to JIRA, they are also great examples of creating plugins within the new plugin system. These plugins are due to be released shortly.

Dashboard Overhaul

The Dashboard has long been one of the best, most loved features in JIRA. With JIRA 3, it gets a whole lot better!

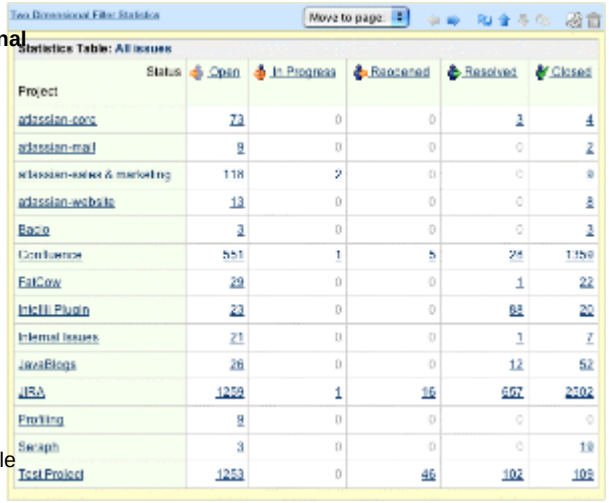
Inline Editing - the usability of the Dashboard has been greatly improved with the new inline editing mode. This allows you to move, add, delete and edit your portlets from the Dashboard itself, instead of having to switch back and forth to the Dashboard configuration screen (see screenshot).

Pluggable Portlets - portlets within JIRA have now moved within the plugin system as well, enabling easy creation and sharing of portlets with other users.



Filter Statistics Portlet - a new, very useful portlet has been added which allows you to calculate statistics for any saved filter. For example, you can create a filter showing the currently open issues assigned to yourself for a particular project - and then put a chart of those issues onto your Dashboard, broken down by component.

Two Dimensional Filter Statistics Portlet - building on the Filter Statistics portlet, this portlet allows you to display statistics from a filter in a configurable table. The X and Y axis of the table are configurable - for example, issue type against priority - and provides a powerful tool for manipulating the display of the statistics.



Multiple Dashboard Pages - And, to save the best for last, you can now have multiple dashboard pages in the Professional and Enterprise editions. This allows you to create multiple, named Dashboard pages which show up as different tabs on screen (see top right hand side of screenshot).

This is extremely useful in a number of scenarios. Often we've noticed that users' Dashboard pages become extremely long (up to 10 screens!) due to adding more statistic portlets - multiple dashboards allows you to split up these long pages. If you work with multiple projects it is fantastic to be able to create a Dashboard per project, each with relevant statistics providing deeper insight into the project.

Custom Fields

Custom Field Types - The custom field system has been completely overhauled for JIRA 3.0 (thanks to Joseph Dane of Hawaii University for the initial impetus!). Custom field types are now JIRA plugins, so new custom field types can be easily created, installed and shared by users. Custom fields themselves can now support multiple values, multiple search methods and can have configurable view / edit / search templates.

For example, imagine you wished to connect certain internal users within your organisation with a particular field in your issue. You could create an 'internal user' custom field type edited with a simple drop down select box of user's full names pulled from a remote web service. To display this custom field, you could have a completely different template which actually included a picture of that user from your intranet!

New System Types - Using this custom field system, we've created a number of new custom field types for JIRA 3.0. The existing custom fields - text field, free text, select list, date picker and number field - all still exist, but now they're joined by the following new types out of the box:

- User Picker - choose a user from the user base via a popup picker window.
- Project Picker - choose from projects that the user can view in the system.
- Cascading Select - choose multiple values using two select lists.
- Multi Checkboxes - choose multiple values using checkboxes.
- Multi Select - choose multiple values in a select list.
- Radio Buttons - a list of radio buttons.
- URL Field - allow the user to input a single URL.
- Read-only Text Field - a read-only text label (only possible to create values programatically).
- Version Picker - choose from available versions in the project.

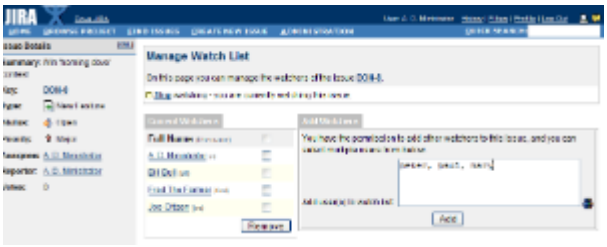
Voting and Watching

The most voted for JIRA feature has arrived! Users with the "Edit Watcher List" permission can now update and remove watchers. In this way, other users can be added as watchers of a particular issue (for example, I may want to make sure that Bob is notified of changes to a given issue).

For visibility, there is also a new governing "View Watchers and Voters" permission. Users with this permission can see the list of people who have voted on or are watching an issue.

The number of votes an issue has can now also be added to the issue navigator.

Other New Features



- **Change Reporter** - the second most requested JIRA feature has also arrived! Users with the "Modify Reporter" permission can now update the reporter of an issue.
- **Clone Issue** - you can now duplicate existing issues, optionally linked to the original issue. The sub-tasks of an issue are also cloned if any exist.
- **Multiple Attachments** - you can now upload multiple attachments at the same time if you want
- **Attachments while Creating** - attachments can also be added while creating the issue in a single step, rather than the existing two step 'create issue', 'attach file' process.
- **XML issue view** - each issue can now be viewed or downloaded as XML, including its comments and custom fields.

Improvements

There were many, many improvements made in JIRA 3.0, here are the major items of note:

- **Version management** - version management has been overhauled and versions now have a due date.
- **Issue Navigator** - sorting of columns has been extended with the ability to sort by Versions, Fix for Versions and Components (sorting on the earliest of each found).
- **CVS integration** - improved hugely in JIRA 3.0. JIRA can now access CVS repositories via SSH and the local file system (in addition to the existing pserver support), as well as display branch information for commits. The performance has also been worked on a lot, such that CVS access now requires a near constant amount of memory regardless of repository size.
- **More languages** - JIRA has now been translated to German, Spanish, Danish, Russian, French and Brazilian Portuguese.
- **Issue linking** - you can now link multiple issues at a time, as well as use search filters to find issues to link.
- **Project keys** - there is a configurable regular expression to govern the project key structure within JIRA.
- **SSO support** - JIRA integrates with a number of existing single sign-on frameworks, and can easily be customised to work with custom SSO systems.
- **Remote API** - the SOAP and XML-RPC remote APIs are distributed as a plugin, which can be updated independently of the main JIRA installation.
- **Remote Issue Creation** - issues can now be created and retrieved via the remote APIs.
- **Field Layouts** - Enterprise users can now define field layout schemes for each issue type within a project.
- **Reports** - there is now a version workload report (thanks to the JetBrains developers for this contribution) which shows the estimated times against developers for a given version.
- **Email notifications** - Improvements have been made to the format of notifications sent out by JIRA, especially text emails.
- **Startup time** - JIRA should now start up *much* faster than it did previously!
- **User interface** - As always, we've worked on improving the user interface in various areas (most notably the browse projects screen and the version administration screen).

and much, much more...

JIRA 3.0 Upgrade Notes

This page lists a few things to be aware of when upgrading from previous releases of JIRA to JIRA 3. To perform the actual upgrade, see the upgrade documentation.

Existing SMTP Mail Server 'From' address may break notifications (JRA-5089)

In JIRA 3, email notification 'From' addresses now contain the reporter name, eg. "Joe Bloggs (JIRA) <jira@company.com>", where "jira@company.com" is set by the admin as the SMTP mail server From address. If you have this address to already include a name (eg "Tech Support <jira@company.com>", then email notifications will fail with errors like:

```
2005-01-06 11:30:53,856 ERROR [atlassian.mail.queue.MailQueueImpl]
com.atlassian.mail.MailException: Sending failed;
  nested exception is: javax.mail.internet.AddressException:
Missing '<' in string ``"Joe Bloggs (JIRA)" <Tech Support <jira@company.com>>'' at position 62
```

Fix

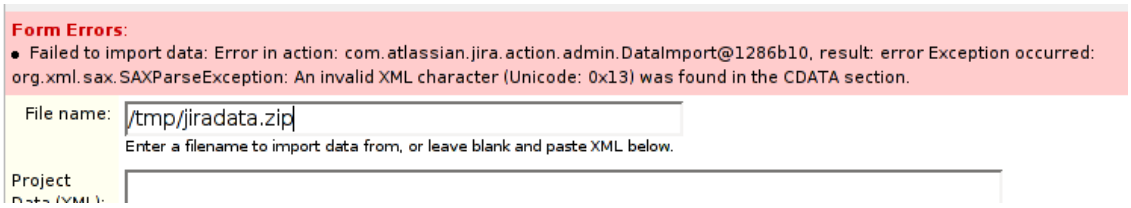
The fix is to edit `WEB-INF/classes/jira-application.properties`, and change the following property value to `false`:

```
jira.option.include.user.in.mail.from.address = true
```

- If using JIRA Standalone, the file is *atlassian-jira/WEB-INF/classes/jira-application.properties*, after which you should run bin/shutdown and bin/startup to restart.
- If using JIRA deployed as a webapp, copy *webapp/WEB-INF/classes/jira-application.properties* to *edit-webapp/WEB-IFN/classes*, make the change to the *edit-webapp* copy, run *build* to rebuild the webapp, and redeploy it on your app server.

Invalid characters break XML import

JIRA's recommended upgrade process involves deploying an XML backup of your data. Some users will find that the import fails with this error:



This is usually because the database contains control characters that cannot be represented in Unicode, and hence XML.

Fix

The fix is to follow these instructions to remove the invalid characters from the XML before import.

JIRA 3.0.3 Release Notes

JIRA 3.0.3 is a bugfix release. For the full list of changes from 2.x releases, see the JIRA 3.0 Release Notes. The JIRA 3.0 Upgrade Notes apply.

Changes since 3.0.2:

Bugs fixed:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Improvements:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.0.2 Release Notes

JIRA 3.0.2 is a bugfix release, mainly to address a performance problem with the 'find issues' page when there are large numbers of projects. For the full list of changes from 2.x releases, see the JIRA 3.0 Release Notes. The JIRA 3.0 Upgrade Notes apply.

Changes since 3.0.1:

Bugs fixed:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Improvements:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.0.1 Release Notes

JIRA 3.0.1 is a minor bugfix release. For the full list of changes from 2.x releases, see the JIRA 3.0 Release Notes. The JIRA 3.0 Upgrade Notes apply.

Issues addressed since 3.0

Bugs fixed:

JIRA 3.0.1 fixes one bug, which prevented users importing backup data into an empty JIRA instance:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

See also JIRA 3.0.2 Release Notes

All JIRA Release Notes (version 3.x and later)

This page lists the release notes from JIRA versions 3.x and later.

You may be interested in the list of upgrade guides from JIRA versions 3.x and later.

No content found for label(s) JIRAreleasenotes.

All JIRA Upgrade Guides (version 3.x and later)

This page lists the upgrade guides from JIRA versions 3.x and later. You can also view the aggregated upgrade guides from JIRA versions 3.x and later (***warning:*** *the aggregated upgrade guides page is big!*).

If upgrading from a previous version of JIRA please pay attention to the *Upgrade Guide* of the version you are upgrading to, and any version

of JIRA that you are 'skipping' during the upgrade.

You may be interested in the list of release notes from JIRA versions 3.x and later.

No content found for label(s) JIRAupgradeguide.

Aggregated JIRA 3.x Upgrade Guides

This page contains a live aggregate of all JIRA upgrade guides since version 3. You can also view the lists of Release Notes or Upgrade Guides for JIRA.

JIRA 2.x to 3

This page lists a few things to be aware of when upgrading from previous releases of JIRA to JIRA 3. To perform the actual upgrade, see the upgrade documentation.

Existing SMTP Mail Server 'From' address may break notifications (JRA-5089)

In JIRA 3, email notification 'From' addresses now contain the reporter name, eg. "Joe Bloggs (JIRA) <jira@company.com>", where "jira@company.com" is set by the admin as the SMTP mail server From address. If you have this address to already include a name (eg "Tech Support <jira@company.com>", then email notifications will fail with errors like:

```
2005-01-06 11:30:53,856 ERROR [atlassian.mail.queue.MailQueueImpl]
com.atlassian.mail.MailException: Sending failed;
  nested exception is: javax.mail.internet.AddressException:
Missing '<' in string ``"Joe Bloggs (JIRA)" <Tech Support <jira@company.com>>'' at
position 62
```

Fix

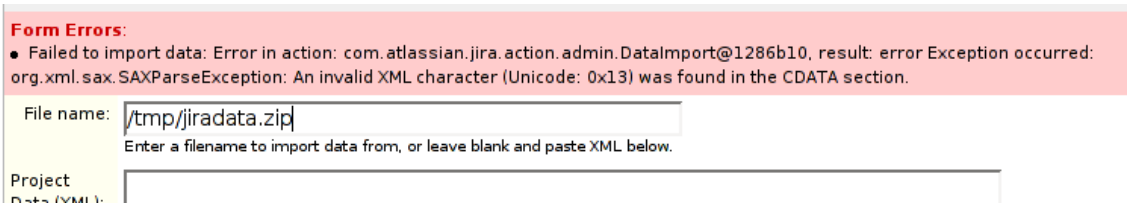
The fix is to edit `WEB-INF/classes/jira-application.properties`, and change the following property value to `false`:

```
jira.option.include.user.in.mail.from.address = true
```

- If using JIRA Standalone, the file is *atlassian-jira/WEB-INF/classes/jira-application.properties*, after which you should run bin/shutdown and bin/startup to restart.
- If using JIRA deployed as a webapp, copy *webapp/WEB-INF/classes/jira-application.properties* to *edit-webapp/WEB-IFN/classes*, make the change to the *edit-webapp* copy, run *build* to rebuild the webapp, and redeploy it on your app server.

Invalid characters break XML import

JIRA's recommended upgrade process involves deploying an XML backup of your data. Some users will find that the import fails with this error:



This is usually because the database contains control characters that cannot be represented in Unicode, and hence XML.

Fix

The fix is to follow these instructions to remove the invalid characters from the XML before import.

JIRA 3.0 to 3.1

This page lists a few things to be aware of when upgrading from JIRA 3.0.x to JIRA 3.1. To perform the actual upgrade, see the upgrade documentation. For upgrading from JIRA 2.x to JIRA 3.x see JIRA 3.0 Upgrade Notes

MySQL Users dB upgrade (JRA-5635)

The size of the descriptor field in the jiraworkflow table has been increased. MySQL users will see warnings when they start their app server. This can be fixed by running the SQL below. This will also allow for Workflows of up to 4GB as opposed to just 64k

```
alter table jiraworkflows change DESCRIPTOR DESCRIPTOR LONGTEXT;
```

JIRA 3.1 to 3.2

This page contains information you need to know when upgrading to JIRA 3.2. The general upgrade instructions can be found here.

1. If you have written any Custom Field Type plugins please refer to this document
2. If you have created any Workflow plugins (custom Validators or Post Functions) please read this document.
3. If you have any custom file based workflows (workflows *not* created through JIRA's Workflow Editor) please read this document.
4. If you wish issues that are associated with the default system workflow and are closed to be bulk editable - please read this.

Notifications now respect permissions

In 3.2, JIRA respects the permission scheme and security levels when sending notifications (see JRA-5743. People who won't be able to see an update online won't get a notification email.

This has one important effect: if you have a project where:

- the notification scheme specifies that a raw email address (eg. developers@mycompany.com) should be notified, and
- 'Browse' permission has not been granted to 'Anyone' (eg. it is granted to 'jira-users'
  then that email address ('developers@mycompany.com' in our example) won't be mailed. As JIRA cannot verify that the recipient(s) of the email address have the 'browse' permission, it makes the conservative assumption that they are not.

This can be fixed by creating a user (eg. 'developers') for the email address, making it a member of a group that has 'Browse' permission, and adding it as a recipient of notifications. The raw email address should then be removed from the notification scheme, as it serves no purpose.

JIRA 3.2 to 3.3

JIRA 3.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.3 from JIRA 3.2.x. If upgrading from an older version of JIRA, please go to the complete list of Upgrade Guides, and read the notes for each version you are skipping during the upgrade.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Known incompatibilities

3.3.x is not a good release for IBM shops:

1. JIRA 3.3.x may not work on **Websphere 5.0.x and 5.1.x** due to JRA-7699
2. When using DB2, JIRA may hang when deleting projects or performing workflow operations. See the full problem description (and possible workaround) in the documentation

Websphere or DB2 users, please stick with 3.2.x or move on to 3.4.x or higher, where these problems have been resolved.

Notes on upgrading

1. Due to web browser caches, changes to JIRA's Issue Navigator might appear corrupted or unstyled. Please refresh your browser's cache (press *Shift+Reload* on the Find Issue's page) for the changes to appear correctly.
2. JIRA's **issue cache** size will be automatically set to 0 during the upgrade, as it is no longer needed due to performance improvements in JIRA (JRA-7166)
3. If you have written any CustomFieldType or CustomFieldSearcher plugins please refer to this document
4. Users with **outgoing** trackback pings enabled (not the default) may wish to disable this until JRA-7589 is fixed, to avoid the risk of the mail queue hanging.
5. If you have bookmarks or deal with hard coded links to the issue navigator, you should read about the changed issue navigator parameters
6. If you are **using JIRA Standalone**, please do *not* simply copy your old `conf/server.xml` file to the new installation of JIRA. Please read this document.
7. If upgrading JIRA in an external Tomcat installation, be sure to delete the `work/` temporary directory before restarting JIRA, to clear cached JSPs from the old JIRA.

JIRA 3.3 to 3.3.x

JIRA 3.3.1 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.3.1 from JIRA 3.3.

> If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below:

1. If you have implemented a cutom Issue Tab Panel plugin you need to be aware of this API change.

> If you are upgrading to JIRA 3.3.1 from a previous version, due to web browser caches, changes to JIRA's Issue Navigator might appear corrupted or unstyled. Please refresh your browser's cache (press Shift+Reload on the Find Issue's page) for the changes to appear correctly.

JIRA 3.3.x to 3.4.x

JIRA 3.4 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.4 from JIRA 3.3.3. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

> Two major new features of JIRA 3.4, wiki renderer previews, and issue types per project require that javascript be enabled to make use of their full functionality. You will still be able to use all the core features of JIRA with javascript disabled.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

1. Please do **not** copy `jira-application.properties` file from your old JIRA installation. Edit the file that is shipped with JIRA 3.4 and make needed changes. New properties have been added to this file so if you simply copy the old file across the following error would occur JRA-8645.
2. If you have written any CustomFieldType or CustomFieldSearcher plugins please refer to Upgrading Custom Field Types in JIRA 3.4
3. The default user preferences are now configured in the jira-application.properties file and are configurable through the admin section of JIRA. Any properties in the old file preferences-default.xml will no longer effect JIRA configuration.
4. Please note that to configure issue types per project you must have JavaScript turned on in your web browser.
5. If you are using MySQL please do not use Connector/J 3.1.11 JDBC Driver as it has the following bug. Connector/J 3.1.10 and earlier work fine.

JIRA 3.4.1 Upgrade Guide

This section contains specific information you need to know when upgrading to JIRA 3.4.1 from JIRA 3.4. If upgrading from JIRA 3.3.3 please read the previous section as well. If upgrading from an older version than JIRA 3.3.3, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

1. Please do **not** copy `jira-application.properties` file from your old JIRA installation. Edit the file that is shipped with JIRA 3.4 and make needed changes. New properties have been added to this file so if you simply copy the old file across the following error would occur JRA-8645.
2. If you have written a CustomFieldType that implements the `com.atlassian.jira.issue.customfields.CustomFieldType` interface directly rather than extending one of the Abstract classes that ship with JIRA please read Upgrading Custom Field Types in JIRA 3.4.1.
3. If you have written an Custom Field Searcher please have a look at Upgrading Custom Field Types in JIRA 3.4.1.
4. JIRA 3.4 and 3.4.1 do not generate an Issue Assigned event. The Issue Updated event is generated instead. In previous versions of JIRA the Issue Assigned event was generated when issues are assigned using the "Assign" operation on the View Issue page. This means that even when the "Assign" operation is used JIRA will send notifications to parties listed under the Issue Updated event. The patch to correct this behaviour is available at JRA-8533.

JIRA 3.4.2 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.4.2 from JIRA 3.4.1. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.4.1 to JIRA 3.4.2.

JIRA 3.4.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.4.3 from JIRA 3.4.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.4.2 to JIRA 3.4.3.

JIRA 3.4.x to 3.5.x

JIRA 3.5 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5 (release notes) from JIRA 3.4.3. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

JIRA 3.5 Jira Service extension

- If you have implemented a custom JIRA service you need to be aware of the following API change.

In JIRA 3.5 the `getName()` and `setName(String name)` methods was added to the `com.atlassian.jira.service.JiraService` interface. This method should return and set the name of the service respectively. The name of the service can be used to identify a service uniquely. (Fixed made due to JRA-8352 bug)

Therefore, if you have implemented this interface, you will need to implement these methods and recompile your service(s) before deploying it into JIRA 3.5. If you have extended a JIRA class instead, e.g. `com.atlassian.jira.service.AbstractService` or `com.atlassian.jira.service.JiraServiceContainer` you do not need to modify your custom services.

Introduction of global Bulk Change permission

JIRA 3.5 introduces the global **Bulk Change** permission. This permission governs the ability to execute the bulk change operations:

- Workflow Transition
- Edit
- Move
- Delete

An upgrade task has been added to grant the new **Bulk Change** permission to all groups with the global **JIRA Users** permission.

The JIRA documentation includes further details on this new permission.

The decision to grant the **Bulk Change** permission should be considered carefully - the permission permits a user to modify a collection of accessible issues at once. For example, in JIRA installations configured to run in 'Public' mode (anybody can sign up and create issues), a user could comment on all accessible issues with the **Bulk Change** and **Add Comments** permission. Undoing such modifications may not be possible through the JIRA UI and may require changes made directly against the database.

CustomFieldPersister changes

CustomFieldPersister is used to store custom field values to database. The methods of this class has been refactored to remove the redundant parameter, `defaultValueMarker`. For example, the create values method went from:

```
void createValues(CustomField field, Long issueId, String defaultValueMarker,
PersistenceFieldType persistenceFieldType, Collection values, String parentKey);
```

to:

```
void createValues(CustomField field, Long issueId, PersistenceFieldType
persistenceFieldType, Collection values, String parentKey);
```

You will need to update and recompile any `CustomFieldType` that you wrote to use this new interface.

VersionCFType Changes

This affects plugin writers who uses the version custom field `VersionCFType`. The change is that previously the Transport Object type was a single `Version` object, but it is now a collection that contains a single `Version` object.

This was done to handle an improved version custom field which can be a mutli-select version custom field as well

JIRA 3.5.1 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5.1 from JIRA 3.5. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.5 to JIRA 3.5.1.

JIRA 3.5.2 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5.2 from JIRA 3.5.1. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Issue Event Changelog Can Now Be Null

- If you have implemented a custom JIRA Issue Event Listener you need to be aware of the following API change.

In JIRA 3.5.2, the IssueEvent object thrown as a result of an edit operation, **may now return null** from a `getChangeLog()` call. The case where this happens is when a user chooses to edit an issue but only leaves a comment and makes no other changes to the issue. Prior to 3.5.2 no event was fired in this case and this was identified as a bug (JRA-9415) and has since been fixed. Check any calls to getChangeLog() for null.

JIRA 3.5.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.5.3 from JIRA 3.5.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.5.3 from JIRA 3.5.2.

JIRA 3.5.x to 3.6.x

JIRA 3.6 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.x from JIRA 3.5.x. If upgrading from an older version of JIRA, please go to the complete list of Upgrade Guides, and read the notes for each version you are skipping during the upgrade.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Database Intensive Upgrade Task

To introduce the Custom events to JIRA, it was necessary to upgrade a large data set within JIRA's database for 3.5.x and earlier releases. Depending on the size of your JIRA data the upgrade task (number 150) might get your DBMS to do a lot of work which might take some time. The exact amount of time also depends on the processing power of the machine running JIRA's database.

Please be patient with the upgrade task and do not restart JIRA while the upgrade is in progress. The upgrade task will report on its progress to JIRA's log file as it upgrades your data.

The following is the sample output that the upgrade task will produce. As you can see the upgrade task took roughly 5 and a half minutes to modify over 660,000 records in the database.

```
11:14:09 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Phone
Support Workflow v.6'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Support
Workflow v.3'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Phone
Support Workflow v.7'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Test'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Copy of
Support Workflow'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Support
Workflow v.4'.
11:14:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Inspecting workflow 'Support
Workflow'.
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150]
************************************************************
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating 660453 records in the
'NotificationInstance' table.
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150] This might take a long time.
Please do NOT stop JIRA.
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150]
************************************************************
11:14:18 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_CREATED'.
11:15:12 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_UPDATED'.
11:15:51 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_ASSIGNED'.
11:16:10 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_RESOLVED'.
11:16:46 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_CLOSED'.
11:16:57 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_COMMENTED'.
11:18:57 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_REOPENED'.
11:19:17 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_DELETED'.
11:19:26 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_MOVED'.
11:19:31 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_WORKLOGGED'.
11:19:37 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_WORKSTARTED'.
11:19:41 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_WORKSTOPPED'.
11:19:43 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Updating records of type
'NOTIFICATION_ISSUE_GENERICEVENT'.
11:21:23 INFO [jira.upgrade.tasks.UpgradeTask_Build150] Update of 'NotificationInstance'
records finished.
```

Workflow Post Functions

Applies to	users with custom workflow XMLs saved on disk - external to JIRA

JIRA stores its workflows in the database. During the upgrade, these workflows will be upgraded automatically. However, if you have stored your workflows on disk (outside the database), you will need to follow these instructions to upgrade the workflows manually.

Previously, workflow post functions referenced the event to fire through a string value of the event name. All post functions now reference the event through a numeric ID value. As mentioned, all workflows stored within JIRA will be automatically updated. However, all workflows saved to disk - external to JIRA - should be updated manually as follows. The actual workflow XML file should be updated as follows:

For each workflow post function that accepts the event ID as an argument:

1. The value of the **name** attribute of the **arg** tag has to be changed from **eventType** to **eventTypeId**
2. The body of the **arg** tag has to change according to the following table:

Event Name	Event Type Id
created	1
updated	2
assigned	3
resolved	4
closed	5
commented	6
reopened	7
deleted	8
moved	9
worklogged	10
workstarted	11
workstopped	12
genericevent	13

By default, the only post functions that accept event IDs are *FireIssueEventFunctions.* Therefore, unless you have implemented your own custom post function that also deals with events, you will only need to update the arg tags for the *FireIssueEventFunctions* everywhere in the workflows.

For example, *FireIssueEventFunction* for create issue workflow transition looked like:

```
<function type="class">
    <arg
name="class.name">com.atlassian.jira.workflow.function.event.FireIssueEventFunction</arg>
     <arg name="eventType">created</arg>
</function>
```

and needs to be changed to:

```
<function type="class">
    <arg
name="class.name">com.atlassian.jira.workflow.function.event.FireIssueEventFunction</arg>
     <arg name="eventTypeId">1</arg>
</function>
```

Custom Events

Applies to	users who have **modified JIRA source code or added custom code to define new notification events**. Also of interest to users wishing to define new notification templates

Releases before JIRA 3.6 did not allow users create custom events. If you have modified the JIRA source to add custom events - please follow these instructions.

If you have previously defined a custom event within JIRA - it is necessary to add appropriate entries to the following files:

- `system-event-types.xml`- used to install and upgrade all event types within the system to the new 3.6 event type object.
- `email-template-id-mappings.xml` - maps the event id to an assocaited velocity template file.

The `system-event-types.xml` file requires name and description details of the previously added custom event. For example, if the custom event type "Issue Frozen" was added to the system - the following entry should be added to the XML file:

```
<eventtype id="10000">
   <name>Issue Frozen</name>
   <description>This is the 'Issue Frozen' event type.</description>
   <notificationName>ISSUE_FROZEN</notificationName>
   <eventName>issuefrozen</eventName>
</eventtype>
```

The elements provide the following information:

- id - the new id for the event type. **All custom event types should be added from ID 10000 and above**
- notificationName - the original name for the event as found in the Notification table
- eventName - the origianl name for the event as found in workflows

The `email-template-id-mappings.xml` file requires an entry mapping the new custom event to an associated velocity email template. This mapping is used when a notification is sent for this event. Following from the above example, the following entry would be made:

```
<templatemapping id="10000">
   <name>Issue Frozen</name>
   <template>issuefrozen.vm</template>
</templatemapping>
```

The id should match that of the event as specified in the `system-event-types.xml` The `template` entity should reference the Velocity template to be used in email notifications of this event. A HTML and text version should be provided in the appropriate directory (html or text) at:

`<JIRA>/src/etc/java/templates/email/`

All custom event types added to the file *system-event-types.xml* should be added with an ID of 10000 and above

Custom Listeners

Applies to	users who have **added custom listeners** to JIRA.

For all users who have **added custom written** listeners to JIRA, it might be necessary to update the listener to follow the new JIRA 3.6 API.

There are two things to look out for:

1. signature change of the *workflowEvent* method
2. change of return type of *getIssue()* method on the *IssueEvent* object

The signature of the method *workflowEvent* in the *IssueEventListener* has changed from:

```
public void workflowEvent(int type, IssueEvent event);
```

to:

```
public void workflowEvent(IssueEvent event);
```

Note: the type parameter has been removed.

If you have implemented *IssueEventListener* directly or have extended *AbstractIssueEventListener* and have overridden the method *workflowEvent*, you will need to change and recompile your listener before installing JIRA 3.6.

In JIRA 3.6, the event type ID can be retrieved by calling the following method on the *IssueEvent* object:

```
Long eventID = event.getId();
```

However, the returned value of the *getId()* method is different to the values of the type parameter that was passed to the *workflowEvent* method. The following table represents these differences:

Event Name	Old ID	New ID
created	0	1
updated	1	2
assigned	2	3
resolved	3	4
closed	4	5
commented	5	6
reopened	6	7
deleted	7	8
moved	8	9
worklogged	9	10
workstarted	10	11
workstopped	11	12
genericevent	-1	13

Also, the *getIssue()* method of the *IssueEvent* object has changed to return an Issue object instead of a *GenericValue* object representing an issue.

Users who have created and added custom listeners must update the listener to now operate with the *Issue* object. For example:

```
Issue issueObject = event.getIssue();
```

As a quick fix, you can modify your listener to use `event.getIssue().getGenericValue()`.

The event type ID constants are now only available from the class *EventType*. Any use of the original constants must be updated to use the *EventType* constants. For listeners that reference an event ID by its numeric value - it is necessary to ensure that the IDs now match those as defined in *EventType*.

Custom permission types

Applies to	users who have modified JIRA source to add new permission types (ie. in addition to the standard 'user', 'group', 'assignee' types).

The SecurityType interface, used to implement permission types ('single user', 'group' etc) has had a `getUsers()` method added. If you have implemented your own SecurityType you will need to implement this. See the source of current implementations (eg. `GroupCF`) for tips.

Plugin upgrades required

As usual, you should check whether the plugins you use are compatible with the new release. Generally, plugins (like the Subversion plugin or JIRA toolkit) need to be upgraded when JIRA is upgraded. See the list of plugins at:

http://confluence.atlassian.com/display/JIRAEXT/Home

JIRA 3.6.1 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.1 from JIRA 3.6. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.6.1 from JIRA 3.6.

JIRA 3.6.2 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.2 from JIRA 3.6.1. If upgrading from an older version of JIRA, please go to the complete list of Upgrade Guides, and read the notes for each version you are skipping during the upgrade.

When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

Maximum Active Databased Connections

Applies to	JIRA Standalone users

In version of JIRA before 3.6.2, the maximum number of database connections was limited to 8 by default. If JIRA was used by more than 8 concurrent users or under very haeavy usages, the users could experience delays or JIRA could hang.

In JIRA 3.6.2 the default number of maximum active database connections has been increased to 20. When upgrading to JIRA 3.6.2, please ensure that your database will allow JIRA to establish 20 connections, or decrease this number to desired value. To adjust the number of connections change the value of the `maxActive` attribute of the `jdbc/JiraDS` resource in `config/server.xml` file. JIRA has to be restarted to apply the change.

JIRA 3.6.3 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.3 from JIRA 3.6.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.6.3 from JIRA 3.6.2.

JIRA 3.6.4 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.4 from JIRA 3.6.3. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading to JIRA 3.6.4 from JIRA 3.6.3.

JIRA 3.6.5 Upgrade Guide

This page contains specific information you need to know when upgrading to JIRA 3.6.5 from JIRA 3.6.4. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading to JIRA 3.6.5 from JIRA 3.6.4.

JIRA 3.6.x to 3.7.x

Once you have upgraded to JIRA 3.7, downgrading to a previous version is not a straightforward task and is not recommended.

JIRA 3.7 Upgrade Notes

This page lists a few things to be aware of when upgrading from previous releases of JIRA to JIRA 3.7. To perform the actual upgrade, see the upgrade documentation.

Note: If you are upgrading from a pre-3.6.5 release, please also refer to the relevant JIRA 3.x Upgrade Guides.

Please note that JIRA 3.7 requires JDK 1.4 or above. Support for JDK 1.3 has been discontinued.

Please note that some new functionality will not be available if you are running JIRA on WebLogic or Orion. The List All Filtersportlet will not be able to fetch the issue counts for each issue. The new 'Charting' View will also be unavailable. The support for WebLogic and Orion will be added in JIRA 3.7.1.

Database Schema Changes

Due to the upgrade of HSQLDB, and to improve compatibility with Firebird and Frontbase, various database tables and columns have been renamed. For more details on the changes please see the JIRA 3.7 Database Schema Changes document.

Therefore, the easiest way to upgrade to JIRA 3.7 is to follow the Upgrading JIRA safely instructions.

If in the past, instead of performing an XML backup and restore, you have been upgrading by "pointing" new version of JIRA at an old database, this is still possible, however the procedure is more complicated. You will need to use SQL scripts to perform database schema changes. For more information please see the SQL Scripts for 3.6.x to 3.7 schema upgrade document.

If you are using HSQLDB with JIRA, you must follow Upgrading JIRA safely instructions (i.e. perform a full XML backup and restore from XML), as simply copying the `.script` file will not work. The format of the `.script` file has changed between the HSQLDB versions, and therefore, copying the `.script` file will result in the following error on startup.

Request Context Changes

In order for plugins, customfields and portlets to function better outside of a web-context (e.g.: displaying a customfield in an e-mail), all direct references to the HttpServletRequest have been replaced by a VelocityRequestContext. If you have deployed your own plugins, customfields or portlets that use the HttpServletRequest directly (i.e.: any references to `${req}`) than they should be changed to use the new `${requestContext}` object. The `${requestContext}` is an implementation of the `VelocityRequestContext` interface.

Currently the `${requestContext}` supports the following properties:

- `${requestContext.baseUrl}` - Returns the same as HttpServletRequest.getContextPath() or the base URL configured in your JIRA instance if no HttpServletRequest is available
- `${requestContext.requestParameters}` - Returns an implementation of `RequestContextParameterHolder` or null if no HttpServletRequest is available
- `${requestContext.requestParameters.servletPath}` - Returns the same as HttpServletRequest.getServletPath()
- `${requestContext.requestParameters.requestURL}` - Returns the same as HttpServletRequest.getRequestURL()
- `${requestContext.requestParameters.queryString}` - Returns the same as HttpServletRequest.getQueryString()

Integrity Checks

In JIRA 3.7 Database Integrity Checks (available from the Administration section) have been re-written to run as multiple transactions, which increased the throughput of the system while the checks are running. In large JIRA 3.6 (and earlier) installations, integrity checks could cause database lock escalation and stop users from performing operations (e.g. viewing issues).

Please note, that due to the change, each integrity check became about 10% slower.

As integrity checks are quite database intensive operations, it is still recommended to run them during off-peak hours (i.e. while the system is not under heavy load).

Change of commentLevel to groupLevel in the Comment and TransitionWorkflow jelly tags

We have changed the AddComment and TransitionWorkflow jelly tag attribute that specifies the group visibility level from 'commentLevel' to be 'groupLevel'. If you have existing jelly tags that use this attribute it will need to change. This was done so that we could introduce the 'roleLevel' attribute which allows you to specify a project role based visibility. Only one of the two attributes can be specified at a time.

Change of level to grouplevel in the XML view of a Comment

1. We have changed the XML view of a comment, as seen in the XML view of an Issue to contain either a 'grouplevel' attribute or a 'rolelevel' attribute. This attribute defines the visibility level specified on the comment. In the past the 'grouplevel' attribute was simply 'level'. If you have any existing custom code that expects the 'level' attribute in the Comment XML it must change to expect 'grouplevel'.
2. In previous versions of JIRA the XML view of the <comment> tag level attribute was always shown, even if there was no value for the level, it was rendered as an empty attribute. We have changed it so that the attributes themselves (grouplevel and rolelevel) do not display if there is no value.

Change to the RemoteComment object used via SOAP/RPC plugin

The RemoteComment object and therefore the remote SOAP/RPC api has changes to almost all properties. The 'roleLevel' attribute was added and the following attributes have changed:

1. level -> groupLevel
2. datePerformed -> created
3. username -> author

ActionManager removed

The `ActionManager` interface has been removed and its functionality has been delegated to new interfaces. For details please refer to ActionManager Removed documentation

Removal of 'Backend Actions'

1. We have removed the 'Backend Action' `com.atlassian.jira.action.action.WorklogCreate` if you were using this class in a plugin or custom code you will now need to use the `com.atlassian.jira.issue.worklog.WorklogManager` this now has method calls to return worklogs for a given user+issue and also create worklog entries.
2. We have removed the 'Backend Action' `com.atlassian.jira.action.action.ActionCreate` if you were using this class to create comments you will need to modify your code to use one of the create methods on the com.atlassian.jira.bc.issue.comment.CommentService

Issue Events

We have modified the `com.atlassian.jira.event.issue.IssueEvent` class to no longer use GenericValues. The GenericValue representing the comment is replaced by `com.atlassian.jira.issue.comments.Comment` class and the GenericValue representing the worklog is replaced by `com.atlassian.jira.issue.worklog.Worklog` class. If you have a custom listener in a previous version of JIRA this will need to be updated to use the newer IssueEvent class and `com.atlassian.jira.event.issue.IssueEventDispatcher.dispatchEvent(...)` methods.

Renaming of XML export file

By popular request, the XML filename (that is, the default filename when you choose to save the XML view in the Issue Navigator) has been changed from `issuenavigator.jspa` to `SearchRequest.xml`. Should you have any external systems or programs that utilise the exported XML file, please be aware of the changed filename.

Confluence Users Only - Pre 2.2.10 Confluence Must Be Patched To Use JIRA Issues Macro

Unable to render {include} The included page could not be found.

JIRA 3.7 Downgrade Notes

Once you have upgraded to JIRA 3.7, downgrading to a previous version is not a straightforward task and is not recommended. Please be aware that in JIRA 3.7 the database schema has changed.

If upgrade to JIRA 3.7 fails, the best way to proceed is to go back to the previous version of JIRA you were using, and to the latest **pre-upgrade** data that you have. The exact steps for doing this depend on how you have upgraded JIRA.

If you have created a new database for JIRA 3.7 by following the Upgrading JIRA safely instructions, you should be able to simply shutdown JIRA 3.7 and bring up the old version of JIRA your were using. The old version should be configured to use its old (unupgraded) database.

If you have upgraded JIRA by pointing JIRA 3.7 to an older database (and ran the SQL Scripts to upgrade the database schema), then you will need to:

1. Create a new database
2. Configure the old version of JIRA you were using to point at the new (empty) database
3. Restore the latest **pre-upgrade** backup that you have
4. Start the old JIRA installation

JIRA 3.7.1 Upgrade Guide

This page contains specific information you need to know when upgrading from JIRA 3.7 to JIRA 3.7.1. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.7 to JIRA 3.7.1.

JIRA 3.7.2 Upgrade Guide

This page contains specific information you need to know when upgrading from JIRA 3.7.1 to JIRA 3.7.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- 3.7.2 will automatically perform a full reindex when upgrading. For more details please see JRA-11861

Upgrading from JIRA 3.7.2 to 3.7.3

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.7.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.7.3 to 3.7.4

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.7.2 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.7.x to 3.8.x

Upgrading from JIRA 3.7.4 to 3.8

Please follow the JIRA general upgrade instructions. Additionally, please note the following:

1. The 'Assign To' field name has been changed to 'Assignee' consistently across JIRA. This means that users need to be aware that the column heading in the Excel export has changed to 'Assignee' from 'Assign To'. Please be aware of this if for example you are exporting JIRA data to Excel and running macros on it. The field has been renamed for the following Issue Navigator Views:
    - Excel (all)
    - Word (all)
    - Full Content
2. The `issuecommentedited.vm` e-mail template for the new Issue Comment Edited event has been added to the `WEB-INF/classes/email-template-id-mappings.xml` file. The id of the e-mail template used for sending Filter Subscriptions has changed to 10000. **If** you have manually modified the `WEB-INF/classes/email-template-id-mappings.xml` file in the version of JIRA you are upgrading from, please do not simply copy the old file to JIRA 3.8. You will need to port your changes to the `WEB-INF/classes/email-template-id-mappings.xml` file that is shipped with JIRA 3.8. If you have not changed the `WEB-INF/classes/email-template-id-mappings.xml` file, you do not need to worry about this.
3. Two columns have been added to the `jiraaction` table to support editable comments.

Upgrading from JIRA 3.7.3 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.8 to 3.8.1

Please follow the JIRA general upgrade instructions.

> **Charting Plugin must be upgraded to v1.3.5**
>
> Please note that the version of JFreeChart included in JIRA 3.8.1 is **not compatible** with older versions of the Charting Plugin. If you have the Charting Plugin installed, please make sure you upgrade it to version 1.3.5 or above.
>
> The updated JFreeChart 1.0.4 version is **not** backwards compatible with the previous 1.0.0pre2 version, so if you have any plugins that utilise JFreeChart, please make sure you test them before upgrading.

Upgrading from JIRA 3.7.4 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.8.x to 3.9.x

Upgrading from JIRA 3.8.1 to 3.9

Please follow the JIRA general upgrade instructions. Additionally, please note the following:

In this version, there has been a change to the database which may cause problems for some customers.

The Recommended Upgrade Method

If you follow the recommended export/import upgrade procedure you should not experience any problems!

Pointing JIRA 3.9 at an existing, non-empty database

Some customers have a good reason for not following the recommended upgrade method. Using this method may result in database errors in your logs. You can avoid this if you modify your table structure manually, but the procedure is different depending on whether you have already started JIRA.

To avoid this, **BEFORE** you upgrade JIRA using this method, you can just drop the qrtz_cron_triggers table. This table has not been used by JIRA before 3.9, so it should be empty.

If you have **ALREADY** started JIRA 3.9 using your existing database, you may see the following log messages when JIRA starts up:

```
2007-04-18 15:31:53,345 main WARN [core.entity.jdbc.DatabaseUtil] Column
"CRON_EXPERSSION" of table "public.qrtz_cron_triggers" of entity  "QRTZCronTriggers"
exists in the database but has no corresponding field
2007-04-18 15:31:53,347 main WARN [core.entity.jdbc.DatabaseUtil] Entity
"QRTZCronTriggers" has 3 fields but table "public.qrtz_cron_triggers"  has 4 columns.
```

The reason for this is that we have incorrectly changed a column in the `qrtz_cron_triggers` table. The intention was to fix a misspelling, but all we did was remove an underscore ("_")! The old column name is "CRON_EXPERSSION". The new column name is "CRONEXPERSSION". Note that both columns spell the word "expression" incorrectly.

To remove the error message, you must remove the old column as it is redundant. This column will not contain any data. The following table shows all columns in the qrtz_cron_triggers table. Columns that should be present are in **green** and columns that should be deleted are in **red**.

Keep	Keep	Keep	Delete
ID	TRIGGER_ID	CRONEXPERSSION	CRON_EXPERSSION

To delete the column, you can use SQL, but this may be slightly different between databases. Here's how it might look:

```
alter table qrtz_cron_triggers drop column CRON_EXPERSSION;
```

The data in this table

If you have users who have subscribed to issue filters, note that existing SimpleTriggers (time intervals) will be automatically converted into CronTriggers during the JIRA upgrade. In some cases, there may not be an exact mapping of time intervals to Cron Expressions, and approximations will be made (e.g. 'Every 5 weeks' will be converted to 'Once a month'). If this happens, the JIRA upgrade process will send an email to the user to inform them of the new schedule.

Upgrading from JIRA 3.8 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.9 to 3.9.1

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.8.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.9/3.9.1 to 3.9.2

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.8.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Upgrading from JIRA 3.9.2 to 3.9.3

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.9.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.9.x to 3.10.x

Upgrading from JIRA 3.9.3 to 3.10

Please follow the JIRA general upgrade instructions, plus note the following:

1. Plugins

There is a new version of the JIRA Calendar Plugin that links to the new 'Project Version' pages. This new version of the plugin is not backwards compatible.

Please note that the Kaamelot plugin for JIRA has not yet been updated. If you are currently using this plugin, you may want to hold off the upgrade to JIRA 3.10 until a compatible version of this plugin has been released.

2. Developer Notes

The ordering of the ListOrderedMap returned by SchemePermissions.getSchemePermissions() has changed. This also means that the order of the RemotePermission[] array returned by the RPC Plugin's JiraSoapService.getAllPermissions() method has changed. If you have extended your instance of JIRA please confirm that any remote applications retrieving permissions via SOAP still work. You may encounter problems if you have been retrieving specific permissions by their array index.

Database changes

In JIRA 3.10, the worklog records have moved from the 'jiraactions' database table to the new 'worklog' table. This new table contains the following columns:

```
Table "public.worklog"
    Column    |           Type           | Modifiers
--------------+--------------------------+-----------
 id           | numeric(18,0)            | not null
 issueid      | numeric(18,0)            |
 author       | character varying(255)   |
 grouplevel   | character varying(255)   |
 rolelevel    | numeric(18,0)            |
 worklogbody  | text                     |
 created      | timestamp with time zone |
 updateauthor | character varying(255)   |
 updated      | timestamp with time zone |
 startdate    | timestamp with time zone |
 timeworked   | numeric(18,0)            |
```

Upgrading from JIRA 3.9.2 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.10 to 3.10.1

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.9.3 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.10.1 to 3.10.2

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.9.3 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.10.x to 3.11.x

- Upgrading from JIRA 3.10.x to 3.11
- Upgrading from JIRA 3.9.x and earlier

Upgrading from JIRA 3.10.x to 3.11

Please follow the JIRA general upgrade instructions, plus note the following:

Administrative notes

- To take advantage of the performance enhancements in JIRA 3.11, it is recommended that you enable GZip compression (unless you are using mod_proxy).
- The jira-application.properties file has a new option, 'progress', for the following attribute:

```
jira.table.cols.subtasks
```

  The 'progress' option controls the display of the '**Progress**' field in issues and reports.
- JIRA 3.11 introduces a bug fix for JRA-12354. This means that the CVS and Perforce plugin will perform better at detecting commits for a particular issue key, avoiding partial matches on similar project keys. If users have taken advantage of the previous relaxed key matching, they can revert to the old behaviour by simply setting the following application property in the jira-application.properties file and restarting JIRA:

```
jira.option.key.detection.backwards.compatible=true
```

Plugins

Updating plugins
If you are using any of the following plugins, you will need to update them to their latest versions when performing the upgrade:

- Perforce plugin
- Subversion plugin
- Toolkit Plugin
- Charting Plugin
- RPC Plugin

3rd Party and personal plugins may also be affected (esp. if using lucene to store dates). These will need to be updated as well.

If these are updated after the upgrade (instead of as part of the upgrade), you will need to do a reindex.

A failure to update these plugins will result in lots of errors that look like:

Error 1

```
2007-07-25 15:23:27,553 http-8090-Processor4 ERROR
[charting.charts.createdvsresolved.CreatedVsResolvedChart] Could not create velocity
parameters For input string: "20070725144811"
java.lang.NumberFormatException: For input string: "20070725144811"
 at java.lang.NumberFormatException.forInputString(NumberFormatException.java:48)
 at java.lang.Long.parseLong(Long.java:415)
 at org.apache.lucene.document.DateField.stringToTime(DateField.java:100)
 at org.apache.lucene.document.DateField.stringToDate(DateField.java:104)
 at
com.atlassian.jira.ext.charting.data.DatePeriodStatisticsMapper.getValueFromLuceneField(DateP
at
com.atlassian.jira.ext.charting.data.OneDimensionalObjectHitCollector.adjustMapForValues(OneD
at
com.atlassian.jira.ext.charting.data.OneDimensionalObjectHitCollector.collect(OneDimensionalO
at org.apache.lucene.search.IndexSearcher$1.collect(IndexSearcher.java:137)
 at org.apache.lucene.search.Scorer.score(Scorer.java:49)
 at org.apache.lucene.search.IndexSearcher.search(IndexSearcher.java:146)
 at org.apache.lucene.search.Searcher.search(Searcher.java:118)
 at
com.atlassian.jira.issue.search.providers.LuceneSearchProvider.search(LuceneSearchProvider.ja
```

Error 2

```
Caused by: java.lang.NoSuchMethodError:
org.apache.lucene.document.Document.add(Lorg/apache/lucene/document/Field;)V
 at com.atlassian.jira.plugin.labels.LabelSearcher.index(LabelSearcher.java:95)
 at
com.atlassian.jira.issue.index.indexers.impl.DefaultCustomFieldIndexer.addIndex(DefaultCustom
at com.atlassian.jira.issue.index.IssueDocument.getDocument(IssueDocument.java:34)
 at
com.atlassian.jira.issue.index.IssueDocumentBuilderImpl.get(IssueDocumentBuilderImpl.java:14)
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$IssueAndCommentCreator.handleIssueI
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$AbstractIssueAndCommentHandler.inde
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer.indexIssuesAndComments(SingleThread
at
com.atlassian.jira.issue.index.MultiThreadedIssueIndexer.indexIssuesAndComments(MultiThreaded
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$2.perform(SingleThreadedIssueIndexe
at
com.atlassian.bonnie.ConcurrentLuceneConnection.withWriter(ConcurrentLuceneConnection.java:29
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$1.perform(SingleThreadedIssueIndexe
at
com.atlassian.bonnie.ConcurrentLuceneConnection.withWriter(ConcurrentLuceneConnection.java:29
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer.indexIssues(SingleThreadedIssueInde
at
com.atlassian.jira.issue.index.SingleThreadedIssueIndexer$6.perform(SingleThreadedIssueIndexe
```

If you see these errors, please ensure that you are using the latest compatible version of the plugin for 3.11. If there is no supported version for 3.11, please contact the plugin developer via the plugin's homepage.

Developer notes

Modification to SOAP clients
If you have written a SOAP client for any JIRA version prior to 3.11 and are invoking any methods to get `RemoteIssueType` you will encounter the bug JRA-13529. The reason for this is that we have added extra information to the `RemoteIssueType` object that indicates if the issue type is a subTask issue type. To avoid the problem you will need to regenerate your remote object stubs against the updated JIRA 3.11 wsdl.

If you would like your SOAP client to work against multiple versions of JIRA then you need to use the latest stubs that have been generated against JIRA 3.11. You will need to **not** use any of the new functionality and you will need to remember that the `isSubTask` variable in the `RemoteIssueType` objects will be defaulted to `false`.

ThreadLocalQueryProfiler searchers have been moved to ThreadLocalSearcherCache
There may be a number of plugins that reference the ThreadLocalQueryProfiler searcher methods directly. These need to now reference the ThreadLocalSearcherCache.

Lucene Upgrade
We upgraded our version of Lucene to 2.2. If your plugin uses to Lucene to index/read data, please ensure that it works with JIRA 3.11. If you are indexing/reading dates, more than likely it will have broken and you will need to use the new Lucene 2 methods.

Database changes

There were no database changes in this release.

Upgrading from JIRA 3.9.x and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.11 to 3.12.x

Upgrading from JIRA 3.11 to 3.12

Please follow the JIRA general upgrade instructions, plus note the following:

1. Everyone who had the '**JIRA Administrators'** global permission before the upgrade will automatically receive the new '**JIRA System Administrators**' global permission during the upgrade. This will ensure that everyone can still perform the same functions they could previously.
2. The following new Seraph property can be used to fix JRA-10508:

```
<!--  If this parameter is set to true, the cookie will never be set secure.
This is useful if you're logging
              into JIRA via https, but want to browse JIRA over http.  This
flag will ensure that the remember me option
              works correctly. -->
        <init-param>
            <param-name>insecure.cookie</param-name>
            <param-value>true</param-value>
        </init-param>
```

3. Due to the Seraph upgrade, to fix JRA-10508 all users will be prompted to log in again. This will also affect users who have the 'Remember me' checkbox ticked.
4. If you are building JIRA from source, please note that Maven2 is now required for a build. This is because the JIRA Fisheye Plugin requires Maven2.
5. If you are using the JIRA Toolkit, it is recommended that you upgrade to the latest version in order to fix JRA-13553
6. Please note that the new Trusted Applications feature is not supported on Orion versions prior to 2.0.5. Also note that Resin2 has problems and you will need to update the Resin extra jars.
7. There is a new database table. Please see the following page for details

Upgrading from JIRA 3.10.2 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Using the Trusted Applications feature with Crowd

Please note that older versions of the Crowd client, (i.e. version 1.2.1 or earlier), can interfere with the correct operation of the Trusted Applications feature. If you are enabling Trusted Applications and using Crowd, please ensure that your Crowd client is **version 1.2.2 or later**.

Upgrading from JIRA 3.12 to 3.12.1

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.11 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.12.1 to 3.12.2

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.11 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

Upgrading from JIRA 3.12.2 to 3.12.3

Please follow the JIRA general upgrade instructions

Upgrading from JIRA 3.11 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.12.x to 3.13

Upgrading from JIRA 3.12.xx to 3.13

Please follow the JIRA general upgrade instructions, plus note the following:

1. Introduction of Favourite Dashboards and Filters

Favourite Dashboards

JIRA 3.13 introduces the favourite dashboards feature, which allows you to add dashboard pages that are owned by you or shared by other users as favourites (and hence, are displayed as tabs on your dashboard). On upgrade to JIRA 3.13, all your dashboard pages will be added as your favourites and displayed on your dashboard. If you do not wish any of your dashboards to be added as favourites, then you can remove them as favourites after the upgrade. See the dashboards documentation for details.

Favourite Filters

Similar to favourite dashboards, JIRA 3.13 introduces the favourite filters feature, which allows you to add issue filters that are owned by you or shared by other users as favourites. On upgrade to JIRA 3.13, all your issue filters will be added as your favourites. If you do not wish any of your filters to be added as favourites, then you can remove them as favourites after the upgrade. See the issue filters documentation for details.

please note, this change will not affect issue filter sharing, e.g. if you are using a shared issue filter in one of your dashboard portlets, it will still be shared with you after the upgrade.

please also note, that any custom developed portlets (or other JIRA objects that use filters that have been developed by 3rd parties) that have a dropdown list (not a pop-up picker) for filters, will now only show a list of the user's favourite filters, instead of all shared filters.

Favourite Filters portlet

The 'List All Filters' portlet has been replaced with the 'Favourite Filters' portlet in this release. Your dashboard will be automatically upgraded if it is currently configured to display the 'List All Filters' portlet.

2. Tomcat, MySQL database connection dropouts

Please note, if you wish to use a MySQL database with JIRA Standalone you must set up the bundled Tomcat server (version 5.5.26) to survive connection closures. You must also do this if you are running JIRA EAR/WAR in Tomcat 5.5.25 or later, or Tomcat 6.0.13 or later. Versions 5.5.25 and above of Tomcat 5, and versions 6.0.13 and above of Tomcat 6, have been noted to exhibit problems maintaining connections to MySQL databases. Please read this document for details on the changes required.

3. Changes to jira-application.properties

jira.subscription.email.max.issues property

The `jira.subscription.email.max.issues` property has been added to the `jira-application.properties` file. This property allows you to specify the maximum number of issues that can be included in an email subscription. The default value for this property is 200. You may wish to update this property after the upgrade if you wish to set a different limit on the number of issues that can be included in an email subscription. See the documentation on Advanced JIRA Configuration for further details on this file.

4. Support for Portlet Plugins with JSP Views Discontinued

Portlet plugins with JSP views are no longer supported. If you have written a custom Portlet plugin and have used a JSP as the view template, you will need to convert your JSP to Velocity.

5. Updates to JIRA SOAP and XML-RPC APIs

com.atlassian.jira.rpc.soap.JiraSoapService

- replaced

```
RemoteProject[] getProjects(String token) throws RemoteException;
```

  with

```
RemoteProject[] getProjectsNoSchemes(String token) throws
RemoteException
```

*You should use **getProjectsNoSchemes()** instead because it much more memory efficient and quicker.*

- added

```
RemoteProject getProjectWithSchemesById(String token, Long
projectId) throws RemoteException;
```

- deprecated

```
RemoteFilter[] getSavedFilters(String token) throws
RemoteException;
```

- added

```
RemoteFilter[] getFavouriteFilters(String token) throws
RemoteException;
```

com.atlassian.jira.rpc.xmlrpc.XmlRpcService

- replaced

```
Vector getProjects(String token) throws Exception;
```

  with

```
Vector getProjectsNoSchemes(String token) throws Exception;
```

- deprecated

```
Vector getSavedFilters(String token) throws Exception;
```

- added

```
Vector getFavouriteFilters(String token) throws Exception;
```

6. Crowd Cache Timeout

This is only applicable if you are using Crowd.

The default timeout for caching user details has changed from 5 minutes to 2 hours. This will improve the performance of the application but will mean that it will take longer for changes to user details to reach the application. Details on how to configure the Crowd caches can be found here.

Upgrading from JIRA 3.12 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 2.6 Release Notes

JIRA 2.6

Only three months after the release of JIRA 2.5 (and two after our last point-release), Atlassian are continuing our tradition of frequent, worthwhile upgrades with JIRA 2.6. In the process, we've fixed more than 110 known bugs, and added over 70 individual improvements or new features. All a free upgrade if you've purchased your JIRA license in the last year.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

As always, we strongly encourage you to back up your data before upgrading.

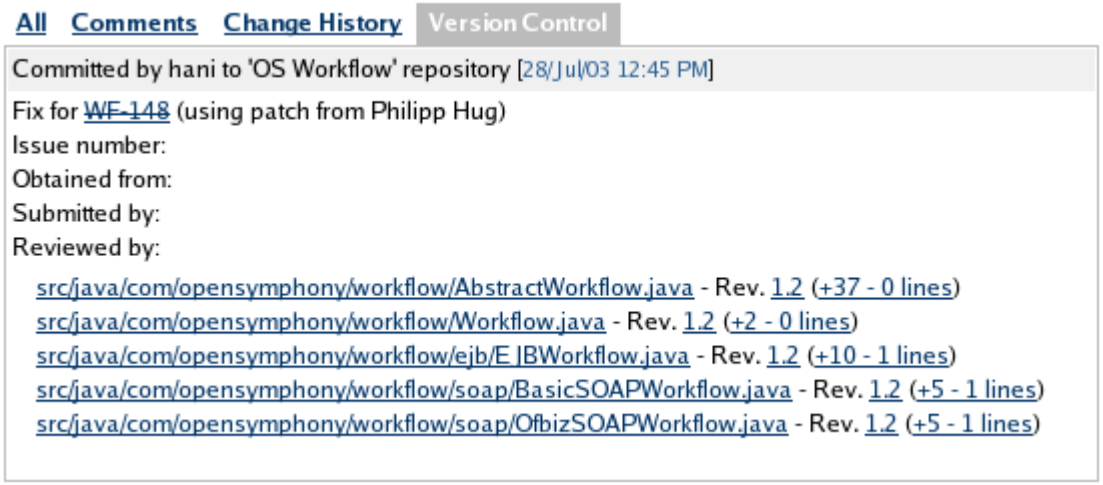
Contents

1. New Features
2. Improvements
3. Notable Bug-fixes
4. Outstanding Issues
5. Enterprise Features

New Features

CVS Integration

Prior to 2.6, the only way to integrate JIRA with CVS was via syncmail, a clumsy and error-prone solution. For 2.6, we have overhauled the CVS support so that now JIRA can get all the information it needs directly from the CVS repository itself. Wherever a JIRA issue key is mentioned in a CVS commit message, JIRA will link the commit to the issue, as shown below:



Enterprise customers also have the ability to associate multiple CVS repositories with a single project, just in case your project spans multiple repositories, or moves from one to another.

Quick Search

The quick search bar on the top right of the screen is a very useful feature in JIRA, allowing you to quickly jump to an issue by its key, or run a full text search. If you use the keyboard shortcut (*ALT+Q*), then it is even quicker.

In JIRA 2.6, we've made the quick search even more powerful: it now tries much harder to guess what you are searching for. It will recognise the following in your search terms, and narrow your search accordingly.

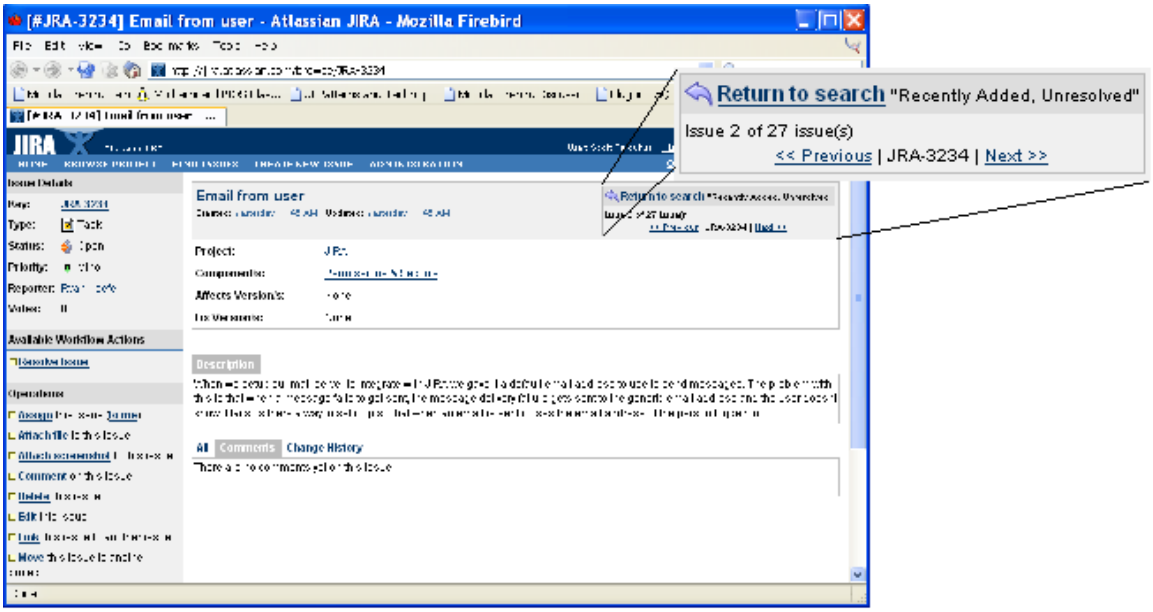
- Project keys
- Project names (single word names only)
- Issue statuses
- Issue types
- "my"

So, for example:

- "JRA open bugs" will search for all open bugs in the JIRA project.
- "my JIRA open improvements" will search for all open improvements in the JIRA project that are assigned to you.

Streamlined Search Results

On the subject of searching, we've improved the site's navigation by giving you the ability to move through your list of search results without returning to the search results page. When you visit an issue from a search, the navigation box shown below will allow you to skip quickly to the next and previous results, conveniently bound to the hot-keys *ALT+N* and *ALT+P* respectively. *ALT+F* will take you back to the search results page.



Per-Search Column Ordering (Enterprise)

Users of the Enterprise version of JIRA can now have custom column-ordering for each of their saved searches (previously, there was just one column-ordering preference that applied to every search). This gives you much more power to build custom issue reports containing just the information you want to know, in the right order.

XML-RPC/SOAP interface to JIRA

JIRA now has an external programmatic API, a much-demanded feature from users who wanted a more direct way to interact with their issue-tracker, or to integrate JIRA more seamlessly with their business. It also provides a raft of new features for Mark Derricutt's Intellij IDEA JIRA Plugin. The API is available in SOAP and XML-RPC flavours, and is documented here.

You can also get an XML view of any issue by adding '?view=rss' to the end of any issue page's URL.

If you come up with some interesting use for the API, or application that takes advantage of it, let us know. Similarly, if you have any suggestions for features that would make the API more useful to you, don't hesitate to tell us.

Screenshots and Thumbnails

Our new screenshot feature makes it easy to attach screenshots to your issue: it's as simple as pasting your screenshot directly into the applet provided, and hitting enter. (Currently, this feature is only available for Windows clients)

Any image that is attached to an issue is automatically displayed as a thumbnail, giving you a much better idea of what each attachment might contain. You can see an example of this online: JRA-2789. These thumbnails can also be configured to display in search results, for organisations that make heavy use of screenshots in issue reports.

Trackback and Confluence Integration

Confluence is the latest Atlassian product, and you can now link Confluence docs to JIRA issues and vice versa. When a JIRA issue contains

a link to a Confluence page the server will automatically inform Confluence that it has been linked to, so Confluence can in turn refer back to the issue. This works both ways, of course, so when a Confluence page refers to a JIRA issue, JIRA also links back to the page. (For example, see http://jira.atlassian.com/browse/JRA-2789)

This is all done with the standard trackback API. Trackback is also widely supported by weblogging software, so you can see when people are talking about a particular issue on their blogs.

If you don't want trackback, you can easily turn it off in the administrative configuration. (By default, JIRA will receive trackback pings, but not send them)

Email Integration

JIRA's email integration is greatly improved. For people sending email to JIRA:

- JIRA can be configured to create new user accounts for anyone who sends an email to JIRA who does not already have an account (great for support!)
- HTML emails that do not contain a plaintext alternative will be converted to text
- Email attachments are automatically added as attachments to the issue

And for people receiving email from JIRA:

- Emails sent by JIRA are properly threaded, so if your email client supports threading, all notifications on a particular issue will be grouped together.
- JIRA's emails will no longer prompt mail clients to send 'vacation' messages (which previously would cause erroneous issues to be created)
- JIRA can be configured to use a different "from" address per project.
- Administrators can now send emails to a group of JIRA users from within JIRA.

Release Notes

JIRA can now produce release notes: a cleanly formatted changelog of issues that have been resolved since the previous version. You can see the 2.6 Professional release notes online, or read the documentation.

The release-notes page contains a convenient text-box that allows you to cut and paste the HTML source directly from JIRA to your website.

Page Compression

JIRA now ships with a gzip filter, which will compress pages before sending them to the web browser. In some pages, this results in a size-reduction of 90%, massively decreasing download times and bandwidth usage for JIRA installations. This is most useful for installations of JIRA on the Internet: if you enable it on your LAN, the time the server takes to compress the pages will likely be greater than the time saved by them being compressed.

It is not enabled by default, but can be enabled from the Administration pages.

Improvements

Internationalisation

- Searching in all UTF-8 languages is now supported.
- The process of translating JIRA has been improved: adding support for a new language is now as simple as dropping in the localised jar file and restarting.
- The Calendar popup window now works in your selected language.
- JIRA i18n now works on Resin 2.1.12 on linux.

Bulk Edit

- Bulk deleteof issues is now supported
- Bulk edit can set issues' fix-for version

Import
There is now a Mantis import available, and the Bugzilla import has been greatly improved.

Other Features

There are 70 new features or improvements. Take a look.

Notable Bug-fixes

There are over 110 Bug Fixes in this release. If you raised a bug, chances are that it is fixed.

Outstanding Issues

- Currently we only have internationalised files for JIRA 2.5.3, so much of the interface is only half translated. We are in the process of co-ordinating the translation of 2.6, and new translations will be made available in future 2.6 point releases.
- If you are upgrading from 2.6 RC1, mail threading may not work correctly for issues that were created before the upgrade unless you upgrade via export / import. Issues created after upgrading to the 2.6 release will thread correctly.
- The printable view for a single issue is broken: it just redirects you to the normal, less printer-friendly page.

JIRA 3.1 Release Notes

JIRA 3.1 Release Notes

Following JIRA 3.0.3, Atlassian is proud to release the latest version of JIRA in Standard, Professional and Enterprise editions - **JIRA 3.1** (download).

> **JIRA 5.0** has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

In the tradition of frequent, worthwhile upgrades, JIRA 3.1 includes over 80 bug fixes, improvements and new features. See the JIRA 3.0 Upgrade Notes before upgrading.

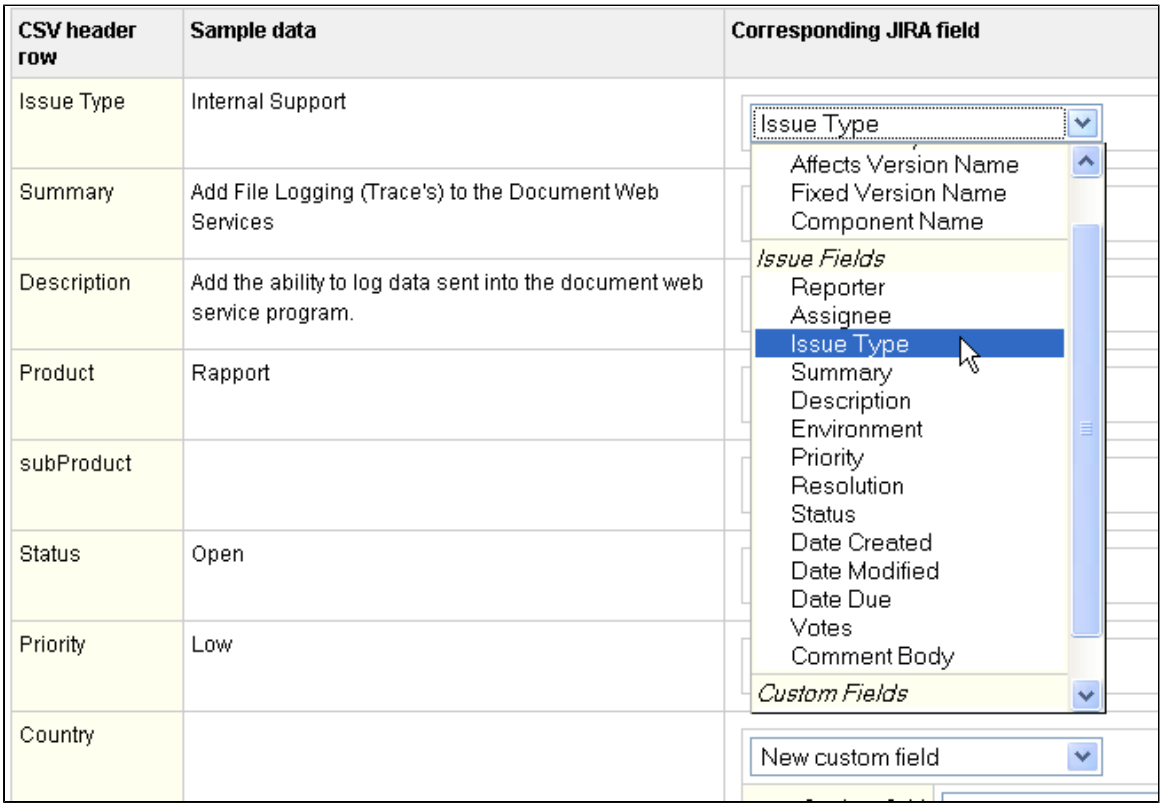
Contents

1. New Features
2. Improvements
3. Bug Fixes
4. Editions

New Features

CSV Importer Wizard

The CSV importer, new in JIRA 3.1, allows you to import issues from any comma-separated file. JIRA's fully configurable wizard will step you through the process of converting your import file into JIRA issues.



The wizard provides the following features:

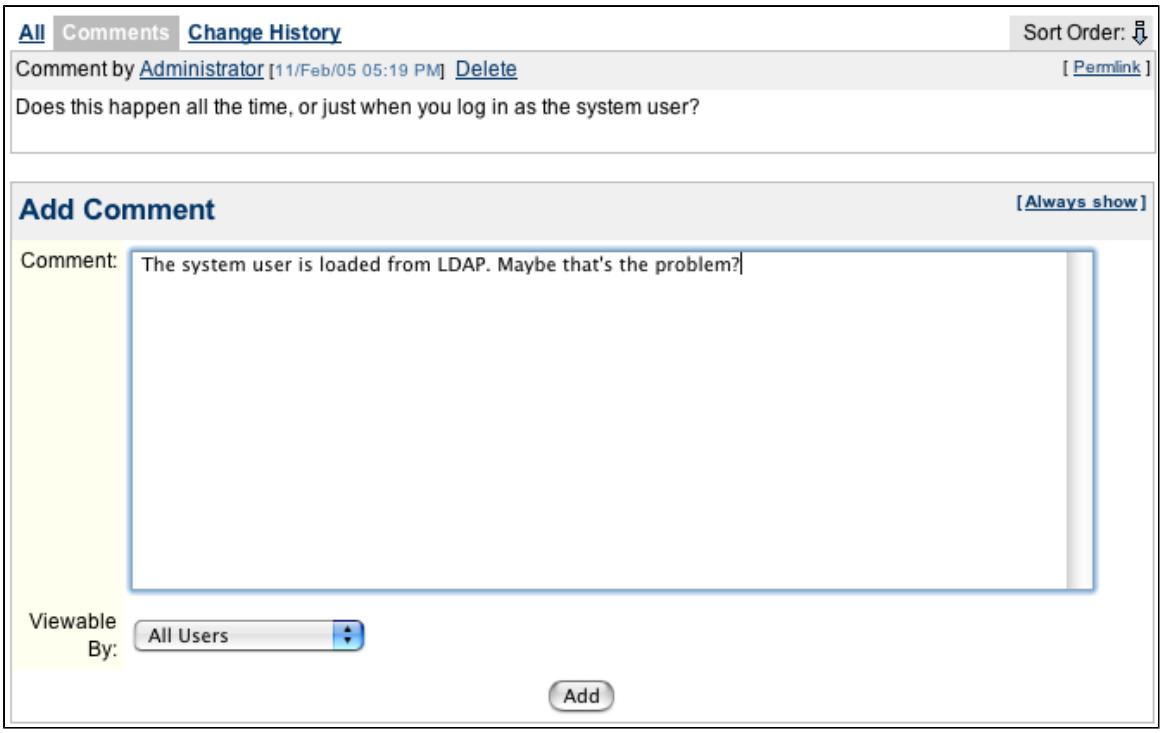
- Automatic creation of missing users, versions and components
- Easy creation of resolutions, priorities and issue types.
- Creation of custom fields on the fly
- Simple mapping of issue values to values in JIRA

Webwork Plugin Type

JIRA 3 plugins allow developers to extend JIRA in many different ways - with new reports, custom fields, workflow conditions and more. JIRA 3.1 gives developers even more power with the webwork plugin type: enabling plugin creators to integrate their own actions right into JIRA's web framework.

Add Comment on 'View Issue' field

A much-requested feature, you can now see what you are commenting on! The comment panel is now displayed in the same page as the issue details, so you don't have to switch between comment and issue to remember what you're talking about.



Workflow Action Keyboard Shortcuts

For the power-user or just the impatient, alt-1 to alt-9 (ctrl-1 to ctrl-9 on OS X) will execute the corresponding workflow action for an issue.

Improvements

RPC / SOAP Improvements
The RPC/SOAP integration with JIRA has been improved to include new types (e.g. RemoteGroup, RemoteUser, RemotePermission, etc.) and new services (e.g. IssueService. ProjectService, UserService, etc.). A full list of the new features is available here.

Improved Performance

JIRA 3.1 includes a number of performance improvements. Most notably, the data import process has been revised - now requiring less memory with the process completing in less time. The import process should also handle invalid characters encountered in the XML. A number of reports regarding slow responses from JIRA while editing workflows have also been addressed.

Assign Issues by Mail

The 'Create Issue Handler' can be configured so that issues created via email are automatically assigned to the user listed in the email's 'CC:' field.

Internet Explorer UI Compatibility

This release contains many fixes addressing issues with Internet Explorer UI incompatibilities. All pages should appear identically - whether viewed in IE or Firefox or Safari or ...

Notable Bug Fixes

JIRA 3.1 includes over 70 Bug Fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Notable Bug Additions

The following bugs have been reported so far, and will be fixed in 3.1.1:

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Edtions

Standard? Professional? Enterprise? Which edition is for you?

In order to provide you with greater choice, JIRA is offered in a number of editions - with different feature sets and capaibilities available in each edition.

Check out the feature comparison at:

http://www.atlassian.com/software/jira/comparison.jsp

or download an evaluation to determine which edition best addresses your requirements.

JIRA 3.1 Upgrade Notes

This page lists a few things to be aware of when upgrading from JIRA 3.0.x to JIRA 3.1. To perform the actual upgrade, see the upgrade documentation. For upgrading from JIRA 2.x to JIRA 3.x see JIRA 3.0 Upgrade Notes

MySQL Users dB upgrade (JRA-5635)

The size of the descriptor field in the jiraworkflow table has been increased. MySQL users will see warnings when they start their app server. This can be fixed by running the SQL below. This will also allow for Workflows of up to 4GB as opposed to just 64k

```
alter table jiraworkflows change DESCRIPTOR DESCRIPTOR LONGTEXT;
```

JIRA 3.1.1 Release Notes

JIRA 3.1.1 Release Notes

In the tradition of frequent and worthwhile updates, JIRA 3.1.1 is released today in Standard, Professional and Enterprise editions. This point release includes over 40 bug fixes along with a number of improvements and new features. It can be downloaded here. See the JIRA 3.1 Upgrade Notes before upgrading.

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

New Features, Improvements and Bug Fixes

This release includes a number of new features and improvements - including:

- Ability to specify custom field values while creating an issue via XML-RPC/SOAP
- Improved translations for various locales.
- New custom field - DateTime
- Fixed portlet hyperlinks

JIRA 3.1.1 also includes over 40 Bug Fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

Important Version-Specific Upgrade Notes

The procedure for upgrading JIRA changes little from one JIRA version to the next, although process has been getting easier for recent major releases of JIRA. Despite this, each major JIRA release comes with specific recommendations for upgrading from the previous major version.

If you plan to skip a few JIRA versions for your next JIRA upgrade, then ***we strongly recommend*** reading the upgrade notes for all major versions between your current version and the version to which you are upgrading. These version-specific upgrade notes contain important information that may be relevant to your current JIRA installation and the JIRA version you are upgrading to.

For example: If you plan to upgrade from JIRA 4.1 to JIRA 4.4, then read the upgrade notes/guides for JIRA 4.2 and JIRA 4.3, as well as those for JIRA 4.4.

We also recommend that you read the upgrade notes for any minor releases listed below, since these notes contain important information that may be relevant to your JIRA upgrade.

From JIRA 4.4, the names of these documents were changed to 'upgrade notes' (from 'upgrade guides') to avoid potential naming ambiguity with the general Upgrading JIRA guide and for consistency with other Atlassian products.

Below is a list of upgrade notes/guides for all previous major and minor releases of JIRA, each of which has important information you should be aware of for your JIRA upgrade:
Upgrade notes/guides for minor releases are indented.

- JIRA 5.0 Upgrade Notes
- JIRA 4.4 Upgrade Notes
    - JIRA 4.4.4 Upgrade Notes
    - JIRA 4.4.3 Upgrade Notes
    - JIRA 4.4.1 Upgrade Notes
- JIRA 4.3 Upgrade Guide
- JIRA 4.2 Upgrade Guide
    - JIRA 4.2.2 Upgrade Guide
- JIRA 4.1 Upgrade Guide
    - JIRA 4.1.2 Upgrade Guide
    - JIRA 4.1.1 Upgrade Guide

- JIRA 4.0 Upgrade Guide
  - JIRA 4.0.1 Upgrade Guide
  - JIRA 4.0.2 Upgrade Guide
- JIRA 3.13 Upgrade Guide
  - JIRA 3.13.1 Upgrade Guide
- JIRA 3.12 Upgrade Guide
- JIRA 3.11 Upgrade Guide
- JIRA 3.10 Upgrade Guide
- JIRA 3.9 Upgrade Guide
- JIRA 3.8 Upgrade Guide
  - JIRA 3.8.1 Upgrade Guide
- JIRA 3.7 Upgrade Guide
  - JIRA 3.7.2 Upgrade Guide
- JIRA 3.6 Upgrade Guide
  - JIRA 3.6.2 Upgrade Guide
- JIRA 3.5 Upgrade Guide
  - JIRA 3.5.2 Upgrade Guide
- JIRA 3.4 and 3.4.1 Upgrade Guide
- JIRA 3.3 Upgrade Guide
  - JIRA 3.3.3 Upgrade Guide
  - JIRA 3.3.1 Upgrade Guide
- JIRA 3.2 Upgrade Guide
- JIRA 3.1 Upgrade Notes
- JIRA 3.0 Upgrade Notes

You will find the upgrade notes attached to the release notes for the relevant version.

RELATED TOPICS

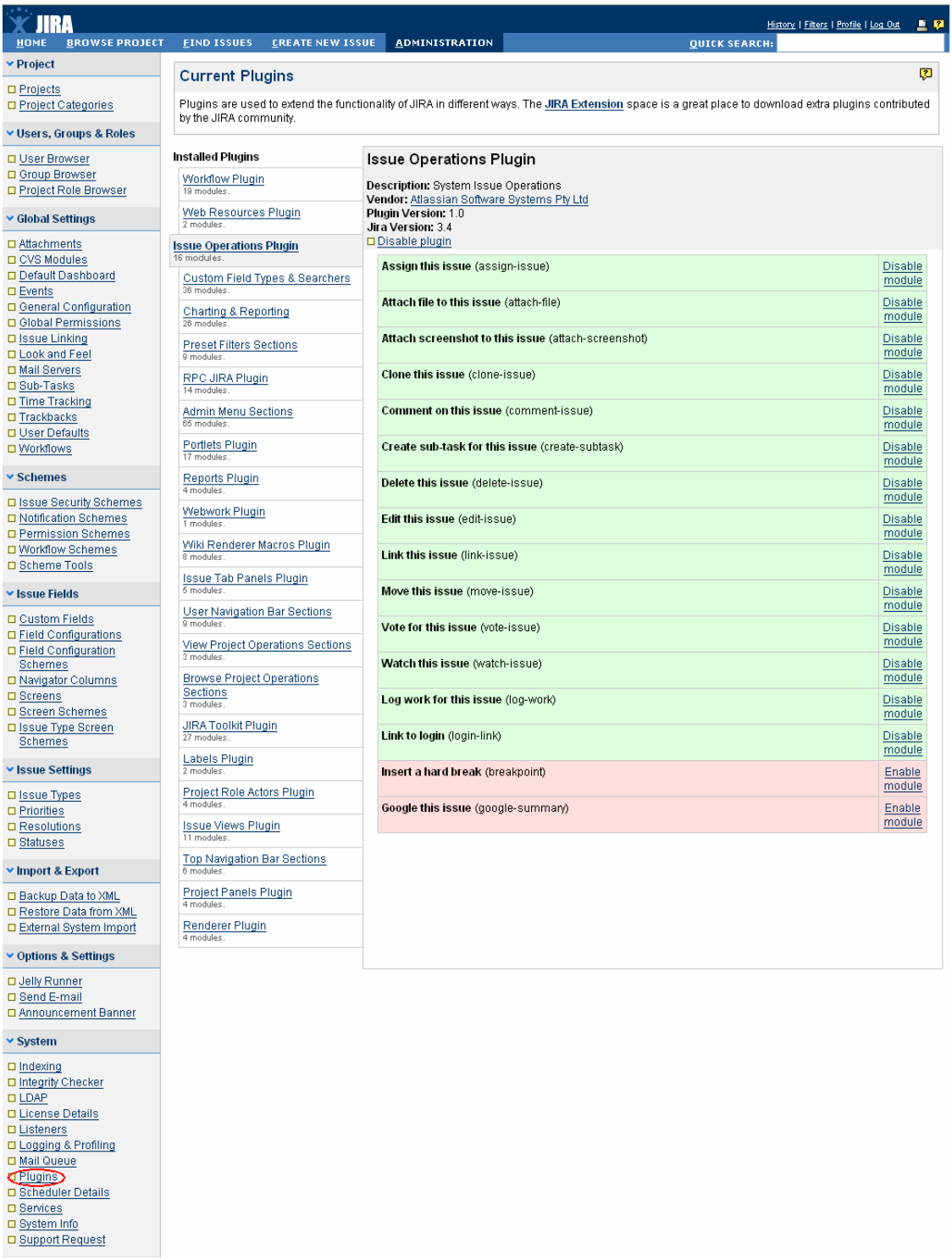
Production Releases
JIRA Releases

JIRA 3.7 Release Notes

Error: RuntimeException occurred while performing an XHTML storage transformation (null)

Issue Operations plugin

To remove an issue operations link (e.g. **Assign this issue**), simply navigate to 'Plugins' (in the 'Adminstration' menu) and click 'Disable module':



JIRA 3.7 Upgrade Guide

Once you have upgraded to JIRA 3.7, downgrading to a previous version is not a straightforward task and is not recommended.

JIRA 3.7 Upgrade Notes

This page lists a few things to be aware of when upgrading from previous releases of JIRA to JIRA 3.7. To perform the actual upgrade, see the upgrade documentation.

Note: If you are upgrading from a pre-3.6.5 release, please also refer to the relevant JIRA 3.x Upgrade Guides.

Please note that JIRA 3.7 requires JDK 1.4 or above. Support for JDK 1.3 has been discontinued.

Please note that some new functionality will not be available if you are running JIRA on WebLogic or Orion. The List All Filtersportlet will not be able to fetch the issue counts for each issue. The new 'Charting' View will also be unavailable. The support for WebLogic and Orion will be added in JIRA 3.7.1.

Database Schema Changes

Due to the upgrade of HSQLDB, and to improve compatibility with Firebird and Frontbase, various database tables and columns have been renamed. For more details on the changes please see the JIRA 3.7 Database Schema Changes document.

Therefore, the easiest way to upgrade to JIRA 3.7 is to follow the Upgrading JIRA safely instructions.

If in the past, instead of performing an XML backup and restore, you have been upgrading by "pointing" new version of JIRA at an old database, this is still possible, however the procedure is more complicated. You will need to use SQL scripts to perform database schema changes. For more information please see the SQL Scripts for 3.6.x to 3.7 schema upgrade document.

If you are using HSQLDB with JIRA, you must follow Upgrading JIRA safely instructions (i.e. perform a full XML backup and restore from XML), as simply copying the `.script` file will not work. The format of the `.script` file has changed between the HSQLDB versions, and therefore, copying the `.script` file will result in the following error on startup.

Request Context Changes

In order for plugins, customfields and portlets to function better outside of a web-context (e.g.: displaying a customfield in an e-mail), all direct references to the HttpServletRequest have been replaced by a VelocityRequestContext. If you have deployed your own plugins, customfields or portlets that use the HttpServletRequest directly (i.e.: any references to `${req}`) than they should be changed to use the new `${requestContext}` object. The `${requestContext}` is an implementation of the `VelocityRequestContext` interface.

Currently the `${requestContext}` supports the following properties:

- `${requestContext.baseUrl}` - Returns the same as HttpServletRequest.getContextPath() or the base URL configured in your JIRA instance if no HttpServletRequest is available
- `${requestContext.requestParameters}` - Returns an implementation of `RequestContextParameterHolder` or null if no HttpServletRequest is available
- `${requestContext.requestParameters.servletPath}` - Returns the same as HttpServletRequest.getServletPath()
- `${requestContext.requestParameters.requestURL}` - Returns the same as HttpServletRequest.getRequestURL()
- `${requestContext.requestParameters.queryString}` - Returns the same as HttpServletRequest.getQueryString()

Integrity Checks

In JIRA 3.7 Database Integrity Checks (available from the Administration section) have been re-written to run as multiple transactions, which increased the throughput of the system while the checks are running. In large JIRA 3.6 (and earlier) installations, integrity checks could cause database lock escalation and stop users from performing operations (e.g. viewing issues).

Please note, that due to the change, each integrity check became about 10% slower.

As integrity checks are quite database intensive operations, it is still recommended to run them during off-peak hours (i.e. while the system is not under heavy load).

Change of commentLevel to groupLevel in the Comment and TransitionWorkflow jelly tags

We have changed the AddComment and TransitionWorkflow jelly tag attribute that specifies the group visibility level from 'commentLevel' to be 'groupLevel'. If you have existing jelly tags that use this attribute it will need to change. This was done so that we could introduce the 'roleLevel' attribute which allows you to specify a project role based visibility. Only one of the two attributes can be specified at a time.

Change of level to grouplevel in the XML view of a Comment

1. We have changed the XML view of a comment, as seen in the XML view of an Issue to contain either a 'grouplevel' attribute or a 'rolelevel' attribute. This attribute defines the visibility level specified on the comment. In the past the 'grouplevel' attribute was simply 'level'. If you have any existing custom code that expects the 'level' attribute in the Comment XML it must change to expect 'grouplevel'.
2. In previous versions of JIRA the XML view of the <comment> tag level attribute was always shown, even if there was no value for the level, it was rendered as an empty attribute. We have changed it so that the attributes themselves (grouplevel and rolelevel) do not display if there is no value.

Change to the RemoteComment object used via SOAP/RPC plugin

The RemoteComment object and therefore the remote SOAP/RPC api has changes to almost all properties. The 'roleLevel' attribute was added and the following attributes have changed:

1. level -> groupLevel
2. datePerformed -> created
3. username -> author

ActionManager removed

The `ActionManager` interface has been removed and its functionality has been delegated to new interfaces. For details please refer to ActionManager Removed documentation

Removal of 'Backend Actions'

1. We have removed the 'Backend Action' `com.atlassian.jira.action.action.WorklogCreate` if you were using this class in a plugin or custom code you will now need to use the `com.atlassian.jira.issue.worklog.WorklogManager` this now has method calls to return worklogs for a given user+issue and also create worklog entries.
2. We have removed the 'Backend Action' `com.atlassian.jira.action.action.ActionCreate` if you were using this class to create comments you will need to modify your code to use one of the create methods on the com.atlassian.jira.bc.issue.comment.CommentService

Issue Events

We have modified the `com.atlassian.jira.event.issue.IssueEvent` class to no longer use GenericValues. The GenericValue representing the comment is replaced by `com.atlassian.jira.issue.comments.Comment` class and the GenericValue representing the worklog is replaced by `com.atlassian.jira.issue.worklog.Worklog` class. If you have a custom listener in a previous version of JIRA this will need to be updated to use the newer IssueEvent class and `com.atlassian.jira.event.issue.IssueEventDispatcher.dispatchEvent(...)` methods.

Renaming of XML export file

By popular request, the XML filename (that is, the default filename when you choose to save the XML view in the Issue Navigator) has been changed from `issuenavigator.jspa` to `SearchRequest.xml`. Should you have any external systems or programs that utilise the exported XML file, please be aware of the changed filename.

Confluence Users Only - Pre 2.2.10 Confluence Must Be Patched To Use JIRA Issues Macro

Unable to render {include} The included page could not be found.

JIRA 3.7 Downgrade Notes

Once you have upgraded to JIRA 3.7, downgrading to a previous version is not a straightforward task and is not recommended. Please be aware that in JIRA 3.7 the database schema has changed.

If upgrade to JIRA 3.7 fails, the best way to proceed is to go back to the previous version of JIRA you were using, and to the latest **pre-upgrade** data that you have. The exact steps for doing this depend on how you have upgraded JIRA.

If you have created a new database for JIRA 3.7 by following the Upgrading JIRA safely instructions, you should be able to simply shutdown JIRA 3.7 and bring up the old version of JIRA your were using. The old version should be configured to use its old (unupgraded) database.

If you have upgraded JIRA by pointing JIRA 3.7 to an older database (and ran the SQL Scripts to upgrade the database schema), then you will need to:

1. Create a new database
2. Configure the old version of JIRA you were using to point at the new (empty) database
3. Restore the latest **pre-upgrade** backup that you have
4. Start the old JIRA installation

ActionManager Removed

From JIRA 3.7, the ActionManager has been refactored into several other interfaces, these include the CommentManager, WorklogManager, ChangeHistoryManager, RepositoryManager and IssueTabPanel. The following table is a mapping of the old ActionManager methods to the new refactored ones (including the new java interface the method resides in).

You will notice that the new methods in JIRA 3.7 take in the Issue object as opposed to the GenericValue.

ActionManager method (Pre JIRA 3.7)	Corresponding method (Post JIRA 3.7)	Corresponding Interface
List **getComments**(GenericValue issue, User user)	List **getCommentsForUser**(Issue issue, User user)	CommentManager
List **getWorklog**(GenericValue issue, User user)	List **getWorklogsForUser**(Issue issue, User user)	WorklogManager
List **getChangeHistory**(GenericValue issue, User remoteUser)	List **getChangeHistoriesForUser**(Issue issue, User remoteUser)	ChangeHistoryManager
List **getWorkflow**(GenericValue issue, User remoteUser)	removed	-
List **getActions**(GenericValue issue, User remoteUser)	List **getActions**(Issue issue, User remoteUser)	AllTabPanel
List **getCommits**(GenericValue issue, User remoteUser)	Map **getCommits**(Issue issue, User remoteUser)	RepositoryManager
Email **createEmail**(GenericValue issue, Message mimeMessage)	removed	-
List **getEmails**(GenericValue issue, User remoteUser)	removed	-
List **getPluginModuleActions**(String moduleKey, GenericValue issue, User remoteUser)	List **getActions**(Issue issue, User remoteUser)	IssueTabPanel

JIRA 3.7 Database Schema Changes

Due to the upgrade of HSQLDB, and to improve compatibility with Firebird and Frontbase, various database tables and columns have been renamed.

The table below summarises the changes to the database schema. Please note, that if you have developed any custom utilities which query or modify the JIRA database directly (i.e. without using the JIRA API), please check whether the utilities need to be updated.

Tables

The following database table has been renamed:

OLD TABLE NAME	NEW TABLE NAME
version	projectversion

Columns

The following database columns have been renamed. Their old and new names, as well as the database table they belong to are shown below:

TABLE NAME	OLD COLUMN NAME	NEW COLUMN NAME

notification	type	notif_type
notification	parameter	notif_parameter
mailserver	type	server_type
jiraeventtype	type	event_type
schemepermissions	type	perm_type
schemepermissions	parameter	perm_parameter
fieldlayout	type	layout_type
schemeissuesecurities	type	sec_type
schemeissuesecurities	parameter	sec_parameter
portletconfiguration	position	positionseq

Special note for MS SQL Server

Additionally to the schema changes described above, for MS SQL Server, all columns of type `TEXT` have been changed to `NTEXT` to ensure that international characters can be safely stored.

Possible upgrade problems + solutions

HSQL DB Upgrade

JIRA 3.7 has been upgraded to use HSQL DB version 1.8. HSQL DB is the in-memory database that ships with the Standalone distribution of JIRA.

If you have upgraded to JIRA 3.7 and have tried to copy across the HSQL DB script files from your 3.6.x or earlier instance you will see an error like the following:

```
2006-09-20 16:33:49,858 [core.entity.jdbc.DatabaseUtil] Unable to esablish a connection with
the database... Error was:org.apache.tomcat.dbcp.dbcp.SQLNestedException: Cannot create
PoolableConnectionFactory (error in script file line: 104 Unexpected token: POSITION in
statement [CREATE TABLE PORTLETCONFIGURATION(ID BIGINT NOT NULL PRIMARY KEY,PORTALPAGE
BIGINT,PORTLET_ID VARCHAR,COLUMN_NUMBER INTEGER,POSITION])
 2006-09-20 16:33:49,858 [core.entity.jdbc.DatabaseUtil] Could not get table name information
from the database, aborting.
 2006-09-20 16:33:51,493 WARN [jira.config.properties.ApplicationPropertiesImpl] Exception
getting property 'jira.extension.container.provider' from database. Using default
 2006-09-20 16:33:51,729 ERROR [ContainerBase.[Catalina].[localhost].[/]] Exception sending
context initialized event to listener instance of class
com.atlassian.jira.upgrade.ConsistencyLauncher
 com.opensymphony.module.propertyset.PropertyImplementationException: Unable to esablish a
connection with the database. (Cannot create PoolableConnectionFactory (error in script file
line: 104 Unexpected token: POSITION in statement [CREATE TABLE PORTLETCONFIGURATION(ID BIGINT
NOT NULL PRIMARY KEY,PORTALPAGE BIGINT,PORTLET_ID VARCHAR,COLUMN_NUMBER INTEGER,POSITION]))
        at
com.opensymphony.module.propertyset.ofbiz.OFBizPropertySet.getKeys(OFBizPropertySet.java:100)
        at
com.opensymphony.module.propertyset.AbstractPropertySet.getKeys(AbstractPropertySet.java:292)
        at
com.opensymphony.module.propertyset.PropertySetCloner.cloneProperties(PropertySetCloner.java:114)
at com.opensymphony.module.propertyset.PropertySetManager.clone(PropertySetManager.java:61)
        at
com.atlassian.jira.propertyset.JiraCachingPropertySet.init(JiraCachingPropertySet.java:411)
        at
com.opensymphony.module.propertyset.PropertySetManager.getInstance(PropertySetManager.java:45)
        at
com.opensymphony.module.propertyset.PropertySetManager.getInstance(PropertySetManager.java:22)
        at
com.atlassian.jira.config.properties.PropertiesManager.loadPropertySet(PropertiesManager.java:79)
```

```
at com.atlassian.jira.config.properties.PropertiesManager.<init>(PropertiesManager.java:25)
        at
com.atlassian.jira.config.properties.PropertiesManager.getInstance(PropertiesManager.java:35)
        at
com.atlassian.jira.config.properties.ApplicationPropertiesImpl.getPropertiesManager(ApplicationProp
at
com.atlassian.jira.config.properties.ApplicationPropertiesImpl.getStringsWithPrefix(ApplicationProp
at com.atlassian.jira.plugin.JiraPluginStateStore.loadPluginState(JiraPluginStateStore.java:41)
        at com.atlassian.plugin.DefaultPluginManager.<init>(DefaultPluginManager.java:29)
        at com.atlassian.jira.plugin.JiraPluginManager.<init>(JiraPluginManager.java:28)
        at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
        at
sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:39)
        at
sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:27
at java.lang.reflect.Constructor.newInstance(Constructor.java:494)
        at
org.picocontainer.defaults.InstantiatingComponentAdapter.newInstance(InstantiatingComponentAdapter.
at
org.picocontainer.defaults.ConstructorInjectionComponentAdapter.instantiateComponent(ConstructorInj
at
org.picocontainer.defaults.InstantiatingComponentAdapter.getComponentInstance(InstantiatingComponen
at
org.picocontainer.defaults.DecoratingComponentAdapter.getComponentInstance(DecoratingComponentAdapt
at
org.picocontainer.defaults.SynchronizedComponentAdapter.getComponentInstance(SynchronizedComponentA
at
org.picocontainer.defaults.DecoratingComponentAdapter.getComponentInstance(DecoratingComponentAdapt
at
com.atlassian.jira.config.component.ProfilingComponentAdapter.getComponentInstance(ProfilingCompone
at
org.picocontainer.defaults.DefaultPicoContainer.getComponentInstance(DefaultPicoContainer.java:298)
at com.atlassian.jira.ComponentManager.quickStart(ComponentManager.java:267)
        at com.atlassian.jira.ComponentManager.start(ComponentManager.java:254)
        at
com.atlassian.jira.upgrade.ConsistencyLauncher.contextInitialized(ConsistencyLauncher.java:25)
        at org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3669)
        at org.apache.catalina.core.StandardContext.start(StandardContext.java:4104)
        at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
        at org.apache.catalina.core.StandardHost.start(StandardHost.java:718)
        at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
        at org.apache.catalina.core.StandardEngine.start(StandardEngine.java:442)
        at org.apache.catalina.core.StandardService.start(StandardService.java:450)
        at org.apache.catalina.core.StandardServer.start(StandardServer.java:683)
        at org.apache.catalina.startup.Catalina.start(Catalina.java:537)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
        at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:585)
```

```
        at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:271)
        at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:409)
```

If you see this error, JIRA will not be able to startup.

Solution

To resolve this problem you will need to 'safely' upgrade JIRA. The detailed instructions for doing this can be found in JIRA's online documentation.

A quick guide on doing this follows:

1. Startup your old version of JIRA, pointing it at your current HSQL DB scripts files
2. Perform an XML backup of JIRA
3. Shutdown you old 3.6.x JIRA instance.
4. Startup your new 3.7 instance of JIRA using a new script location

```
<Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"
 username="sa"
 password=""
 driverClassName="org.hsqldb.jdbcDriver"
 url="jdbc:hsqldb:${catalina.home}/database/jiradb37"
 minEvictableIdleTimeMillis="4000"
 timeBetweenEvictionRunsMillis="5000"
/>
```

5. Perform an XML import

SQL Scripts for 3.6.x to 3.7 schema upgrade

Audience	People who are upgrading from pre-JIRA 3.6.x to post-3.7, and cannot use the recommended 'XML backup/restore' upgrade method because it would take too long.

If you are upgrading JIRA by the recommended method, ignore this page.

Background

In general, there are two ways to upgrade JIRA's database, both of which are described in the Upgrading JIRA document:

- **XML backup/restore (recommended)** — doing a full XML export and import into the new database.
- **Connect JIRA to a copy of your old database** — connecting the new JIRA to the old database, and letting it automatically upgrade the database tables.

In 3.7.x, **the 'Connect JIRA to a copy of your old database' method will not work**. If you point JIRA 3.7.x to an older database, JIRA will print a warning and refuse to do anything. This is because a large number of database schema changes were made between 3.6 and 3.7 (see JIRA 3.7 Database Schema Changes) and these changes are too great for JIRA's database engine to upgrade automatically.

So, the **vast majority of users** should follow the XML backup/restore method described in the Upgrading JIRA document.

However, there are a handful of users with large installations, for which a full export/import is impractical as it takes a relatively long time. These users may need to use the 'Connect JIRA to a copy of your old database' method — **This page is intended for these users.** We have provided SQL scripts for each database, which will make the required changes to a 3.6.x database so it can be upgraded without a full export/import.

These scripts will only work on **JIRA 3.6.x** databases (they refer to a table only added in 3.6). If you are upgrading from an earlier release, please:

1. Download JIRA 3.6.5 Standalone
2. Back-up your database, and create a copy to be upgraded to 3.7.
3. Configure it to point to your 3.7 copy of the database
4. (optionally) Edit `atlassian-jira/WEB-INF/classes/jira-application.properties` and set `jira.autoexport=false` to speed up the process.
5. Start JIRA Standalone. By watching the logs (`atlassian-jira.log` or `logs/catalina.out`, you will see JIRA automatically upgrading tables to the 3.6.x format.
6. Proceed with the instructions below.

If you are using HSQLDB with JIRA, you **must** follow the 'XML backup/restore' instructions in the Upgrading JIRA guide, as simply copying the `.script` file **will not work**. The format of the `.script` file has changed between the HSQLDB versions, and therefore, copying the `.script` file will result in the following error on startup.

DB2 upgrade notes

1. Shutdown your JIRA instance
2. Perform a backup of your DB2 database: **db2 backup database sample to /home/db2/backups**
3. Download the following script db2_3.7_migration.ddl and modify the connect statement within the file
4. Execute the script using the following command: **db2 +c -t -v -f db2_3.7_migration.ddl**
   If you see errors like:

```
insert into SCHEMEISSUESECURITIES (select ID, SCHEME, SECURITY, TYPE, PARAMETER from
TMP_SCHEMEISSUESECURITIES)
SQL0100W  No row was found for FETCH, UPDATE or DELETE; or the result of a
query is an empty table.  SQLSTATE=02000
```

   This is OK, it simply means that the inner SQL query did not return any data to be inserted into the new temporary table. This can occur if you are not using certain features in JIRA.
5. Point your new instalation of JIRA 3.7 at your DB2 database and watch for any errors during the statup sequence.
6. If you see any other errors please contact support for further assistance.

PostgreSQL 8+ upgrade notes

1. Shutdown your JIRA instance
2. Perform a backup of your PostgreSQL database **pg_dump -d *database name* > *backup filename*.sql**
   - for example **pg_dump -d jiradb > jiradb_365_14112006.sql**
3. Download the following script postgres_3.7_migration.sql
4. Execute the script using the following command: **psql -Ujirauser -v schema_name=public -d jiradb -f postgres_3.7_migration.sql**
   a. Where -U is the username, -v is the name of your schema -d is the database -f is the location of the script file
5. Point your new instalation of JIRA 3.7 at your database and watch for any errors during the startup sequence.
6. If you see any errors please contact support for further assistance.

Oracle 10g upgrade notes

1. Shutdown your JIRA instance
2. Perform a backup of your Oracle database. There are multiple strategies here, so we will leave this up to your DBA.
3. Download the following script oracle_3.7_migration.sql
4. Connect to SQL*Plus and execute the following script:

```
SQL*Plus: Release 10.2.0.1.0 - Production on Thu Oct 19 12:56:11 2006
Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> @/home/oracle/oracle_3.7_migration.sql
```

5. If you see any errors please contact support for further assistance.
6. Point your new instalation of JIRA 3.7 at your DB2 database and watch for any errors during the statup sequence.

Microsoft SQL Server upgrade notes

1. Shutdown your JIRA instance
2. Perform a backup of your SQL Server database: **osql -U *username* -P *password* -Q "BACKUP DATABASE *db_name* TO DISK = *backup_path_and_filename*"**
    - for example **osql -U sa -P secret -Q "BACKUP DATABASE jiradb TO DISK = 'C:\MyBackup.dat'"**

1. Download the following script sqlserver_3.7_migration.sql
2. Execute the script: **osql -U *username* -P *password* -d *db_name* -i mssql_3.7_migration.sql**
    - for example **osql -U sa -P secret -d jiradb -i sqlserver_3.7_migration.sql**
3. If everything goes well the following should be displyed

```
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'NOTIF_TYPE'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'NOTIF_PARAMETER'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'SEVER_TYPE'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'EVENT_TYPE'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'PERM_TYPE'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'PERM_PARAMETER'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'LAYOUT_TYPE'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'SEC_TYPE'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'SEC_PARAMETER'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The COLUMN was renamed to 'POSITIONSEQ'.
Caution: Changing any part of an object name could break scripts and stored procedures.
The object was renamed to 'projectversion'.
```

4. Point your new instalation of JIRA 3.7 at your SQL Server database and watch for any errors during the statup sequence.

Sybase upgrade notes

1. Shutdown your JIRA instance
2. Perform a backup of your SQL Server database
    - for example using **isql** tool

```
1> dump database db_name to "backup_path_and_filename"
2> go
```

3. Download the following script sybase_3.7_migration.sql
4. Execute the script: **osql -U *username* -P *password* -D *db_name* -i sybase_3.7_migration.sql**
    - for example **osql -U sa -P -D jiradb -i sybase_3.7_migration.sql**
5. If everything goes well the following should be displyed

```
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Column name has been changed.
(return status = 0)
Object name has been changed.
(return status = 0)
```

6. Point your new instalation of JIRA 3.7 at your Sybase database and watch for any errors during the statup sequence.

MySQL upgrade notes

1. Shutdown your JIRA instance
2. Perform a backup of your MySQL database: **mysqldump --opt *db_name* > *db_name*.sql**
    - for example **mysqldump --opt jiradb > jiradb_before37.sql**
3. Download the following script mysql_3.7_migration.sql
4. Execute the script: **mysql --user=*username* --password=*password db_name* < mysql_3.7_migration.sql**
    - for example **mysql --user=root --password=password jiradb < mysql_3.7_migration.sql**
5. Point your new installation of JIRA 3.7 at your MySQL database and watch for any errors during the statup sequence.

Attachments

Name	Size	Creator	Creation Date	Comment
File sqlserver_3.7_migration.sql	12 kB	Jed Wesley-Smith	Mar 22, 2007 00:36	
File postgres_3.7_migration.sql	0.9 kB	Justin Koke	Oct 18, 2006 23:08	
File oracle_3.7_migration.sql	0.7 kB	Justin Koke	Oct 18, 2006 23:08	
File db2_3.7_migration.ddl	4 kB	Justin Koke	Oct 18, 2006 20:46	
File mysql_3.7_migration.sql	0.7 kB	Dylan Etkin	Sep 20, 2006 01:02	
File sybase_3.7_migration.sql	0.5 kB	Dylan Etkin	Sep 20, 2006 01:02	

JIRA 3.7.4 Release Notes

JIRA 3.7.4 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Atlassian Software Systems is proud to announce the release of JIRA 3.7.4 in Standard, Professional and Enterprise editions. This point release includes 31 bug fixes and improvements, notably:

- **JIRA startup** — speed has been improved for Oracle - JRA-12049
- **Project roles** — a few minor bug fixes.
- **Getting ready for Crowd** — updates to JIRA infrastructure in preparation for integrating with Crowd in JIRA 3.8.

JIRA 3.7.4 can be downloaded here, and of course is free to all customers who purchased their JIRA licence or maintenance within the last 12 months.

If upgrading, please refer to the JIRA 3.7.4 Upgrade Guide.

Not using 3.7? Learn about all the new features you're missing out on!

Weblogic Users

Please note that there is a known Weblogic and Firefox issue that will affect JIRA 3.7.x when using Weblogic and Firefox. See the issue for more detail.

JIRA 3.7.4 includes the following bug fixes.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.7.4 Upgrade Guide

Upgrading from JIRA 3.7.3 to 3.7.4

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.7.2 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.7.3 Release Notes

JIRA 3.7.3 Release Notes

JIRA 5.0 has been released. Read the full JIRA 5.0 Release Notes and latest Upgrade Notes.

Recommended Upgrade

JIRA 3.7.3 contains a security update and is highly recommended, especially for instances of JIRA that are accessible via the Internet.

Atlassian Software Systems is proud to announce the release of JIRA 3.7.3 in Standard, Professional and Enterprise editions. This point release includes 15 bug fixes and improvements, notably:

- **Excel formatting** has been rectified (JRA-11928, JRA-8899).
- **French translations** have been rolled back (temporarily) to pre-3.7.1 (JRA-11973, JRA-11873).
- **Security Update** - a problem was fixed where, at worst, a malicious user could indirectly email (and obtain a list of) other users. No other project information could be leaked, apart from the users' emails. This bug affects all previous versions.

JIRA 3.7.3 can be downloaded here, and of course is free to all customers who purchased their JIRA licence or maintenance within the last 12 months.

If upgrading, please refer to the JIRA 3.7.3 Upgrade Guide.

Not using 3.7? Learn about all the new features you're missing out on!

Weblogic Users

Please note that there is a known Weblogic and Firefox issue that will affect JIRA 3.7.x when using Weblogic and Firefox. See the issue for more detail.

JIRA 3.7.3 includes the following 15 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.7.3 Upgrade Guide

Upgrading from JIRA 3.7.2 to 3.7.3

Please follow the JIRA general upgrade instructions.

Upgrading from JIRA 3.7.1 and earlier

In addition to the above, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

JIRA 3.7.2 Release Notes

JIRA 3.7.2 Release Notes

Recommended Upgrade

This upgrade is strongly recommended for all users of JIRA 3.7.x as it contains a fix for a critical search indexing issue. Any users of JIRA 3.7 or 3.7.1 should upgrade immediately.

Atlassian Software Systems is proud to announce the release of JIRA 3.7.2 in Standard, Professional and Enterprise editions. This point release includes 32 bug fixes and improvements, including:

- Most notably, 3.7.2 includes a fix for a critical search indexing issue, which makes it a **highly recommended upgrade** for JIRA 3.7 and 3.7.1 users.
- A few small bugs in the CSV importer have been fixed (JRA-11847, JRA-11842).
- Deleting role members in Sybase now works (JRA-11890).

JIRA 3.7.2 can be downloaded here, and of course is free to all customers who purchased their JIRA license or maintenance within the last 12 months.

Not using 3.7? Learn about all the great new features you're missing out on!

If upgrading, please read refer to the JIRA 3.7.2 Upgrade Guide.

Weblogic Users

Please note that there is a known Weblogic and Firefox issue that will affect JIRA 3.7.x when using Weblogic and Firefox. See the issue for more detail.

JIRA 3.7.2 includes the following 32 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.7.2 Upgrade Guide

JIRA 3.7.2 Upgrade Guide

This page contains specific information you need to know when upgrading from JIRA 3.7.1 to JIRA 3.7.2. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- 3.7.2 will automatically perform a full reindex when upgrading. For more details please see JRA-11861

JIRA 3.7.1 Release Notes

JIRA 3.7.1 Release Notes

Atlassian Software Systems is proud to announce the release of JIRA 3.7.1 in Standard, Professional and Enterprise editions. This point release includes 34 bug fixes and improvements, including many user interface and consistency fixes. It can be downloaded here.

Not using 3.7? Learn about all the **new features you're missing out on**!

If upgrading from an earlier version please read through the JIRA 3.7.1 Upgrade guide.

Weblogic Users

Please note that there is a known Weblogic and Firefox issue that will affect JIRA 3.7.x when using Weblogic and Firefox. See the issue for more detail.

JIRA 3.7.1 includes the following 34 bug fixes and improvements.

Error rendering macro 'jiraissues' : JIRA project does not exist or you do not have permission to view it.

JIRA 3.7.1 Upgrade guide

JIRA 3.7.1 Upgrade Guide

This page contains specific information you need to know when upgrading from JIRA 3.7 to JIRA 3.7.1. If upgrading from an older version of JIRA, please read the *Upgrade Guide* for each version your are skipping during the upgrade. The complete list of Upgrade Guides is available here.
When upgrading JIRA please follow the general upgrade instructions keeping in mind the information below.

- There are no specific instructions you need to be aware of related to upgrading from JIRA 3.7 to JIRA 3.7.1.

Beta Releases

The following Beta releases are currently available for download. Your help with testing them is very appreciated! Please log the bugs you find on http://jira.atlassian.com in the "JIRA" project.

Beta Releases

- JIRA 5.0 RC 3 Release Notes
- JIRA 5.0 RC 2 Release Notes
- JIRA 5.0 RC 1 Release Notes
- JIRA 5.0 Beta 3 Release Notes
- JIRA 5.0 Beta 2 Release Notes
- JIRA 5.0 Beta 1 Release Notes
- JIRA 4.4 RC 1 Release Notes
- JIRA 4.4 Beta 1 Release Notes
- JIRA 4.3 RC1 Release Notes
- JIRA 4.3 Beta 1 Release Notes
- JIRA 4.2 RC1 Release Notes
- JIRA 4.2 Beta 3 Release Notes
- JIRA 4.2 Beta 2 Release Notes
- JIRA 4.2 Beta 1 Release Notes
- JIRA 4.0 RC1 Release Notes
- JIRA 4.0 Beta 5 Release Notes
- JIRA 4.0 Beta 4 Release Notes
- JIRA 4.0 Beta 3 Release Notes
- JIRA 4.0 Beta 2 Release Notes
- JIRA 4.0 Beta 1 Release Notes

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

JIRA 5.0 RC 3 Release Notes

24 January 2012

JIRA 5.0 RC 3 (a.k.a 5.0 milestone 11 or 'm11') is a public development release leading up to **JIRA 5.0**. A **Release Candidate (RC)** is a preliminary release leading up to the official release of a JIRA version. RC releases are a fairly stable snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use RC releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 5.0 RC 3 release. Thank you for your feedback during the recent Beta and EAP releases and please keep providing it here.

Upgrading to JIRA 5.0 RC 3:

- JIRA EAP/Beta/RC releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

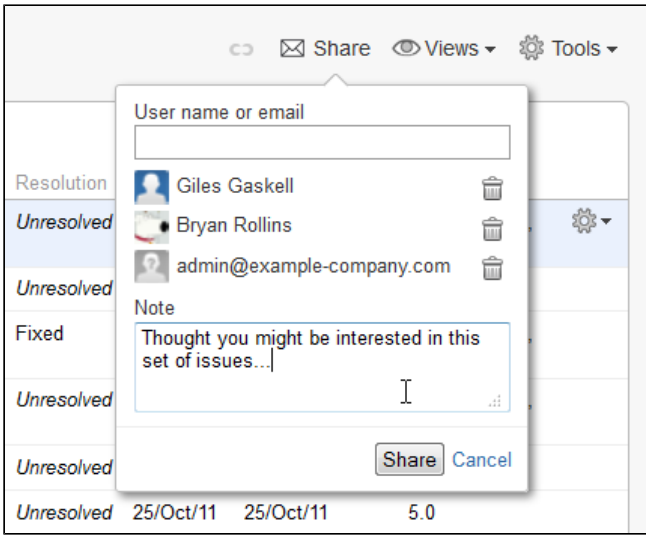
Overview

JIRA 5.0 connects people, teams and other applications together. JIRA 5.0 helps connect people and teams by allowing them to share issues and search results and mention other team members in an issue. People and teams are better connected to other applications with remote 'issue links' to other JIRA sites or web page URLs and activity streams that span all your linked Atlassian applications.

JIRA 5.0 also introduces a large number of improvements for developers to help connect JIRA to other applications, including the introduction of:

- New REST APIs to manage every facet of an issue,
- New remote 'issue link' and activity stream features, specifically for integration with other applications, and
- A stable Java API for JIRA.

Highlights:

Remote 'issue links'	Sharing issues and 'mentions'
	
• Create links to JIRA issues on another JIRA site • Create links to any URL • Create links via JIRA's Java or REST API	• Share issues quickly with other users • Mention a user in an issue comment • JIRA uses autocomplete whenever you 'share' or 'mention' a user

And More:

- Search for issues based on their history
- Activity streams now show activity from other applications
- Manage other users shared filters and dashboards
- Administration user interface improvements

- REST API (with tutorials) for working with issues in JIRA
- Stable Java API
- Performance improvements
- New troubleshooting and debugging tools
- New email handler wizard
- Enhancements to the 'view issue' page
- JIRA to JIRA issue copy

Thank you for your interest in JIRA 5.0 RC 3
Download RC 3

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights

1

Remote 'issue links'

The *remote 'issue links'* feature provides a powerful way to link JIRA issues to items external to your JIRA installation, residing on external applications.

Along with a Java and REST API to add these links, end users can also:

- Add an issue link from a JIRA issue to an issue on another JIRA site, including reciprocal links between these issues.
- Search for a Confluence page from a JIRA issue and add an issue link to that page.
- Add an issue link from a JIRA issue to any web page URL, such as a page of documentation, a technical note, or any other page on another web site.

Check out a live example here.

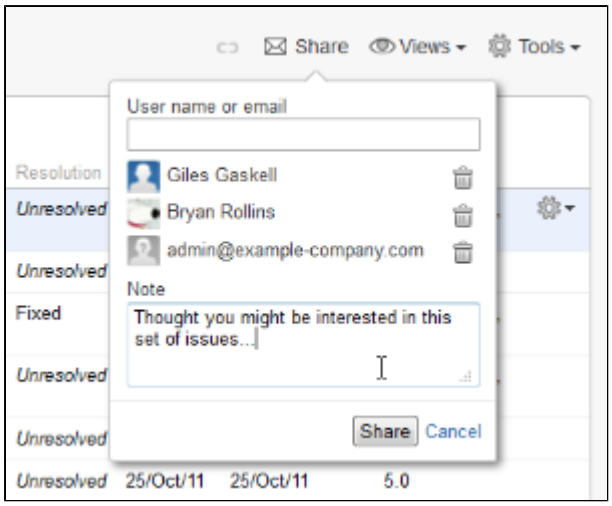
If you are a:

- JIRA user — see Linking Issues for details on using remote 'Issue Links' within JIRA.
- JIRA system administrator — see Configuring Issue Linking for details on how to make remote 'Issue Linking' available to your users by setting up the required application links between JIRA and other applications.
- JIRA developer — see JIRA Remote Issue Links on our developer documentation site.

^Top

2

Sharing issues and 'mentions'



Screenshot: Sharing a list of issues with other users

Screenshot: Mentioning a user in an issue comment

Need someone else to take a look at a JIRA issue or a list of issues?

- View any issue or a list of issues on the issue navigator, click the **Share** button at the top-right (or type **s**) and specify JIRA users (based on their names or user names) or *any* email address of people you want to share the issue with.

  Recipients will be emailed a link to the issue (or a list of issues 'shared' via the issue navigator).

- Mention other JIRA users in an issue's **Description** or **Comment** field when creating, editing or commenting on an issue.

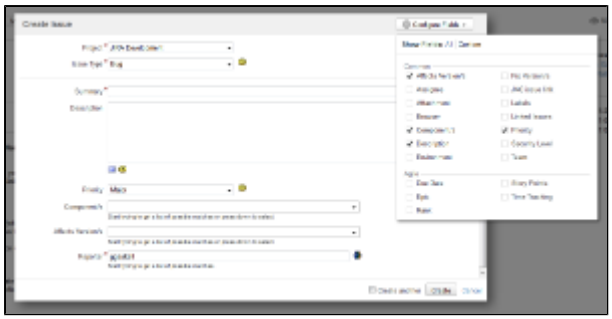
  JIRA users will receive details of the issue in an email message (sent to the addresses registered with their user accounts). The message's subject line will indicate that the person who used this feature 'mentioned' them on that issue.

Please Note:

- Users required the **Browse Users** global permission to access the **Share** button or the autocomplete feature when 'mentioning' a user. However, if you know the username of a JIRA user, you can still mention them.
- JIRA system administrators will need to configure JIRA's outgoing SMTP mail server for the 'share' and 'mentions' features to work.

^Top

Rapidly create and edit issues



JIRA 5.0 lets you create and edit issues and sub-tasks much faster. Creating and editing is now performed in a dialog box (rather than a separate form), so that you do not need to leave the current page or have it reload just to start creating or editing an issue or sub-task.

You can customise fields on the **Create Issue**/**Edit Issue** dialog boxes by removing or adding fields fields through the **Configure Fields** button. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly used fields whenever you create or edit an issue.

The **Create Issue** dialog box allows you to rapidly create a series of related issues with similar options. When you select the **Create another** check box before clicking the **Create** button, JIRA creates your issue and automatically pre-populating a new **Create Issue** dialog box with your previous field values, whilst leaving the **Summary** field blank. Note that this feature does not carry across any attachments that were attached to your previously created issue.

The **Assignee**, **Project** and **Issue Type** fields use 'autocompletion' too. Hence, you no longer a need to scroll through a whole raft of items to specify these fields when creating or editing an issue.

You can easily access this feature by typing '**c**' to access the create issue dialog box or '**e**' from a currently selected issue on the issue navigator or 'view issue' page to access the edit dialog box for that issue.

^Top

Search for issues based on their history

"CHANGED" operator introduced

Introduced in JIRA 4.4.3, JQL's "CHANGED" operator can accept the optional predicates **FROM**, **TO**, **ON**, **DURING**, **BEFORE**, **AFTER** and **BY**, and can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

For example, this link shows all the issues logged against the JIRA project on our 'jira.atlassian.com' site, whose **Fix Version** field was changed to "5.0".

You can also create more complex JQL queries with the "CHANGED" operator by fine-tuning them with predicates. For example, the following JQL query:

```
status changed FROM "In QA Review" TO "QA Rejected" BY freddo AFTER startOfWeek() BEFORE
endOfWeek()
```

will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo' between the start and end of the current week.

You can use complex queries such as these to generate the 'Single Level Group By Report' in the screenshot above, which shows grouping by 'Team'.

"WAS" operator enhanced

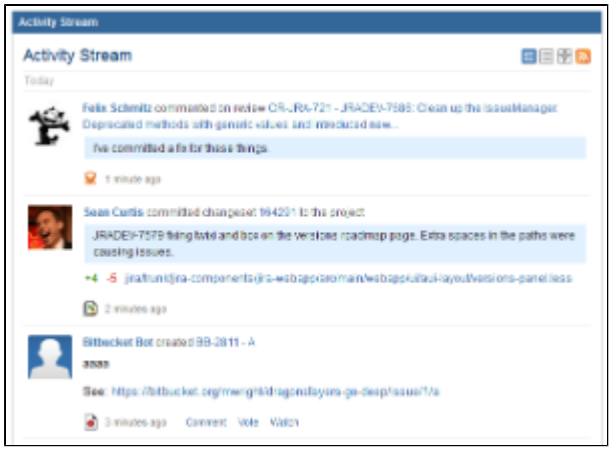
Also introduced in JIRA 4.4.3 was the ability of the "WAS" operator to work with the **Fix Version** field. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

^Top

Activity streams now show activity from other applications



Screenshot: Activity stream gadget showing activity from other Atlassian applications

Screenshot: Activity stream gadget showing activity from non-Atlassian applications

We have expanded the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

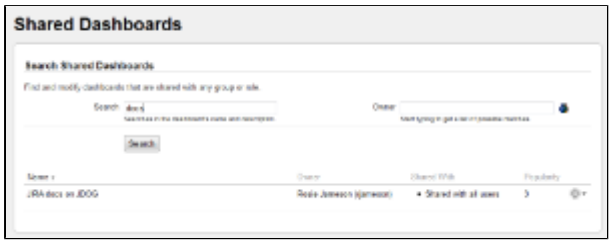
With these new features, you can:

- See Confluence page updates from your activity streams in JIRA, then drill down into those Confluence pages for more information

- See updates from another JIRA site. For example, activity streams on your development team's JIRA site (behind the firewall) can include activity on your support team's customer facing JIRA site.

^Top

Manage other users' shared filters and dashboards



Introduced in JIRA 4.4.1, JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboard. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

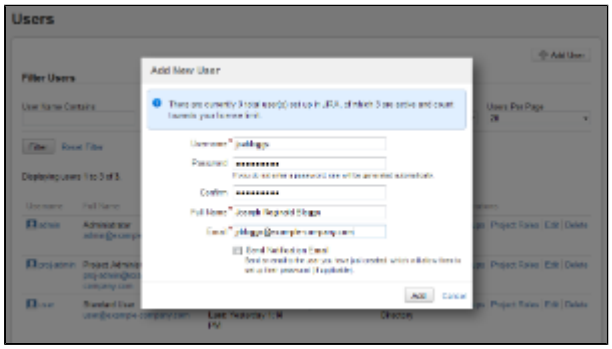
You can access these features by selecting **Administration > Users > Shared Filters** or **Shared Dashboards** (or using the keyboard shortcut **g + g +** start typing **shared filters** or **shared dashboard**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

^Top

Administration user interface improvements



Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 provides further improvements to the

Administration UI by converting forms on various Administration pages to convenient dialog boxes.

For example, the form for adding users is now a dialog box, which is accessed by clicking **Add User** on the **Users** page of JIRA's Administration area.

In addition to the dialog box for adding a new user, the 'Attachments' and 'Workflows' pages have been redesigned and the forms associate with these pages have been converted into convenient dialog boxes too.

^Top

REST API (with tutorials) for working with issues in JIRA

JIRA's REST API has undergone a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues and editing existing ones.
- Delete existing issues and their subtasks.
- Create remote 'issue links'.
- Retrieve metadata from your favourite filters and dashboards.
- Retrieve metadata about your permissions.
- Almost all system fields and JIRA's built-in custom field types are supported.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2' (or '`latest`' to use the latest REST API version available with your version of JIRA).

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

You might also want to try out Atlassian's new REST API Browser, which is available in the Atlassian Plugin SDK and can be accessed by adding `/plugins/servlet/restbrowser#/user/search` to the end of the URL for accessing JIRA (e.g. `http://localhost:2990/jira/plugins/servlet/restbrowser#/user/search`) within the SDK environment.

^Top

Stable Java API

JIRA's Java API has undergone a significant number of changes and improvements to provide the following:

- More stability and compatibility with future versions of JIRA. See our Java API Policy for JIRA for details.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.
- Active Objects is bundled with JIRA 5.0. A new version of Active Objects will be available in the official JIRA 5.0 release. However, if you are developing against JIRA 5.0 RC 3, we recommend that you can obtain the latest version of Active Objects (0.18.4) from our public Maven repository. Refer to our instructions on how to upgrade to this version of Active Objects in JIRA 5.0 RC 3.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

If you are developing plugins for JIRA 5.0, please also check out our newly published set of JIRA plugin tutorials.

^Top

Performance improvements

Lucene 3.2 is now fully integrated into JIRA. Customers with moderate to large JIRA installations may notice a significant performance improvement in searches, particularly when other actions and operations such as creating and editing issues are conducted simultaneously.

The content of each 'activity' tab on the 'view issue' page now loads independently of the rest of the 'view issue' page. Instead, the content of an activity tab now only loads when the user clicks the tab. This revised behaviour allows the information on these tabs to show up more rapidly.

^Top

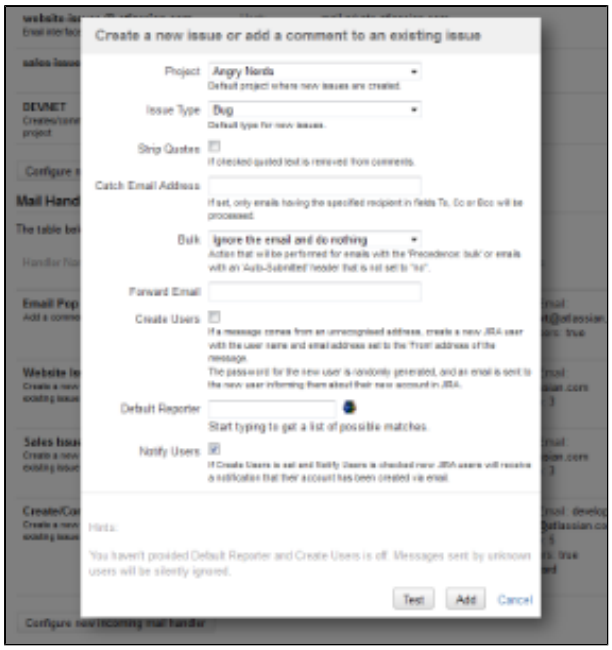
New troubleshooting and debugging tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

- A number of email debugging tools are now provided to System Administrators in the new **Logging and Profiling** page under **Troubleshooting and Support**
  - Enable or disable mail logging
  - Turn debug mail logging on or off
  - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New email handler wizard



JIRA 5.0 incorporates a new mail handler wizard that greatly simplifies the process of configuring incoming mail handlers for creating issues or comments from email messages.

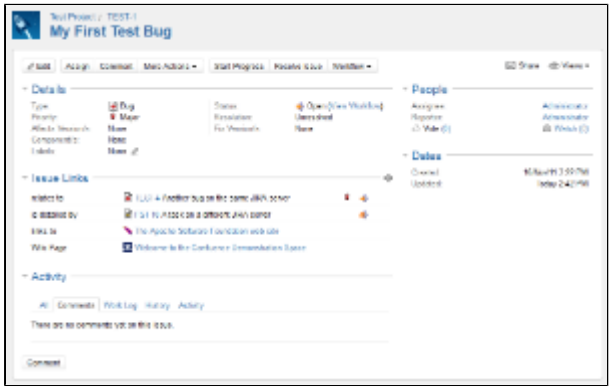
There is no longer a need to configure a JIRA service and enter a complex string of mail handler parameters to handle your email message Instead, simply you configure your mail handler through a convenient wizard.

Improvements have been made to the layout of JIRA's mail configuration options. The configuration options in the **Mail Servers** administration page have been separated into two separate pages — one for **Outgoing Mail** (SMTP) and another for **Incoming Mail** (POP/IMAP) configurations. The configuration options for mail handlers have been moved from the **Services** administration page and incorporated into the **Incoming Mail** page.

^Top

13

Enhancements to the 'view issue' page



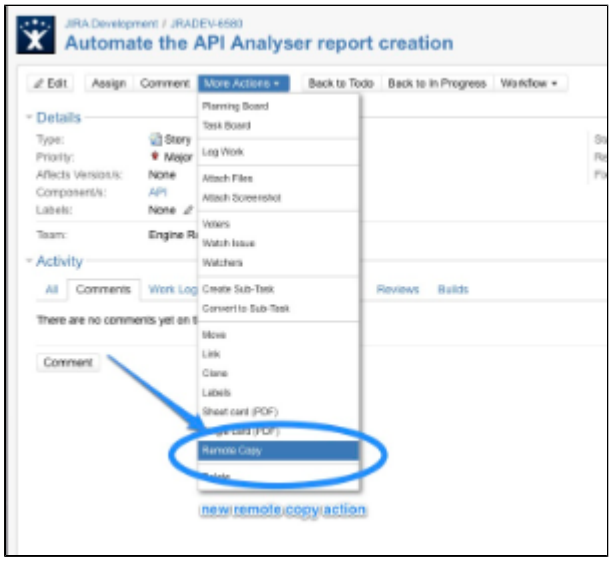
The 'view issue' page has the following enhancements:

- The **Issue Links** section of the 'view issue' page has been redesigned to cater for remote issue links (above) and makes better use of space — the separate line used to group issue link types in earlier versions of JIRA has been removed.
- The right-hand side of the view issue page now uses a fixed width. Hence, when the 'view page' is maximised on large, high resolution monitors, the association between field names and values is not lost.
- **For plugin developers** - the left hand side of the 'view issue' page can be customised via web panels. Hence, you can now insert your own custom panels anywhere below the operations bar on the 'view issue' page.
  The web panel location for the left-hand side of the 'view issue' page is `atl.jira.view.issue.left.context` — refer to the View Issue Page Locations page of the Web Fragments guide in our Atlassian Developers documentation site for more

information. Also see our plugin tutorial for an example of customising the right-hand side of the 'view issue' page (a plugin developer-feature which became available in JIRA 4.4).

^Top

New plugin to try out — JIRA to JIRA issue copy



This new JIRA 5.0-compatible feature currently under development as a plugin, allows you to copy issues from one JIRA site to another.

After establishing an Application Link between your JIRA site and another and one or more Project Links across these sites, a new 'Remote Copy' action will become available when viewing an issue belonging to a project involved in this project link.

Using JIRA to JIRA issue copy

After clicking the **Remote Copy** menu item, the Remote Copy 'wizard' will prompt you to choose a remote JIRA project to copy the issue to and will then automatically map fields between your local and remote JIRA projects.

Most system fields and some custom fields are supported. For a detail list of supported fields, see our Issue Field copy rules documentation

When copying an issue, you can define a bidirectional remote 'issue link' between the issues. If the remote site is on a version of JIRA that does not support remote 'issue links' (prior to 5.0), you can create a single remote 'issue link' from the local issue to the remote issue and a comment will be added to the remote issue with a link to the local issue it was copied from.

Configuring JIRA to JIRA issue copy

The JIRA to JIRA issue copy feature is available to all users by default, although it can be restricted to a particular group.

If certain mandatory System or Custom fields in your local JIRA project are unlikely to exist or have valid values in the remote JIRA projects from which you are copying issues, you can specify default values for these required fields in your local JIRA project. These default values will also be used in the event that the user copying an issue remotely does not have permission to set these fields in your local JIRA project.

Getting the JIRA to JIRA issue copy plugin

The JIRA to JIRA issue copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between.

⚠ The JIRA to JIRA issue copy plugin is not bundled with JIRA, although it can currently be download from from the Atlassian Plugin Exchange.

^Top

Other enhancements and fixes

When you access your new or upgraded JIRA 5.0 installation, JIRA launches the **What's New in JIRA ...** dialog box, which provides brief overview of the new features available in that JIRA version.

✅ This dialog box can be prevented from showing up whenever you access JIRA by selecting the **Don't show again** check box at the base of the dialog box. However, you can access this dialog box again by choosing the **What's New** item from your user name drop down menu.

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

JIRA 5.0 RC 2 Release Notes

8 December 2011

JIRA 5.0 RC 2 (a.k.a 5.0 milestone 10 or 'm10') is a public development release leading up to **JIRA 5.0**. A **Release Candidate (RC)** is a preliminary release leading up to the official release of a JIRA version. RC releases are a fairly stable snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use RC releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 5.0 RC 2 release. Thank you for your feedback during the recent Beta and EAP releases and please keep providing it here.

Upgrading to JIRA 5.0 RC 2:

- JIRA EAP/Beta/RC releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes. If you are a plugin developer, please also note the minor differences in markup/style handling between RC 1 and RC 2.

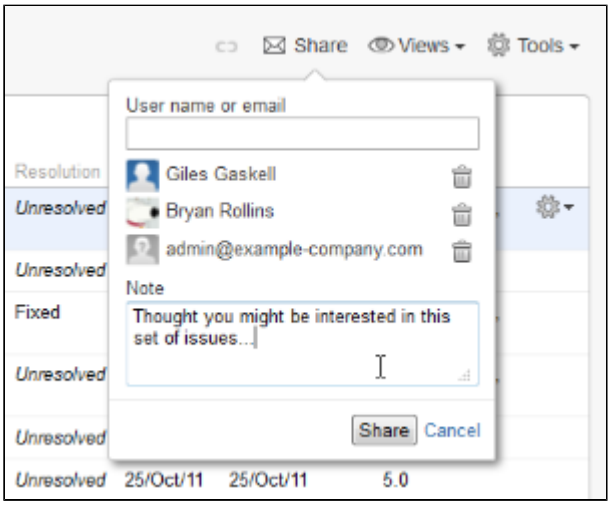
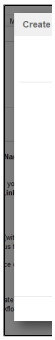
Overview

The focus of JIRA 5.0 is to help connect people, teams and other applications together. JIRA 5.0 helps connect people and teams by allowing them to share issues and search results and mention other team members in an issue. People and teams are better connected to other applications with Remote 'Issue Links' to other JIRA sites or web page URLs and Activity Streams that span all your linked Atlassian applications.

JIRA 5.0 also introduces a large number of improvements for developers to help connect JIRA to other applications, including the introduction of:

- New REST APIs to manage every facet of an issue,
- New Remote 'Issue Link' and Activity Stream features, specifically for integration with other applications, and
- A stable Java API for JIRA.

Highlights:

Remote 'issue links'	Sharing issues and 'mentions'	
		

• Create links to JIRA issues on another JIRA site • Create links to any URL • Create links via JIRA's Java or REST API	• Share issues quickly with other users • Mention a user in an issue comment • JIRA uses autocomplete whenever you 'share' or 'mention' a user	

And More:

- Search for issues based on their history
- Activity streams now show activity from other applications
- Manage other users shared filters and dashboards
- Administration user interface improvements

- REST API (with tutorials) for working with issues in JIRA
- Stable Java API
- Performance improvements
- New troubleshooting and debugging tools
- New email handler wizard
- JIRA to JIRA issue copy

Thank you for your interest in JIRA 5.0 RC 2
Download RC 2

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights

1

Remote 'issue links'

The *remote 'issue links'* feature provides a powerful way to link JIRA issues to items external to your JIRA installation, residing on external applications.

Along with a Java and REST API to add these links, end users can also:

- Add an issue link from a JIRA issue to an issue on another JIRA site via application links.
    - Add a reciprocal issue link from your JIRA issue to an issue another JIRA site.
- Search for a Confluence page from a JIRA issue and add an issue link to that page via application links.
- Add an issue link from a JIRA issue to any web page URL, such as a page of documentation, a technical note, or any other page o another web site.

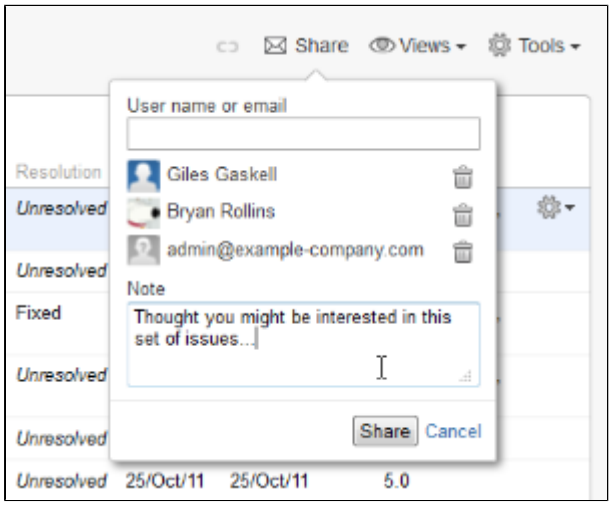
Check out a live example here.

If you are a:

- JIRA user — see Linking Issues for details on using remote 'Issue Links' within JIRA.
- JIRA system administrator — see Configuring Issue Linking for details on how to make remote 'Issue Linking' available to your users by setting up the required application links between JIRA and other applications.
- JIRA developer — see JIRA Remote Issue Links on our developer documentation site.

^Top

Sharing issues and 'mentions'



Screenshot: Sharing an issue with other users

Screenshot: Mentioning a user in an issue comment

Want to let others know about a JIRA issue or a list of search results quickly? Simply visit a JIRA issue or a list of issues on the issue navigator and then do either of the following:

- Click the new **Share** button at the top-right (or type **s**) and specify JIRA users (based on their names or user names) or *any* email address of people you want to share the page with. Add an optional note and then click **Share**.

    JIRA users specified in the **User name or email** field will receive details of the issue or a link to a list of issues in an email message (sent to the addresses registered with their JIRA user accounts). The message's subject line will indicate that the person who used this feature 'shared' the issue(s) with them.

- In an issue comment or the issue's description field, type the @ symbol and begin typing the name or username of a JIRA user. Select the appropriate user from the dropdown. You can do this with multiple users and after submitting the form, those users will b notified that you mentioned them in the comment or the issue's description field.

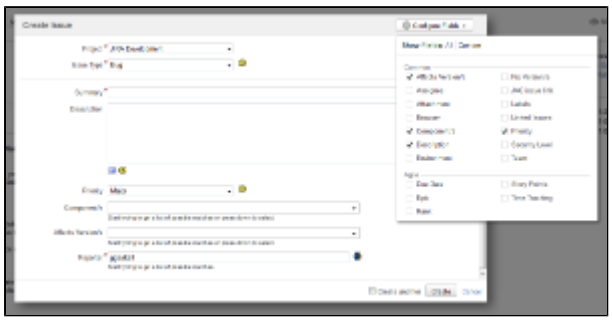
Users will receive details of the issue in an email message (sent to the addresses registered with their JIRA user accounts) The message's subject line will indicate that the person who used this feature 'mentioned' them on that issue.

Please Note:

- Users required the **Browse Users** global permission to access the **Share** button or the autocomplete feature when 'mentioning' a user. However, if you know the username of a JIRA user, you can still mention them.
- JIRA system administrators will need to configure JIRA's outgoing mail server for the 'share' and 'mentions' features to wor

^Top

Create and edit issues rapidly



JIRA 5.0 provides a *rapid and customisable* dialog box for creating and editing JIRA issues and sub-tasks. Unlike the old Create/Edit Issue forms, these dialog boxes do not require a page reload.

By default, the Create/Edit Issue dialog boxes show all available fields, which you can quickly remove or re-add. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly edited fields whenever you create or edit an issue.

The Create/Edit Issue dialog box allows you to rapidly create a series of related issues with similar options. When you select the '**Create another**' check box before clicking the '**Create**' button, JIRA creates your issue and automatically pre-populates a new 'Create Issue' dialog box with your previous options, whilst leaving the 'Summary' field blank. Note that this feature does not carry across any attachments that were attached to your previously created issue.

Note that the 'Assignee', 'Project' and 'Issue Type' fields use 'autocompletion' too. Hence, you no longer a need to scroll through a whole raft of items to specify these fields when creating or editing an issue.

You can easily access this feature by typing '**c**' to access the create issue dialog box or '**e**' from a currently selected issue on the Issue Navigator or 'View Issue' page to access the edit dialog box for that issue.

^Top

Search for issues based on their history

"CHANGED" operator introduced

Introduced in JIRA 4.4.3, JQL's "CHANGED" operator can accept the optional predicates **FROM**, **TO**, **ON**, **DURING**, **BEFORE**, **AFTER** and **BY**, and can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

For example, this link shows all the issues logged against the JIRA project on our 'jira.atlassian.com' site, whose **Fix Version** field was changed to "5.0".

You can also create more complex JQL queries with the "CHANGED" operator by fine-tuning them with predicates. For example, the following JQL query:

```
status changed FROM "In QA Review" TO "QA Rejected" BY freddo AFTER startOfWeek() BEFORE
endOfWeek()
```

will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo' between the start and end of the current week.

You can use complex queries such as these to generate the 'Single Level Group By Report' in the screenshot above, which shows grouping by 'Team'.

"WAS" operator enhanced

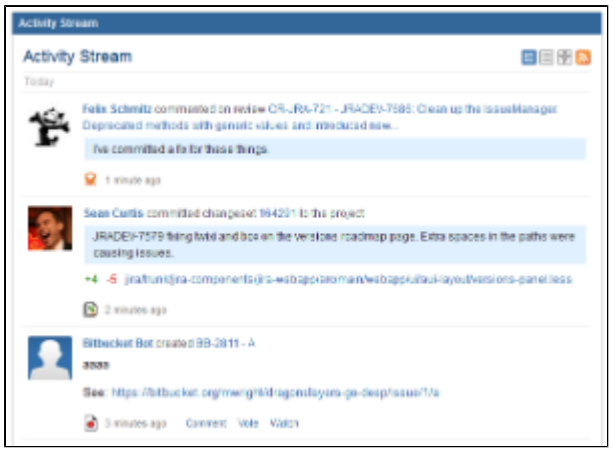
Also introduced in JIRA 4.4.3 was the ability of the "WAS" operator to work with the **Fix Version** field. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

^Top

Activity streams now show activity from other applications



Screenshot: Activity stream gadget showing activity from other Atlassian applications

Screenshot: Activity stream gadget showing activity from non-Atlassian applications

We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

With these new features, you can:

- See Confluence page updates from your activity streams in JIRA, then drill down into those Confluence pages for more information.
- See updates from another JIRA site. For example, activity streams on your development team's JIRA site (behind the firewall) can include activity on your support team's customer facing JIRA site.

^Top

Manage other users' shared filters and dashboards

Introduced in JIRA 4.4.1, JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboard. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

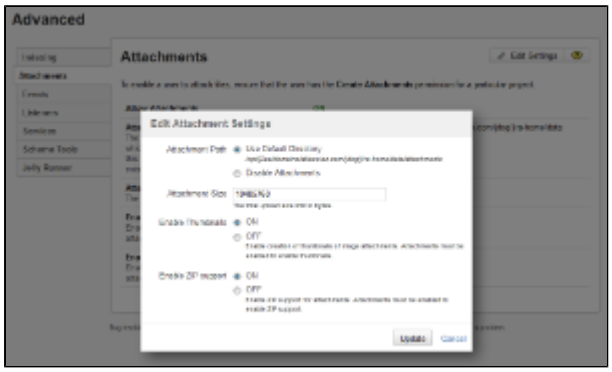
On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

^Top

7

Administration user interface improvements



Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 will provide further improvements to the Administration UI by converting various forms and pages in this area to convenient dialog boxes.

In addition to the dialog box for adding a new user, the 'Attachments' and 'Workflows' pages have been redesigned and the forms associate with these pages have been converted into convenient dialog boxes too.

For example, the form for editing attachment settings is now a dialog box, which is accessed by clicking '**Edit Settings**' on the 'Attachments page of JIRA's Administration area.

^Top

8

REST API (with tutorials) for working with issues in JIRA

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues and editing existing ones.

- Delete existing issues and their subtasks.
- Create remote 'issue links'.
- Retrieve metadata from your favourite filters and dashboards.
- Retrieve metadata about your permissions.
- Almost all system fields and JIRA's built-in custom field types are supported.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2' (or '`latest`' to use the latest REST API version available with your version of JIRA).

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

You might also want to try out Atlassian's new REST API Browser, which is available in the Atlassian Plugin SDK and can be accessed by adding `/plugins/servlet/restbrowser#/user/search` to the end of the URL for accessing JIRA (e.g. `http://localhost:2990/jira/plugins/servlet/restbrowser#/user/search`) within the SDK environment.

^Top

Stable Java API

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA. See our Java API Policy for JIRA for details.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.
- Active Objects is bundled with JIRA 5.0. A new version of Active Objects will be available in the official JIRA 5.0 release. However, if you are developing against JIRA 5.0 RC 2, we recommend that you can obtain the latest version of Active Objects (0.18.4) from our public Maven repository. Refer to our instructions on how to upgrade to this version of Active Objects in JIRA 5.0 RC 2.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

If you are developing plugins for JIRA 5.0, please also check out our newly published set of JIRA plugin tutorials.

^Top

Performance improvements

Lucene 3.2 is now fully integrated into JIRA. Customers with moderate to large JIRA installations may notice a significant performance improvement in searches, particularly when other actions and operations such as creating and editing issues are conducted simultaneously.

The content of each 'activity' tab on the 'view issue' page now loads independently of the rest of the 'view issue' page. Instead, the content of an activity tab now only loads when the user clicks the tab. This revised behaviour allows the information on these tab to show up more rapidly.

^Top

New troubleshooting and debugging tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

- A number of email debugging tools are now provided to System Administrators in the new 'Logging and Profiling' page under 'Troubleshooting and Support'
  - Enable or disable mail logging
  - Turn debug mail logging on or off
  - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New email handler wizard

JIRA 5.0 incorporates a new email handler wizard that greatly simplifies the process of configuring incoming mail handlers for creating issue or comments from email messages.

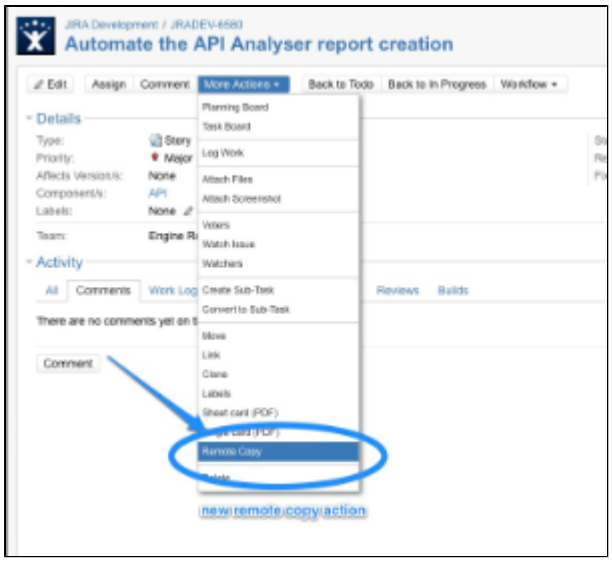
You no longer have to configure a JIRA service and enter in a complex string of Handler parameters for your email handler. Instead you configure your email handler through a convenient wizard.

Improvements have been made to the layout of JIRA's mail configuration options. The configuration options in the 'Mail Servers' tab has bee separated into two tabs — one for 'Outgoing Mail' (SMTP) and another for 'Incoming Mail' (POP/IMAP) configuration. The configuration options for mail handlers in the 'Services' tab has been incorporated into the 'Incoming Mail' tab.

^Top

13

New plugin to try out — JIRA to JIRA issue copy



This new JIRA 5.0-compatible feature currently under development as a plugin, allows you to copy issues from one JIRA site to another.

After establishing an Application Link between your JIRA site and another and one or more Project Links across these sites, a new 'Remote Copy' action will become available when viewing an issue belonging to a project involved in this project link.

Using JIRA to JIRA issue copy

After clicking the '**Remote Copy**' menu item, the Remote Copy 'wizard' will prompt you to choose a remote JIRA project to copy the issue to and will then automatically map fields between your local and remote JIRA projects.

Most system fields and some custom fields are supported. For a detail list of supported fields, see our Issue Field copy rules documentation

When copying an issue, you can define a bidirectional remote 'issue link' between the issues. If the remote site is on a version of JIRA that does not support remote 'issue links' (prior to 5.0), you can create a single remote 'issue link' from the local issue to the remote issue and a comment will be added to the remote issue with a link to the local issue it was copied from.

Configuring JIRA to JIRA issue copy

The JIRA to JIRA issue copy feature is available to all users by default, although it can be restricted to a particular group.

If certain mandatory System or Custom fields in your local JIRA project are unlikely to exist or have valid values in the remote JIRA projects from which you are copying issues, you can specify default values for these required fields in your local JIRA project. These default values will also be used in the event that the user copying an issue remotely does not have permission to set these fields in your local JIRA project.

Getting the JIRA to JIRA issue copy plugin

The JIRA to JIRA issue copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between.

⚠ The JIRA to JIRA issue copy plugin is not bundled with JIRA, although it can currently be download from from the Atlassian Plugin Exchange.

^Top

Other enhancements and fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

Plugin Developers: Differences between RC1 and RC2

There were some minor changes to markup/style handling between RC1 and RC2, notably:

- If you are not using the JIRA decorators, but your plugin includes markup with `<div class="content-body">` elements, you will need to add an additional `aui-panel` class attribute to these elements, i.e. `<div class="content-body aui-panel">`
- If your plugin was replicating the behaviour of the 'view issue' page, there were some changes made to the markup:
    1. A new class `page-type-viewissue` is required on the body.
    2. The content of the stalker (`<header id="stalker" class="stalker">`) is now wrapped in a `div` element with a `stalker-content` class attribute, i.e. `<header id="stalker" class="stalker"> ... </header>` is now `<header id="stalker" class="stalker"><div class="stalker-content"> ... </div></header>`

^Top

JIRA 5.0 RC 1 Release Notes

21 November 2011

JIRA 5.0 RC 1 (a.k.a 5.0 milestone 9 or 'm9') is a public development release leading up to **JIRA 5.0**. A **Release Candidate (RC)** is a preliminary release leading up to the official release of a JIRA version. RC releases are a fairly stable snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use RC releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 5.0 RC 1 release. Thank you for your feedback during the recent Beta and EAP releases and please keep providing it here.

Upgrading to JIRA 5.0 RC 1:

- JIRA EAP/Beta/RC releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

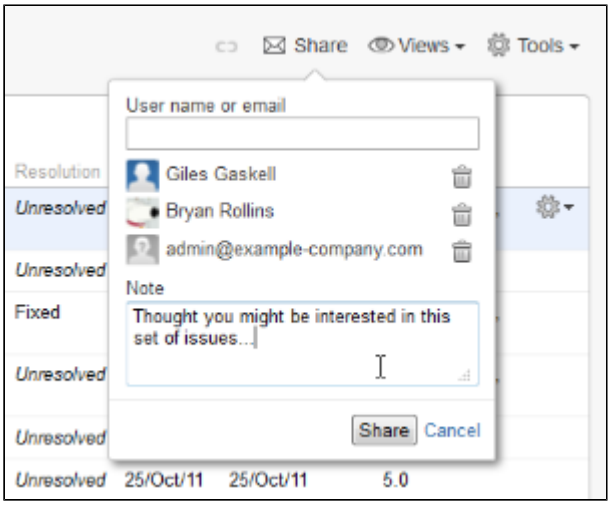
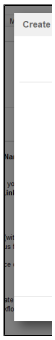
Overview

The focus of JIRA 5.0 is to help connect people, teams and other applications together. JIRA 5.0 helps connect people and teams by allowing them to share issues and search results and mention other team members in an issue. People and teams are better connected to other applications with Remote 'Issue Links' to other JIRA sites or web page URLs and Activity Streams that span all your linked Atlassian applications.

JIRA 5.0 also introduces a large number of improvements for developers to help connect JIRA to other applications, including the introduction of:

- New REST APIs to manage every facet of an issue,
- New Remote 'Issue Link' and Activity Stream features, specifically for integration with other applications, and
- A stable Java API for JIRA.

Highlights:

Remote 'Issue Links'	Sharing Issues and 'Mentions'	
		

• Create links to JIRA issues on another JIRA site • Create links to any URL • Create links via JIRA's Java or REST API	• Share issues quickly with other users • Mention a user in an issue comment • JIRA uses autocomplete whenever you 'share' or 'mention' a user	

And More:

- More Enhancements to JQL
- Activity Streams Now Show External Content
- Manage Other Users Shared Filters and Dashboards
- Administration User Interface Improvements

- REST API Improvements
- Java API Improvements
- Performance Improvements
- New Troubleshooting and Debugging Tools
- New Email Handler Wizard
- JIRA to JIRA Issue Copy

Thank you for your interest in JIRA 5.0 RC 1
Download RC 1

Please be aware of a known issue specific to this JIRA 5.0 RC 1 Release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights

1

Remote 'Issue Links' *(improved in RC 1)*

The Remote 'Issue Links' feature provides a powerful way to link JIRA issues to items external to your JIRA installation, residing on external applications.

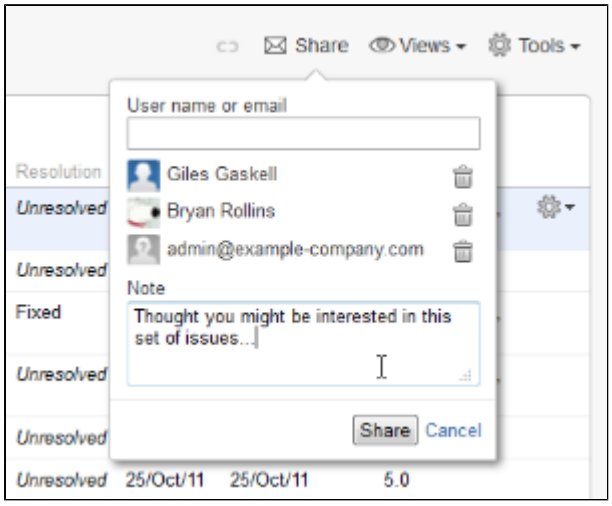
Along with a Java and REST API to add these links, end users can also:

- Add a link from a JIRA issue to another issue on a different JIRA installation or site using application links with trusted applications OAuth-based authentication.
- Add a reciprocal link from a JIRA issue to another issue on a different JIRA installation or site using application links.
- Search for a Confluence page from a JIRA issue and add a link to that page from the issue (new in RC 1).
- Add a link from a JIRA issue to any web page URL, such as a page of documentation, a technical note, or any other page on another web site.

For more information about this feature, see JIRA Remote Issue Links.

^Top

Sharing Issues and 'Mentions'



Screenshot: Sharing an issue with other users

Screenshot: Mentioning a user in an issue comment

Want to let others know about a JIRA issue or a list of search results quickly? Simply visit a JIRA issue or a list of issues on the issue navigator and then do either of the following:

- Click the new **'Share**' button at the top-right (or type '**s**') and specify JIRA users (based on their names or user names) or *any* email address of people you want to share the 'view issue' page with. Add an optional note and then click '**Share**'.

  JIRA users specified in the 'User name or email' field will receive details of the issue or a link to a list of issues in an email message (sent to the addresses registered with their JIRA user accounts). The message's subject line will indicate that the person who used this feature 'shared' the issue(s) with them.

- In an issue comment or the issue's description field, type the '@' symbol and begin typing the name or username of a JIRA user. Select the appropriate user from the dropdown. You can do this with multiple users and after submitting the form, those users will b notified that you mentioned them in the comment or the issue's description field.

  Users will receive details of the issue in an email message (sent to the addresses registered with their JIRA user accounts) The message's subject line will indicate that the person who used this feature 'mentioned' them on that issue.

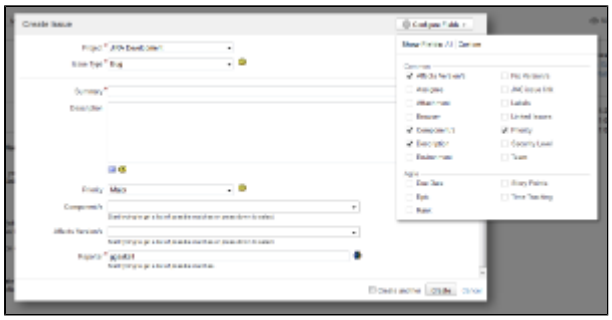
Please Note:

- Users required the 'Browser User' global permission to access the 'Share' button or the autocomplete feature when 'mentioning' a user. However, if you know the username of a JIRA user, you can still mention them.
- JIRA system administrators will need to configure JIRA's outgoing mail server for the 'Share' and 'mentions' features to work.

^Top

3

Create and Edit Issues Rapidly



JIRA 5.0 provides a rapid and customisable dialog box for creating and editing JIRA issues and sub-tasks. Unlike the old Create/Edit Issue forms, these dialog boxes do not require a page reload.

By default, the Create/Edit Issue dialog boxes show all available fields, which you can quickly remove or re-add. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly edited fields whenever you create or edit an issue.

The Create/Edit Issue dialog box allows you to rapidly create a series of related issues with similar options. When you select the '**Create another**' check box before clicking the '**Create**' button, JIRA creates your issue and automatically pre-populates a new 'Create Issue' dialog box with your previous options, whilst leaving the 'Summary' field blank. Note that this feature does not carry across any attachments that were attached to your previously created issue.

You can easily access this feature by typing '**c**' to access the create issue dialog box or '**e**' from a currently selected issue on the Issue Navigator or 'View Issue' page to access the edit dialog box for that issue.

^Top

4

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements

Enhancements were introduced in JIRA 4.4 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

Introduced in JIRA 4.4.1, JQL's "CHANGED" operator can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status changed FROM "In QA Review" to "QA Rejected" BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo', and after the start and before the end of the current week.

You can use complex queries such as these to generate the 'Single Level Group By Report' in the screenshot above, which shows grouping by 'Team'.

The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

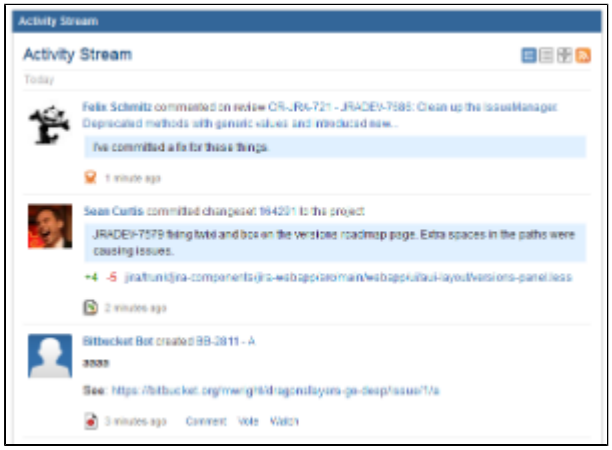
The "WAS" operator can now be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

^Top

Activity Streams Now Show External Content



Screenshot: Activity stream gadget showing activity from other Atlassian applications

Screenshot: Activity stream gadget showing activity from non-Atlassian applications

We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

^Top

Manage Other Users' Shared Filters and Dashboards

Introduced in JIRA 4.4.1, JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboard. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

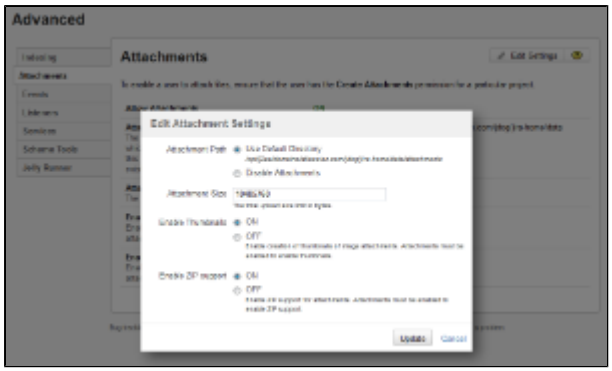
You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

^Top

Administration User Interface Improvements



Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 will provide further improvements to the Administration UI by converting various forms and pages in this area to convenient dialog boxes.

In addition to the dialog box for adding a new user, the 'Attachments' and 'Workflows' pages have been redesigned and the forms associate with these pages have been converted into convenient dialog boxes too.

For example, the form for editing attachment settings is now a dialog box, which is accessed by clicking '**Edit Settings**' on the 'Attachments page of JIRA's Administration area.

^Top

8

REST API Improvements

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues and editing existing ones.
- Delete existing issues and their subtasks.
- Create remote 'issue links'.
- Retrieve metadata from your favourite filters and dashboards.
- Retrieve metadata about your permissions.
- Almost all system fields and JIRA's built-in custom field types are supported.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2' (or '`latest`' to use the latest REST API version available with your version of JIRA).

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details. We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

You might also want to try out Atlassian's new REST API Browser, which is available in the Atlassian Plugin SDK and can be accessed by adding `/plugins/servlet/restbrowser#/user/search` to the end of the URL for accessing JIRA (e.g. `http://localhost:2990/jira/plugins/servlet/restbrowser#/user/search`) within the SDK environment.

^Top

9

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Customers with moderate to large JIRA installations may notice a significant performance improvement in searches, particularly when other actions and operations such as creating and editing issues are conducted simultaneously.

The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

^Top

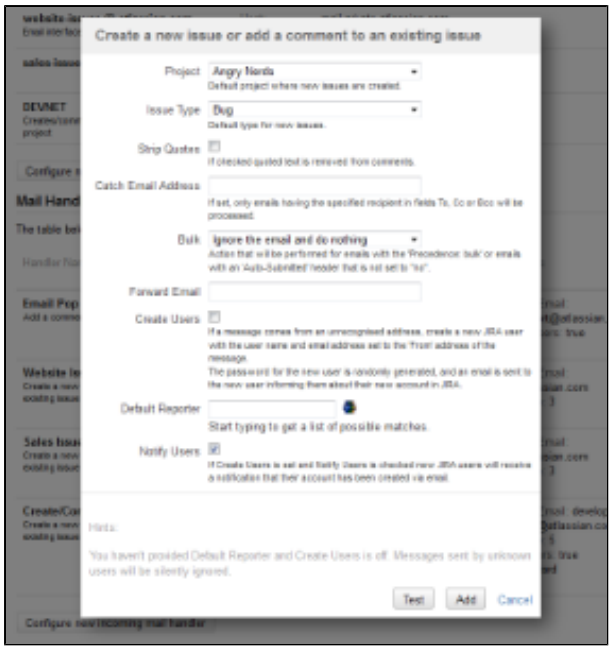
New Troubleshooting and Debugging Tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

- A number of email debugging tools are now provided to System Administrators in the new 'Logging and Profiling' page under 'Troubleshooting and Support'
  - Enable or disable mail logging
  - Turn debug mail logging on or off
  - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New Email Handler Wizard



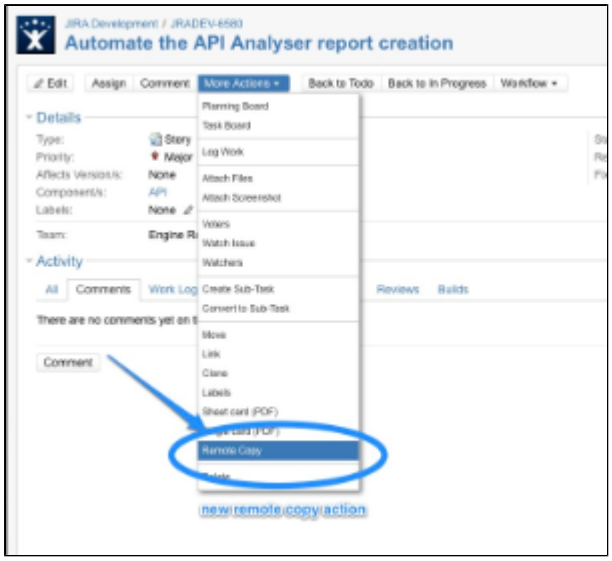
JIRA 5.0 incorporates a new email handler wizard that greatly simplifies the process of configuring incoming mail handlers for creating issue or comments from email messages.

You no longer have to configure a JIRA service and enter in a complex string of Handler parameters for your email handler. Instead you configure your email handler through a convenient wizard.

Improvements have been made to the layout of JIRA's mail configuration options. The configuration options in the 'Mail Servers' tab has bee separated into two tabs — one for 'Outgoing Mail' (SMTP) and another for 'Incoming Mail' (POP/IMAP) configuration. The configuration options for mail handlers in the 'Services' tab has been incorporated into the 'Incoming Mail' tab.

^Top

***New Plugin to Try Out* — JIRA to JIRA Issue Copy**



This new JIRA 5.0-compatible feature currently under development as a plugin, allows you to copy issues from one JIRA site to another.

After establishing an Application Link between your JIRA site and another and one or more Project Links across these sites, a new 'Remote Copy' action will become available when viewing an issue belonging to a project involved in this project link.

Using JIRA to JIRA Issue Copy

After clicking the '**Remote Copy**' menu item, the Remote Copy 'wizard' will prompt you to choose a remote JIRA project to copy the issue to and will then automatically map fields between your local and remote JIRA projects.

Most system fields and some custom fields are supported. For a detail list of supported fields, see our Issue Field copy rules documentation

When copying an issue, you can define a bidirectional remote 'issue link' between the issues. If the remote site is on a version of JIRA that does not support remote 'issue links' (prior to 5.0), you can create a single remote 'issue link' from the local issue to the remote issue and a comment will be added to the remote issue with a link to the local issue it was copied from.

Configuring JIRA to JIRA Issue Copy

The JIRA to JIRA issue copy feature is available to all users by default, although it can be restricted to a particular group.

If certain mandatory System or Custom fields in your local JIRA project are unlikely to exist or have valid values in the remote JIRA projects from which you are copying issues, you can specify default values for these required fields in your local JIRA project. These default values will also be used in the event that the user copying an issue remotely does not have permission to set these fields in your local JIRA project.

Getting the JIRA to JIRA Issue Copy Plugin

The JIRA to JIRA issue copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between.

The JIRA to JIRA issue copy plugin is not bundled with JIRA, although it can currently be download from from the Atlassian Plugin Exchange.

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

Known Issues in JIRA 5.0 RC 1

JIRA 5.0 RC 1 possesses a bug (JRA-26300) which prevents the ability to attach files to issues when using the Safari web browser. This will be fixed in JIRA 5.0 RC 2.

^Top

JIRA 5.0 Beta 3 Release Notes

1 November 2011

JIRA 5.0 Beta 3 (a.k.a 5.0 milestone 8 or 'm8') is a public development release leading up to **JIRA 5.0**. A **Beta release** is a preliminary release leading up to the official release of a JIRA version. Beta releases are a snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use Beta releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 5.0 Beta 3 release. Thank you for your feedback during the recent EAP releases and please keep providing it here.

Upgrading to JIRA 5.0 Beta 3:

- JIRA EAP/Beta releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Overview

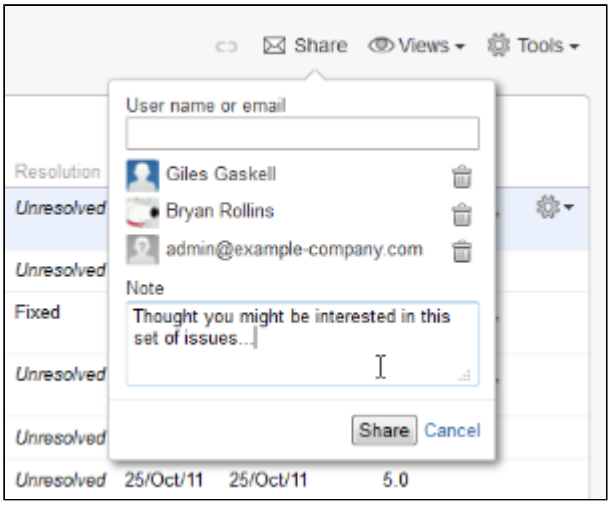
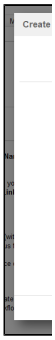
The focus of JIRA 5.0 is to help connect people, teams and other applications together. JIRA 5.0 helps connect people and teams by allowing them to share issues and search results and mention other team members in an issue. People and teams are better connected to other applications with Remote 'Issue Links' to other JIRA sites or web page URLs and Activity Streams that span all your linked Atlassian applications.

JIRA 5.0 also introduces a large number of improvements for developers to help connect JIRA to other applications, including the introduction of:

- New REST APIs to manage every facet of an issue,
- New Remote 'Issue Link' and Activity Stream features, specifically for integration with other applications, and

- A stable Java API for JIRA.

Main Highlights:

Remote 'Issue Links'	Sharing Issues and 'Mentions'	
		
• Create links to JIRA issues on another JIRA site • Create links to any URL • Create links via JIRA's Java or REST API	• Share issues quickly with other users • Mention a user in an issue comment • JIRA uses autocomplete whenever you 'share' or 'mention' a user	

Other Highlights:

- More Enhancements to JQL
- Activity Streams Now Show External Content
- Manage Other Users Shared Filters and Dashboards
- Administration User Interface Improvements ***(improved in Beta 3)***

- REST API Improvements ***(improved in Beta 3)***
- Java API Improvements
- Performance Improvements
- New Troubleshooting and Debugging Tools
- New Email Handler Wizard
- Remote Issue Copying

Thank you for your interest in JIRA 5.0 Beta 3
Download Beta

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights

1

Remote 'Issue Links' *(improved in Beta 3)*

The Remote 'Issue Links' feature provides a powerful way to link JIRA issues to items external to your JIRA installation, residing on external applications.

Along with a Java and REST API to add these links, end users can also:

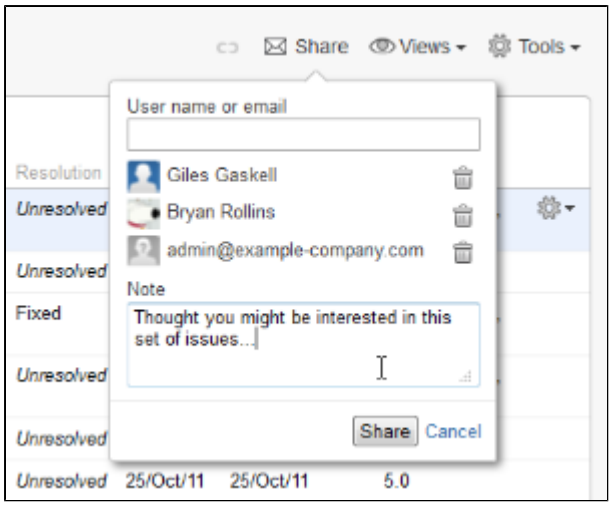
- Add a link from a JIRA issue to another issue on a different JIRA installation or site using application links with trusted applications o OAuth-based authentication (improved in Beta 3).
- Add a reciprocal link from a JIRA issue to another issue on a different JIRA installation or site using application links.
- Add a link from a JIRA issue to any web page URL, such as a page of documentation, a technical note, or any other page on another web site.

For more information about this feature, see JIRA Remote Issue Links.

^Top

2

Sharing Issues and 'Mentions' *(improved in Beta 3)*



Screenshot: Sharing an issue with other users

Screenshot: Mentioning a user in an issue comment

Want to let others know about a JIRA issue or a list of search results quickly? Simply visit a JIRA issue or a list of issues on the issue navigator and then do either of the following:

- Click the new **'Share'** button at the top-right (or type '**s**') and specify JIRA users (based on their names or user names) or *any* email address of people you want to share the page with. Add an optional note and then click '**Share**'.

  JIRA users specified in the 'User name or email' field will receive details of the issue or a link to a list of issues in an email message (sent to the addresses registered with their JIRA user accounts). The message's subject line will indicate that the person who used this feature 'shared' the issue(s) with them.

- In an issue comment or the issue's description field (new in Beta 3), type the '@' symbol and begin typing the name or username of a JIRA user. Select the appropriate user from the dropdown. You can do this with multiple users and after submitting the form, those users will be notified that you mentioned them in the comment or the issue's description field.

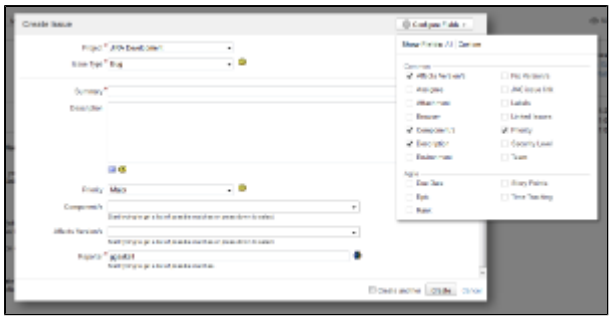
  Users will receive details of the issue in an email message (sent to the addresses registered with their JIRA user accounts). The message's subject line will indicate that the person who used this feature 'mentioned' them on that issue.

Users required the '**Browser User**' global permission to access the '**Share**' button or the autocomplete feature when 'mentioning' a user. However, if you know the username of a JIRA user, you can still mention them

^Top

3

Create and Edit Issues Rapidly *(improved in Beta 3)*



JIRA 5.0 provides a rapid and customisable dialog box for creating and editing JIRA issues and sub-tasks (new in Beta 3). This feature replaces the Create and Edit Issue forms.

The Create/Edit Issue dialog boxes show all available fields by default. However, you can quickly remove or re-add fields via the '**Configure Fields**' button at the top-right of the dialog box. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly edited fields whenever you create or edit an issue.

If you select the '**Create another**' check box, JIRA will create your issue and automatically pre-populate a new 'Create Issue' dialog box with your previous options, whilst leaving the 'Summary' field blank. This allows you to rapidly create a series of related issues with similar options. Note that this feature does not carry across any attachments that were attached to your previously created issue.

Consistent with JIRA's existing keyboard shortcuts, type '**c**' to access the create issue dialog box or '**e**' from a currently selected issue on the Issue Navigator or 'View Issue' page to access the edit dialog box for that issue.

^Top

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements

Enhancements were introduced in JIRA 4.4 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

In JIRA 5.0, JQL supports the new "CHANGED" operator, which can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status changed FROM "In QA Review" to "QA Rejected" BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo', and after the start and before the end of the current week.

You can use complex queries such as these to generate the 'Single Level Group By Report' in the screenshot above, which shows grouping by 'Team'.

The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

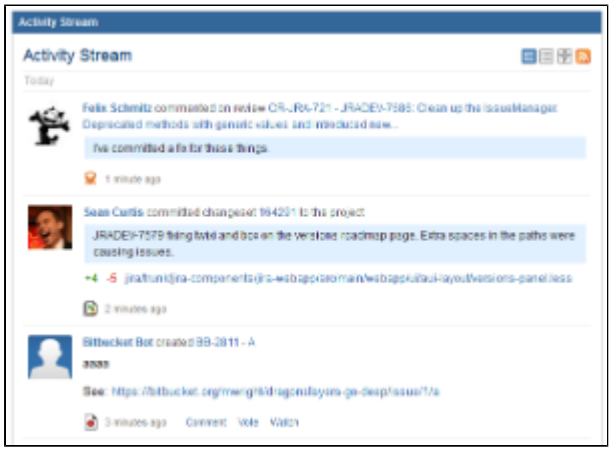
The "WAS" operator can now be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

^Top

Activity Streams Now Show External Content



Screenshot: Activity stream gadget showing activity from other Atlassian applications

Screenshot: Activity stream gadget showing activity from non-Atlassian applications

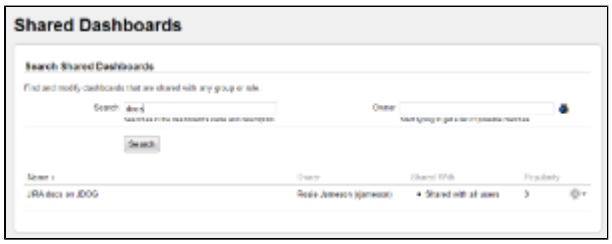
We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

^Top

Manage Other Users' Shared Filters and Dashboards



JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboards. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

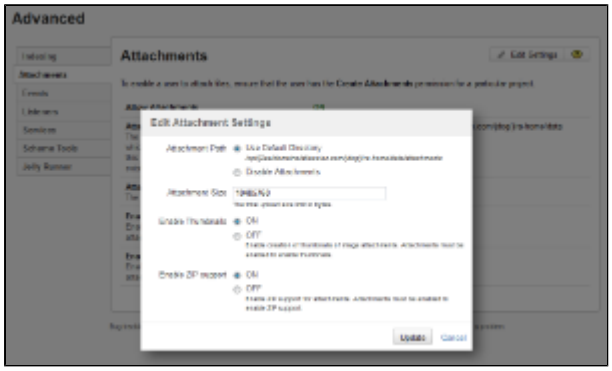
Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard.

^Top

Administration User Interface Improvements *(improved in Beta 3)*



Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 will provide further improvements to the Administration UI by converting various forms and pages in this area to convenient dialog boxes.

In addition to the dialog box for adding a new user, the 'Attachments' and 'Workflows' pages (the latter of which is improved in Beta 3) have been redesigned and the forms associated with these pages have been converted into convenient dialog boxes too.

For example, the form for editing attachment settings is now a dialog box, which is accessed by clicking '**Edit Settings**' on the 'Attachments' page of JIRA's Administration area.

^Top

REST API Improvements *(improved in Beta 3)*

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues and editing existing ones.
- Delete existing issues and their subtasks.
- Create remote 'issue links'.
- Retrieve metadata from your favourite filters and dashboards (new in Beta 3).
- Retrieve metadata about your permissions (new in Beta 3).
- Almost all system fields and JIRA's built-in custom field types are supported (improved in Beta 3).

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2' (or '`latest`' to use the latest REST API version available with your version of JIRA).

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details. We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

^Top

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

^Top

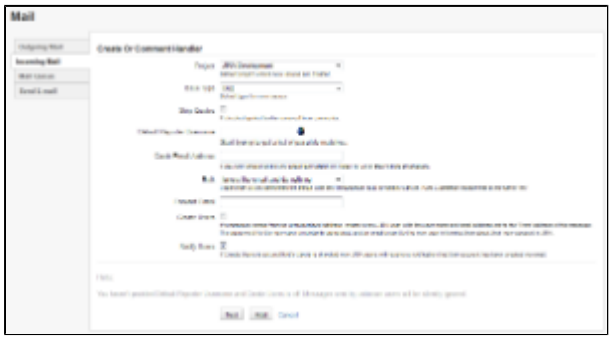
New Troubleshooting and Debugging Tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

- A number of email debugging tools are now provided to System Administrators in the new 'Logging and Profiling' page under 'Troubleshooting and Support'
  - Enable or disable mail logging
  - Turn debug mail logging on or off
  - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New Email Handler Wizard



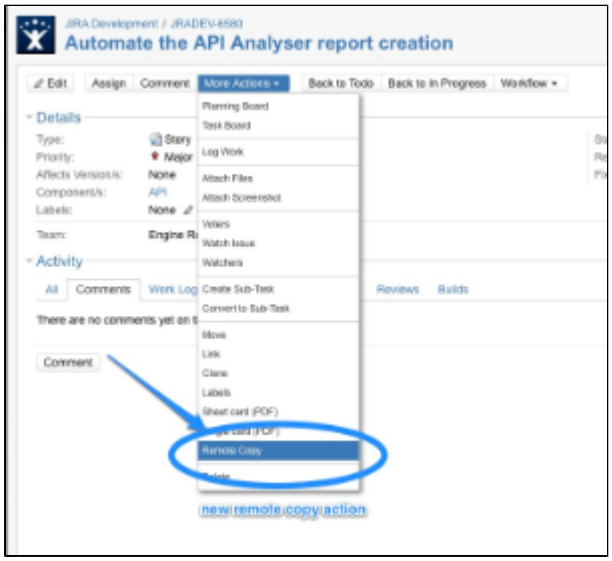
JIRA 5.0 incorporates a new email handler wizard that greatly simplifies the process of configuring incoming mail handlers for creating issue or comments from email messages.

You no longer have to configure a JIRA service and enter in a complex string of Handler parameters for your email handler. Instead you configure your email handler through a convenient wizard.

Improvements have been made to the layout of JIRA's mail configuration options. The configuration options in the 'Mail Servers' tab has beer separated into two tabs — one for 'Outgoing Mail' (SMTP) and another for 'Incoming Mail' (POP/IMAP) configuration. The configuration options in the 'Mail Handlers' tab has been incorporated into the 'Outgoing Mail' tab.

^Top

New Plugin to Try Out — Remote Issue Copying



This new JIRA 5.0-compatible feature currently under development as a plugin, allows you to copy issues from one JIRA site to another.

After establishing an Application Link between your JIRA site and another, a new '**Remote Copy**' action will become available when viewing an issue.

While this action is available to everyone by default, it can be restricted to a particular group.

You will be prompted to map field values by field names for JIRA's built-in (system) fields and/or to configure default values for required fields.

You will require the appropriate permissions to set the field value on the target site.

Custom fields are generally supported, although so far, we have only provided a mapper for the SelectCFType custom field type. Supporting more custom fields is a matter of writing more mappers (which we intend to make pluggable for the final JIRA 5.0 release).

The Remote Issue Copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between. You also need to configure the following before you can copy issues between your JIRA sites:

1. An Application Link between your JIRA sites — see Adding an Application Link.
2. One or more Project Links between JIRA projects across these JIRA sites — Adding Project Links between Applications.

The Remote Issue Copying plugin is not yet bundled with JIRA. However, you can download it from from the following link:

https://maven.atlassian.com/content/groups/public/com/atlassian/cpji/cpji-jira-plugin/2.1-beta1/cpji-jira-plugin-2.1-beta1.jar

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here. Some notable fixes in JIRA 5.0 Beta 3 include:

- A critical issue (JRA-25914) which may cause data corruption in a high-load environment was fixed.

^Top

JIRA 5.0 Beta 2 Release Notes

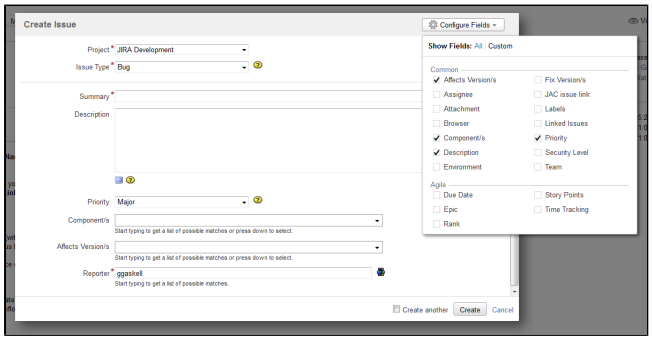
18 October 2011

JIRA 5.0 Beta 2 (a.k.a 5.0 milestone 7 or 'm7') is a public development release leading up to **JIRA 5.0**. A **Beta release** is a preliminary release leading up to the official release of a JIRA version. Beta releases are a snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use Beta releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 5.0 Beta 2 release. Thank you for your feedback during the recent EAP releases and please keep providing it here.

The focus of JIRA 5.0 is on making JIRA easier to use and manage. There are also a large number of improvements for the JIRA developer community. New REST APIs have been added to create issues, a stable JIRA API is being refined and every block area on the 'View Issue' page is now a Web Panel.

Main Highlights of JIRA 5.0 Beta 2:

Create and Edit Issues Rapidly *(improved in Beta 2)*	Remote 'Issue Links' *(improved in Beta 2)*	
		
• Create and edit issues rapidly in a dialog box • Create multiple issues in succession • Customise to show fields you use most often	• Create links to JIRA issues on another JIRA site • Create links to any URL • Create links via JIRA's Java or REST API	

Other Highlights:

- More Enhancements to JQL
- Activity Streams Now Show External Content
- Manage Other Users Shared Filters and Dashboards
- Administration User Interface Improvements ***(improved in Beta 2)***

- REST API Improvements ***(improved in Beta 2)***
- Java API Improvements
- Performance Improvements
- New Troubleshooting and Debugging Tools
- New Email Handler Wizard ***(improved in Beta 2)***
- Remote Issue Copying

Thank you for your interest in JIRA 5.0 Beta 2
Download Beta

Please be aware of known issues specific to this JIRA 5.0 Beta 2 Release.

Upgrading to JIRA 5.0 Beta 2

JIRA EAP releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 5.0 Beta 2

1

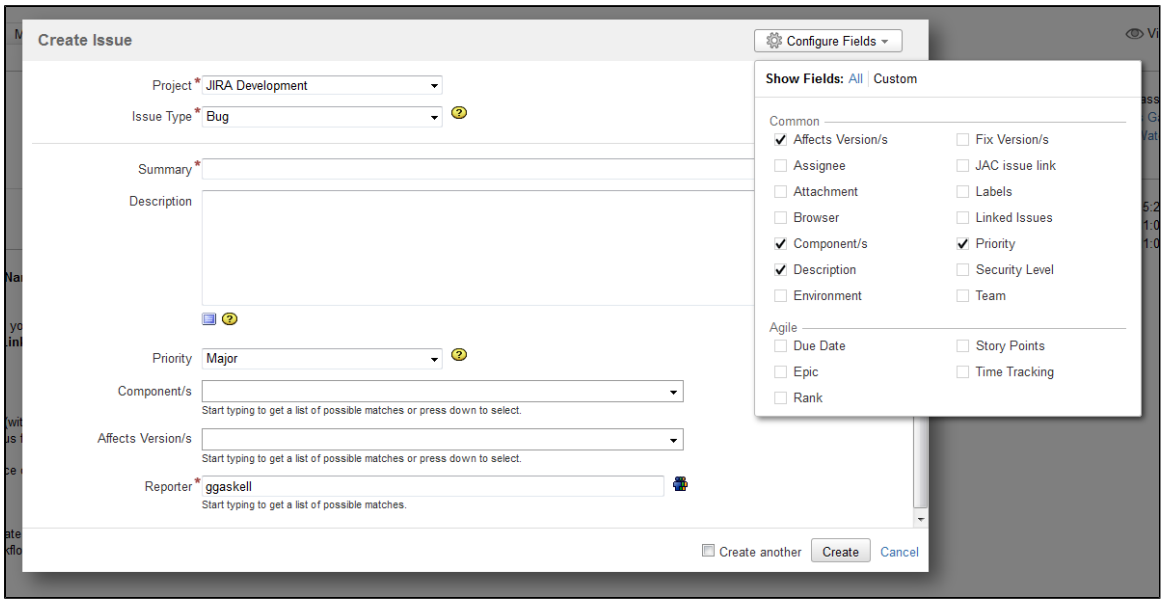
Create and Edit Issues Rapidly *(improved in Beta 2)*

JIRA 5.0 provides a rapid and customisable dialog box for creating and editing JIRA issues, which replaces the Create and Edit Issue forms.

The Create/Edit Issue dialog boxes show all available fields by default. However, you can quickly remove or re-add fields via the '**Configure Fields**' button at the top-right of the dialog box. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly edited fields whenever you create or edit an issue.

If you select the '**Create another**' check box, JIRA will create your issue and automatically pre-populate a new 'Create Issue' dialog box with your previous options, whilst leaving the 'Summary' field blank. This allows you to rapidly create a series of related issues with similar options. Note that this feature does not carry across any attachments that were attached to your previously created issue.

Consistent with JIRA's existing keyboard shortcuts, type '**c**' to access the create issue dialog box or '**e**' from a currently selected issue on the Issue Navigator or 'View Issue' page to access the edit dialog box for that issue.



^Top

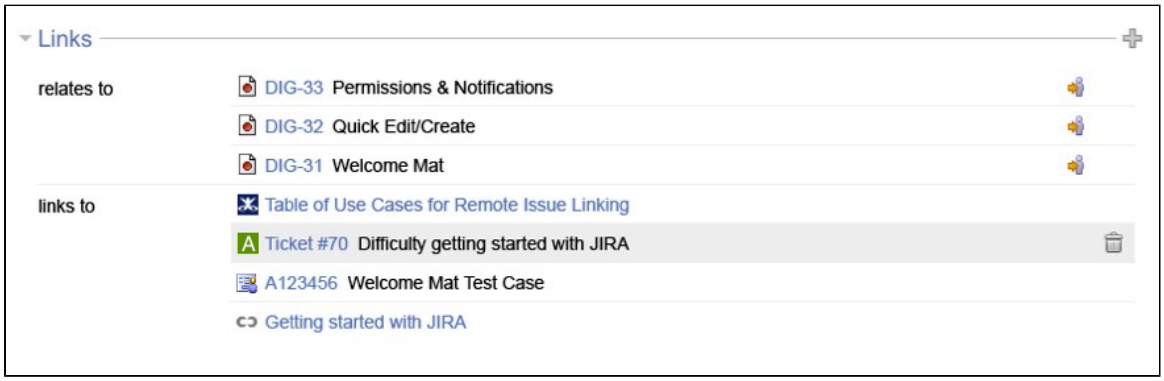
2

Remote 'Issue Links' *(improved in Beta 2)*

The Remote 'Issue Links' feature provides a powerful way to link JIRA issues to items external to your JIRA installation, residing on external applications. Along with a Java and REST API to add these links, end users can also:

- Add a link from a JIRA issue to another issue on a different JIRA installation or site using application links with trusted applications authentication.
  OAuth-based application links will also work but currently possess some minor UI bugs to be fixed.
- Add a reciprocal link from a JIRA issue to another issue on a different JIRA installation or site using application links with either trusted applications or OAuth-based authentication.
- Add a link from a JIRA issue to any web page URL, such as a page of documentation, a technical note, or any other page on another web site.

For more information about this feature, see JIRA Remote Issue Links.



^Top

Sharing Issues and 'Mentions' *(new in Beta 2)*

Want to let others know about a JIRA issue quickly? Simply visit that JIRA issue and then do either of the following:

- Click the new **'Share'** button (or type '**s**') and specify the people you want to share the page with. Add an optional note then click '**Share**'.

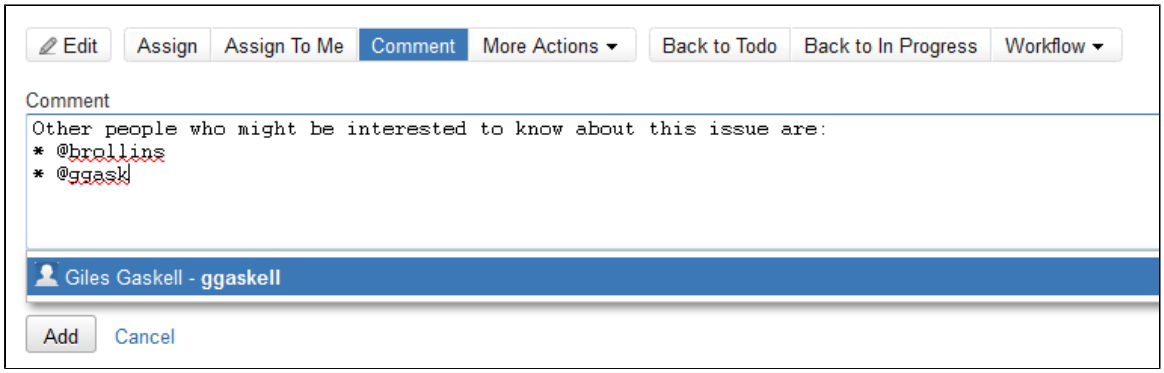
  Screenshot: Sharing an issue with other users

⚠ Please be aware that the 'User name or email' field only accepts JIRA users in Beta 2. Arbitrary email addresses currently cannot be entered.

Users specified in the 'User name or email' field will receive details of the issue in an email message (sent to the addresses registered with their JIRA user accounts). The email message's subject line will indicate that the person who used this feature 'shared' the issue with them.

- In an issue comment, type the '@' symbol and begin typing the name or username of a JIRA user. Select the appropriate user from the dropdown. You can do this with multiple users and after adding your comment, those users will be notified that you mentioned them in a comment.

  Screenshot: Mentioning a user in an issue comment



  Users will receive details of the issue in an email message (sent to the addresses registered with their JIRA user accounts). The email message's subject line will indicate that the person who used this feature 'mentioned' them on that issue.

  The 'Mentions' feature has a few bugs in Beta 2:
  - You currently cannot 'mention' users when editing a comment or using the 'Add Comment' dialog box.
  - Some minor UI bugs are still present.

^Top

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements

Enhancements were introduced in JIRA 4.4 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

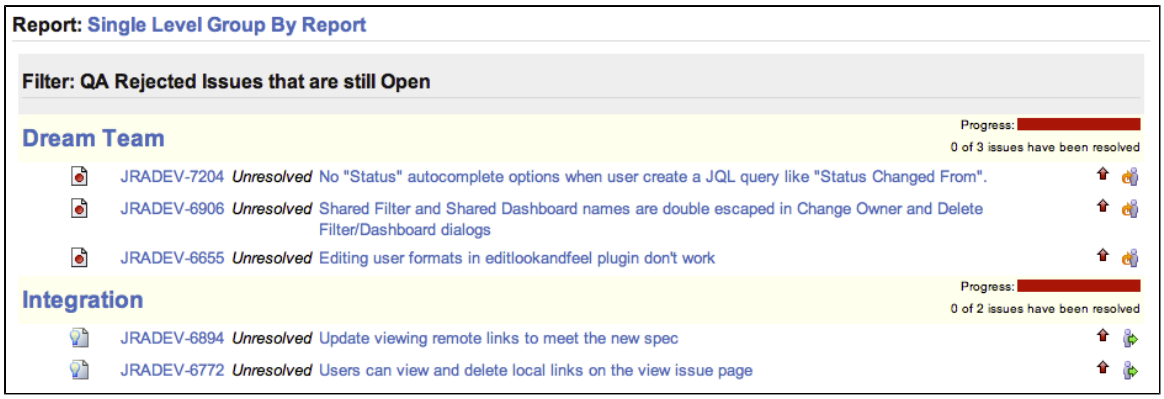
In JIRA 5.0, JQL supports the new "CHANGED" operator, which can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status changed FROM "In QA Review" to "QA Rejected" BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo', and after the start and before the end of the current week.

You can use complex queries such as these to generate the following 'Single Level Group By Report', which in the example below, shows grouping by 'Team'.



The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

The "WAS" operator can now be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

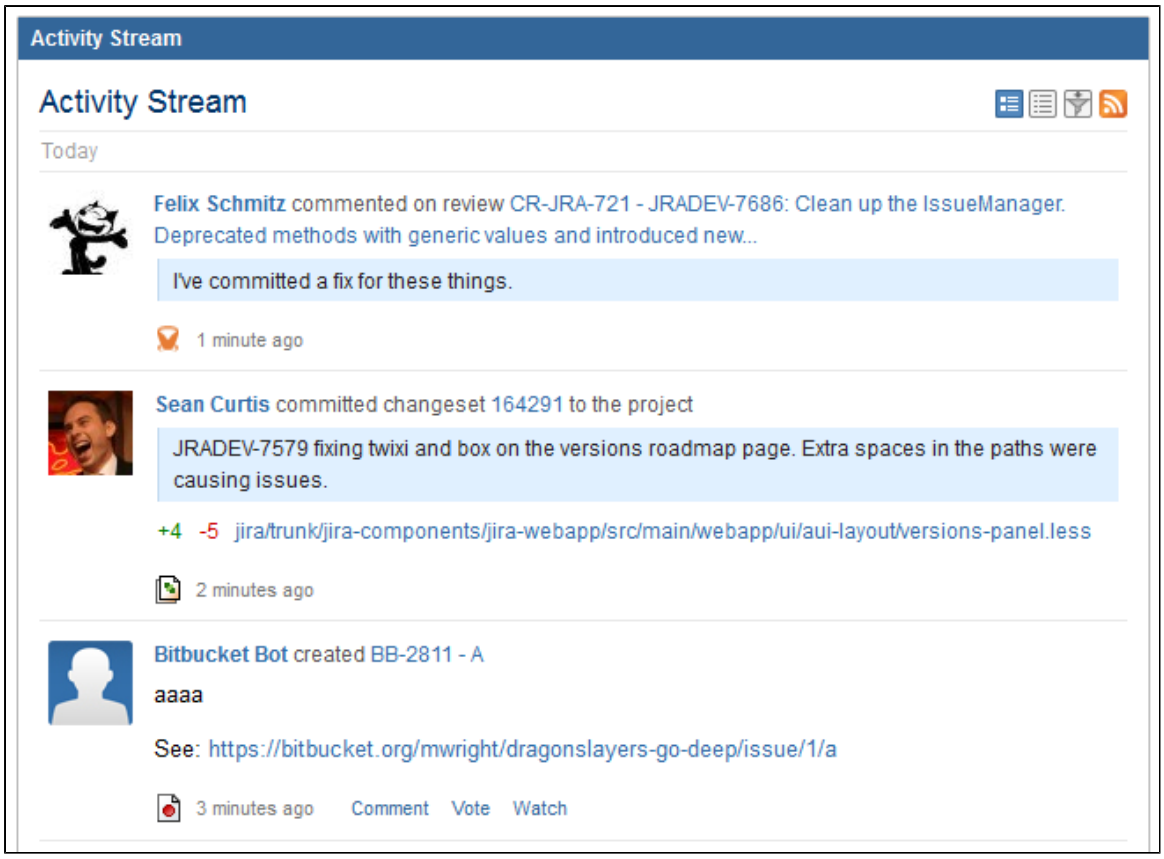
^Top

Activity Streams Now Show External Content

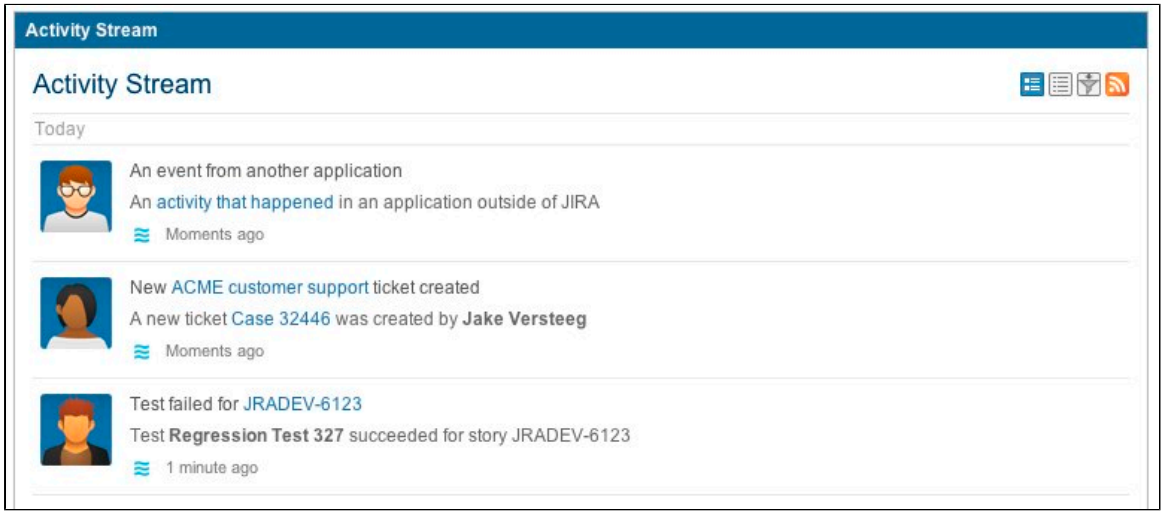
We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Screenshot: Activity stream gadget showing activity from other Atlassian applications



Screenshot: Activity stream gadget showing activity from non-Atlassian applications



Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

^Top

Manage Other Users' Shared Filters and Dashboards

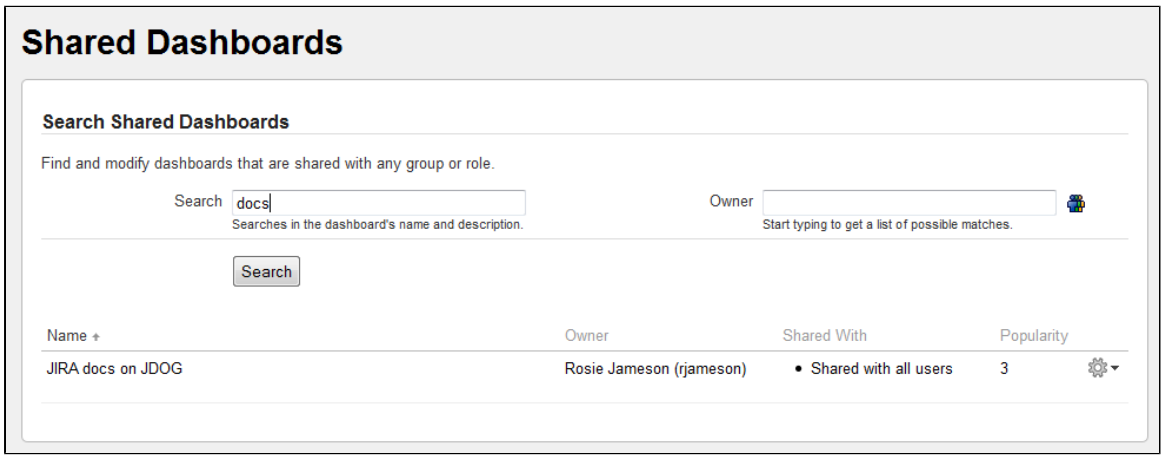
JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboards. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard:



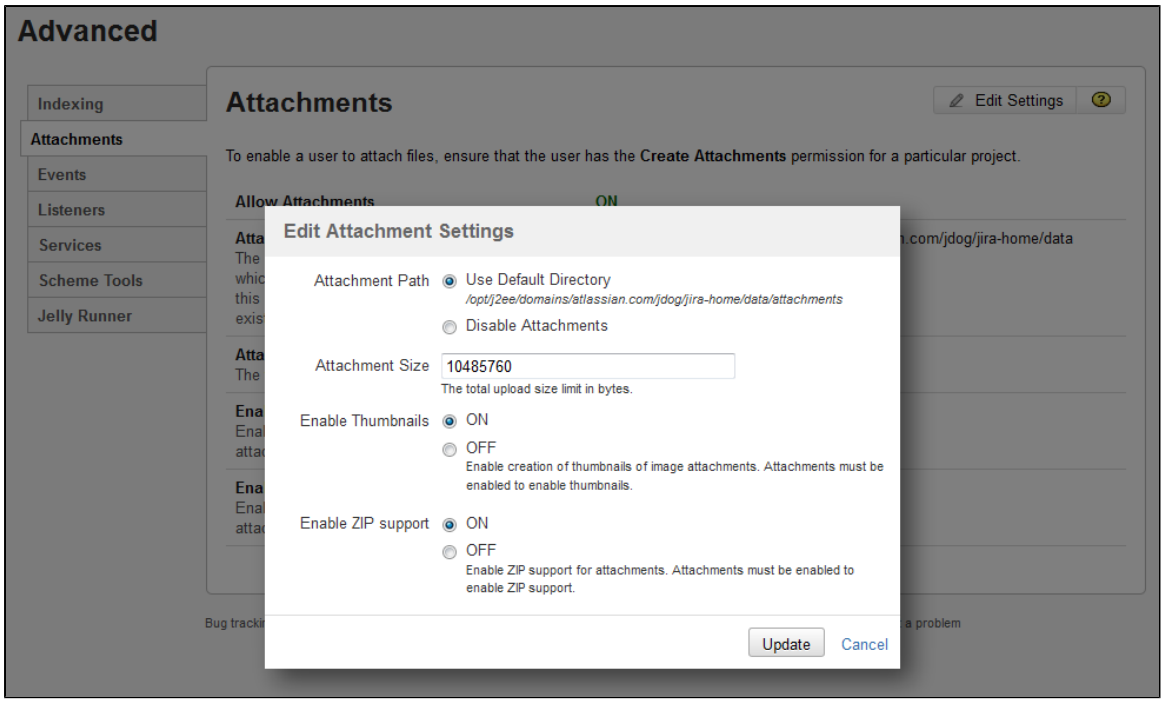
^Top

Administration User Interface Improvements *(improved in Beta 2)*

Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 will provide further improvements to the Administration UI by converting various forms and pages in this area to convenient dialog boxes.

In addition to the dialog box for adding a new user, in Beta 2, the 'Attachments' and 'View Workflows' pages have been redesigned and the forms associated with these pages have been converted into convenient dialog boxes too.

For example, the form for editing attachment settings is now a dialog box, which is accessed by clicking '**Edit Settings**' on the 'Attachments' page of JIRA's Administration area:



^Top

8

REST API Improvements *(improved in Beta 2)*

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues.
- Retrieve metadata for editing existing issues.
- Delete existing issues and their subtasks.
- Create remote issue links.
- More system fields are supported by the REST API.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2'.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details. We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

^Top

9

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.

- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

^Top

New Troubleshooting and Debugging Tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

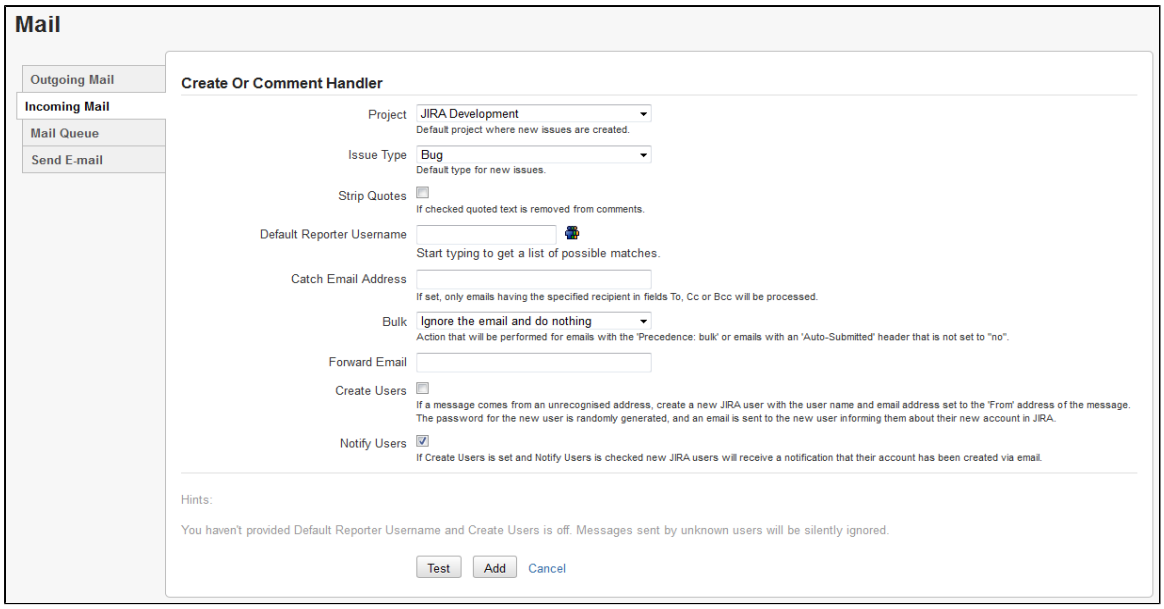
- A number of email debugging tools are now provided to System Administrators in the new 'Logging and Profiling' page under 'Troubleshooting and Support'
  - Enable or disable mail logging
  - Turn debug mail logging on or off
  - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New Email Handler Wizard *(improved in Beta 2)*

JIRA 5.0 incorporates a new email handler wizard that greatly simplifies the process of configuring incoming mail handlers for creating issues or comments from email messages.

You no longer have to configure a JIRA service and enter in a complex string of Handler parameters for your email handler. Instead you configure your email handler through a convenient wizard:



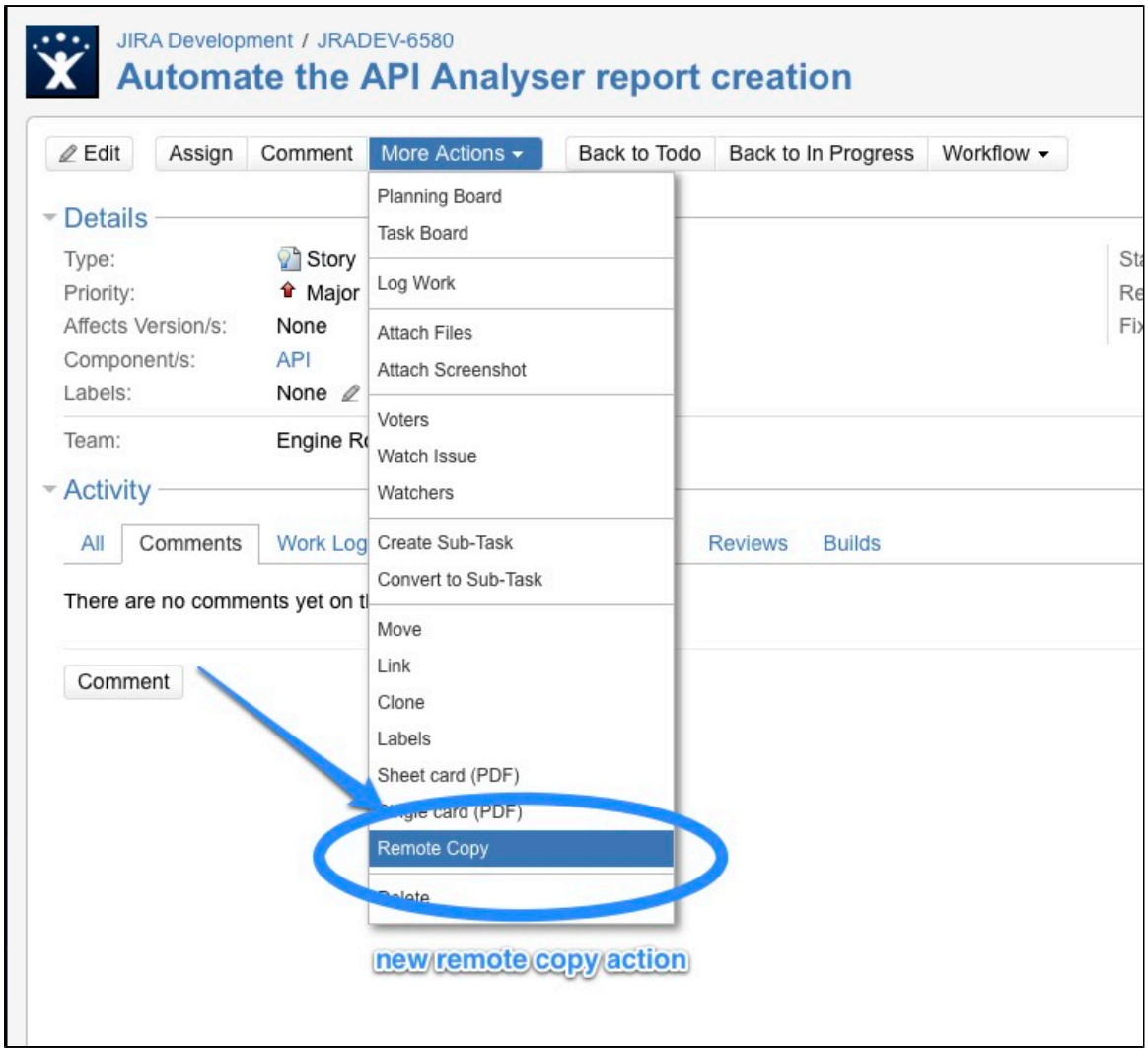
In Beta 2, we've improved the layout of JIRA's mail configuration options. The configuration options in the 'Mail Servers' tab has been separated into two tabs — one for 'Outgoing Mail' (SMTP) and another for 'Incoming Mail' (POP/IMAP) configuration. The configuration options in the 'Mail Handlers' tab has been incorporated into the 'Outgoing Mail' tab.

^Top

New Plugin to Try Out — Remote Issue Copying

This new JIRA 5.0-compatible feature, currently undergoing development as a plugin, allows you to copy issues from one JIRA site to another.

Once you have an Application Link established between your JIRA site and another, a new issue action 'Remote Copy' will appear in the view issue page. You can limit this action to a particular user group, but by default everyone can use it.



You will be prompted to map field values by field names for JIRA's built-in (system) fields and/or to configure default values for required fields.
You will require the appropriate permissions to set the field value on the target site.

Custom fields are generally supported, although so far, we have only provided a mapper for the SelectCFType custom field type. Supporting more custom fields is a matter of writing more mappers (which we intend to make pluggable for the final JIRA 5.0 release).

The Remote Issue Copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between. You also need to configure the following before you can copy issues between your JIRA sites:

1. An Application Link from the source JIRA site to the target JIRA site — see Adding an Application Link.
2. A Project Link from the source JIRA Project to the target JIRA Project — Adding Project Links between Applications.

The Remote Issue Copying plugin is not yet bundled with JIRA. However, you can download it from from the following link:

- https://maven.atlassian.com/content/groups/public/com/atlassian/cpji/cpji-jira-plugin/2.1-beta1/cpji-jira-plugin-2.1-beta1.jar

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

Known Issues in JIRA 5.0 Beta 2

- JIRA 5.0 EAP and Beta releases currently possess a critical issue (JRA-25914) which may cause data corruption in a high-load environment. This issue will be resolved in 5.0 Beta 3. However, as mentioned above, JIRA EAP and Beta releases should not be

used in production environment, nor are they supported.
- When you use the dialog box for creating an issue and you select the '**Create another**' check box before clicking the '**Create**' button, an error will be generated if you had uploaded attachments to that issue.

JIRA 5.0 Beta 1 Release Notes

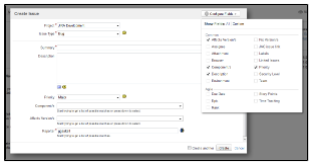
28 September 2011

JIRA 5.0 Beta 1 (a.k.a 5.0 milestone 6 or 'm6') is a public development release leading up to **JIRA 5.0**. A **Beta release** is a preliminary release leading up to the official release of a JIRA version. Beta releases are a snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use Beta releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 5.0 Beta 1 release. Thank you for your feedback during the recent EAP releases and please keep providing it here.

The focus of JIRA 5.0 is on making JIRA easier to use and manage. There are also a large number of improvements for the JIRA developer community. New REST APIs have been added to create issues, a stable JIRA API is being refined and every block area on the 'View Issue' page is now a Web Panel.

Main Highlights of JIRA 5.0 Beta 1:

Customisable Dialog Box for Creating and Editing Issues	Remote Links	More Enhancements to JQL	Activity Streams Now Show External Content
			

Other Highlights:

- REST API Improvements
- Java API Improvements
- Performance Improvements
- New Troubleshooting and Debugging Tools
- New Email Handler Wizard
- Remote Issue Copying

Thank you for your interest in JIRA 5.0 Beta 1
Download Beta

Please be aware of known issues specific to this JIRA 5.0 Beta 1 Release.

Upgrading to JIRA 5.0 Beta 1

JIRA EAP releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 5.0 Beta 1

1

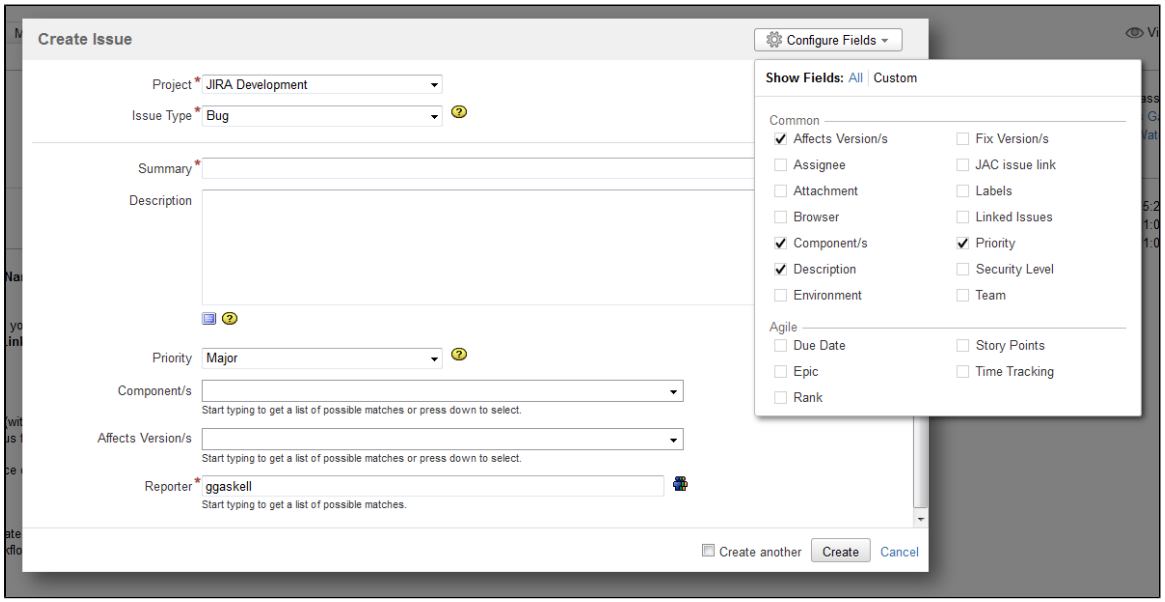
Customisable Dialog Box for Creating and Editing Issues *(new in Beta 1)*

JIRA 5.0 provides a convenient and customisable dialog box for creating and editing JIRA issues, replacing the Create and Edit Issue forms.

The Create/Edit Issue dialog boxes provide a standard set of default fields. However, you can quickly add or remove fields via the '**Configure Fields**' button at the top-right of the dialog box. JIRA remembers your last set of field choices, giving you a personally customised dialog box that presents you with your most commonly edited fields whenever you create or edit an issue.

If you select the '**Create another**' check box, JIRA will create your issue and automatically pre-populate a new 'Create Issue' dialog box with your previous options, whilst leaving the 'Summary' field blank. This allows you to rapidly create a series of related issues with similar options. Note that this feature does not carry across any attachments that were attached to your previously created issue.

Type '**c**' to access the create issue dialog box or '**e**' from a currently selected issue on the Issue Navigator or 'View Issue' page to access the edit dialog box for that issue.



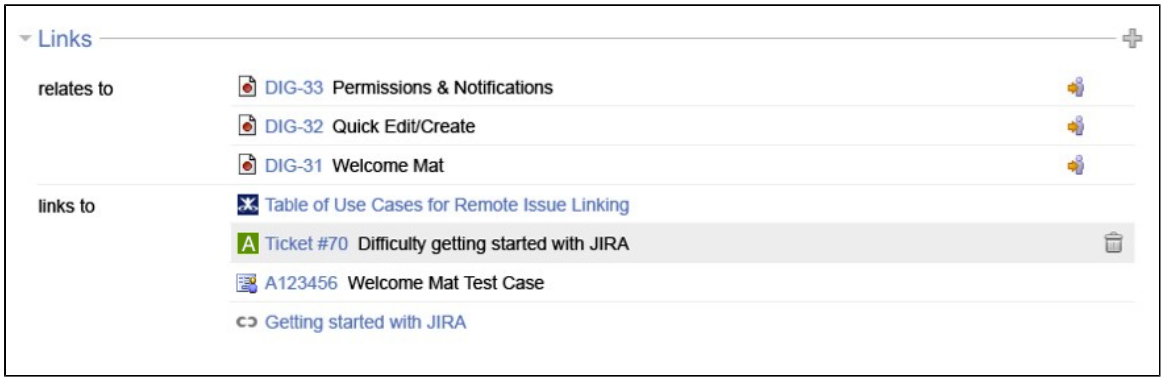
^Top

Remote Links *(improved in Beta 1)*

The Remote Issue Linking feature provides a powerful way to link JIRA issues to items external to your JIRA installation or external applications. Along with a Java and REST API to add these links, end users can also add links from a JIRA issue to:

- another JIRA issue on a different JIRA installation or site using application links with trusted applications authentication.
  OAuth-based application links will also work but currently possess some minor UI bugs to be fixed.
- any web page URL, such as a page of documentation, a technical note, or any other page on another web site.

For more information about this feature, see JIRA Remote Issue Links.



^Top

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements

Enhancements were introduced in JIRA 4.4 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

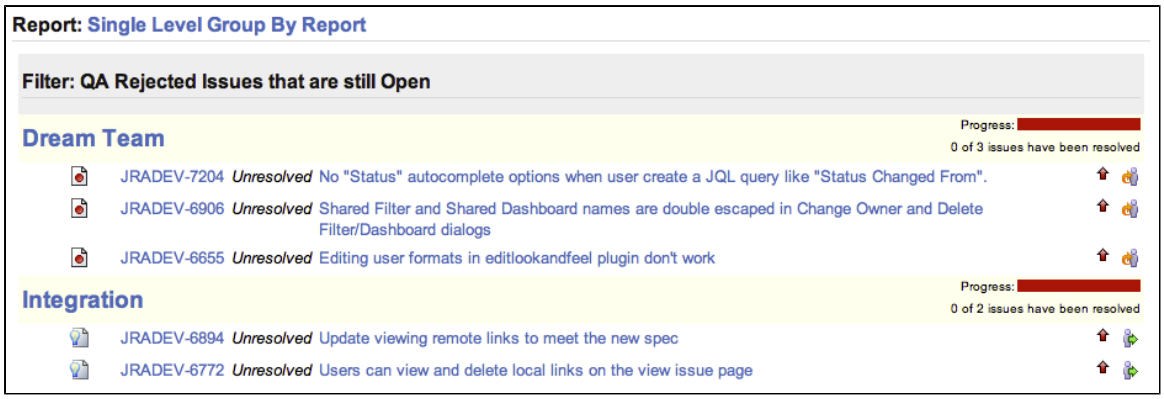
In JIRA 5.0, JQL supports the new "CHANGED" operator, which can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status changed FROM "In QA Review" to "QA Rejected" BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In QA Review" but changed to "QA Rejected", by user 'freddo', and after the start and before the end of the current week.

You can use complex queries such as these to generate the following 'Single Level Group By Report', which in the example below, shows grouping by 'Team'.



The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.

The "WAS" operator can now be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

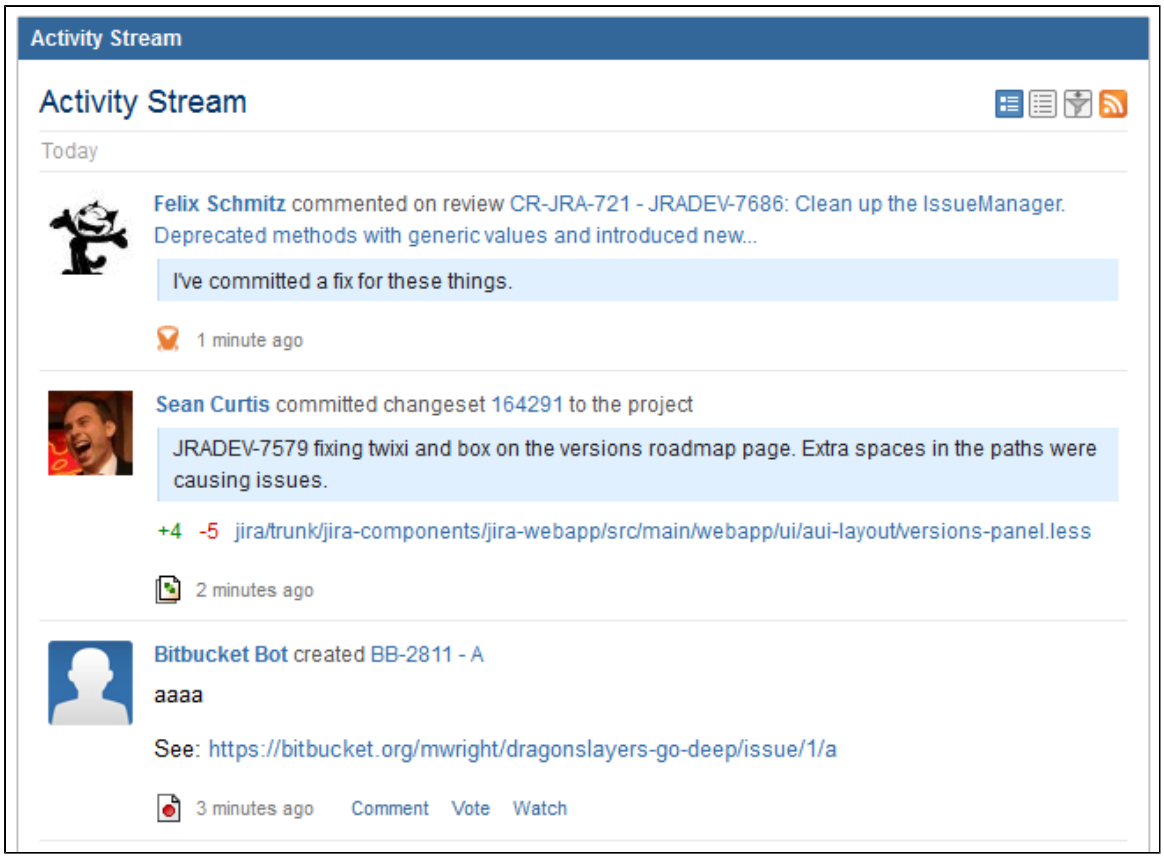
^Top

Activity Streams Now Show External Content

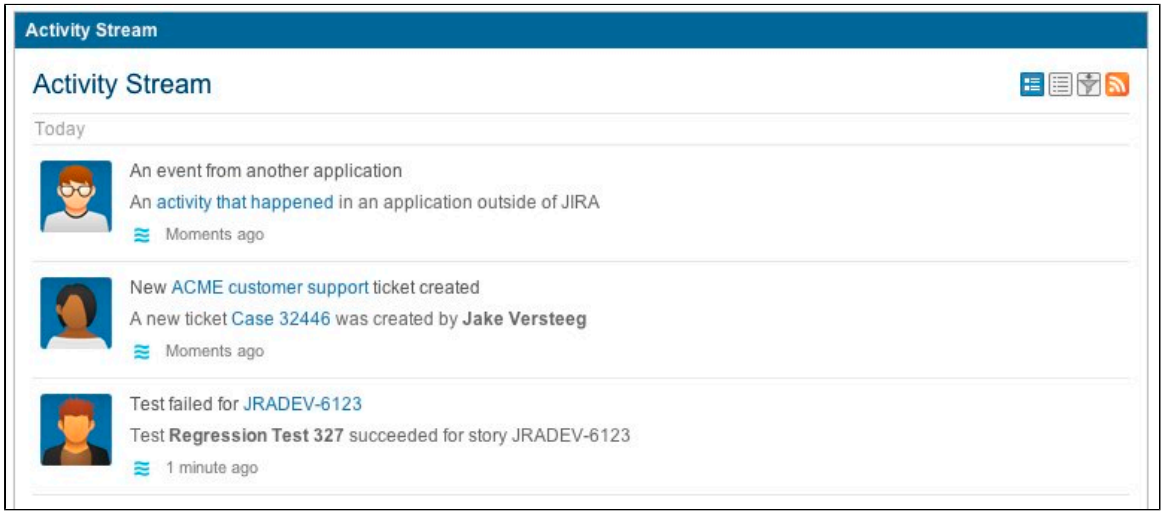
We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Screenshot: Activity stream gadget showing activity from other Atlassian applications



Screenshot: Activity stream gadget showing activity from non-Atlassian applications



Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

^Top

Manage Other Users' Shared Filters and Dashboards

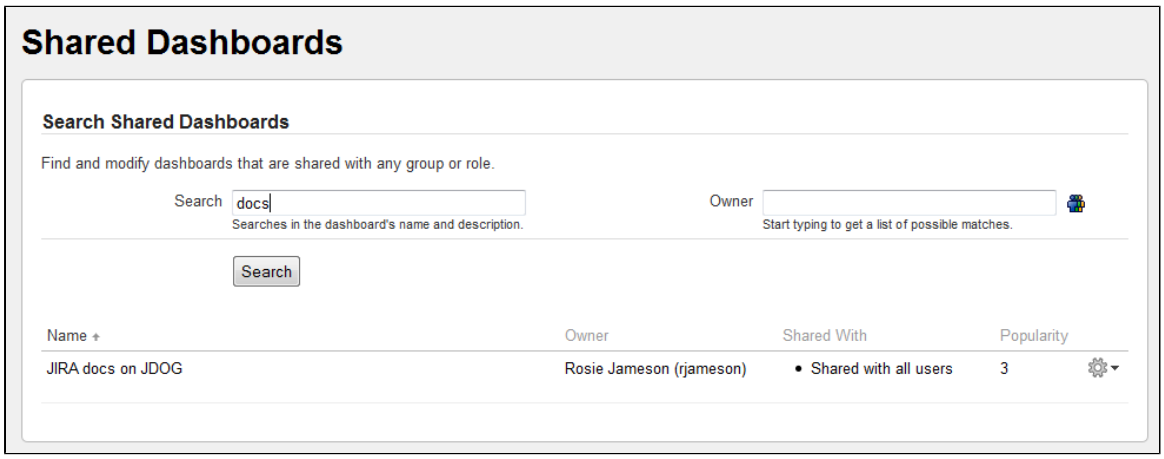
JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboards. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard:



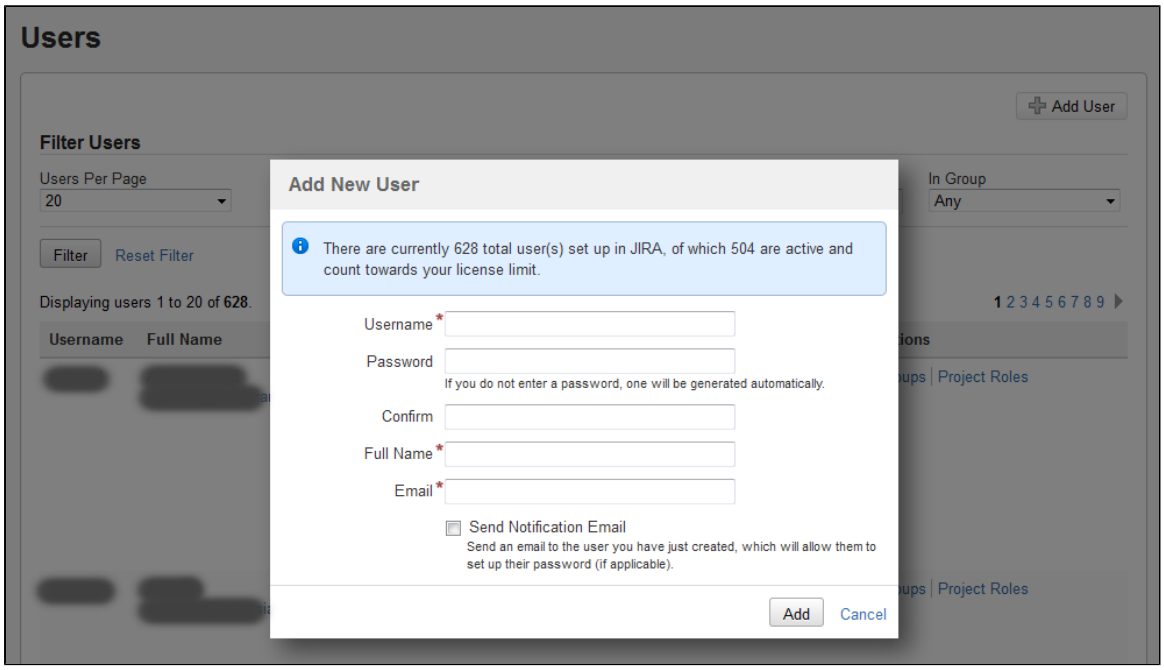
^Top

6

Administration User Interface Improvements

Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 will provide further improvements to the Administration UI by converting various forms and pages in this area to convenient dialog boxes.

The form for adding a new user, accessed by clicking 'Add User' on the 'User Browser' page of JIRA's Administration area, is now a dialog box:



^Top

REST API Improvements

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues.
- Retrieve metadata for editing existing issues.
- Delete existing issues and their subtasks.
- Create remote issue links.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2'.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details. We also have a series of REST API Tutorials to help you get started using our new REST API improvements.

^Top

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Refer to the Preparing for JIRA 5.0 section of our developer documentation site for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

^Top

New Troubleshooting and Debugging Tools

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

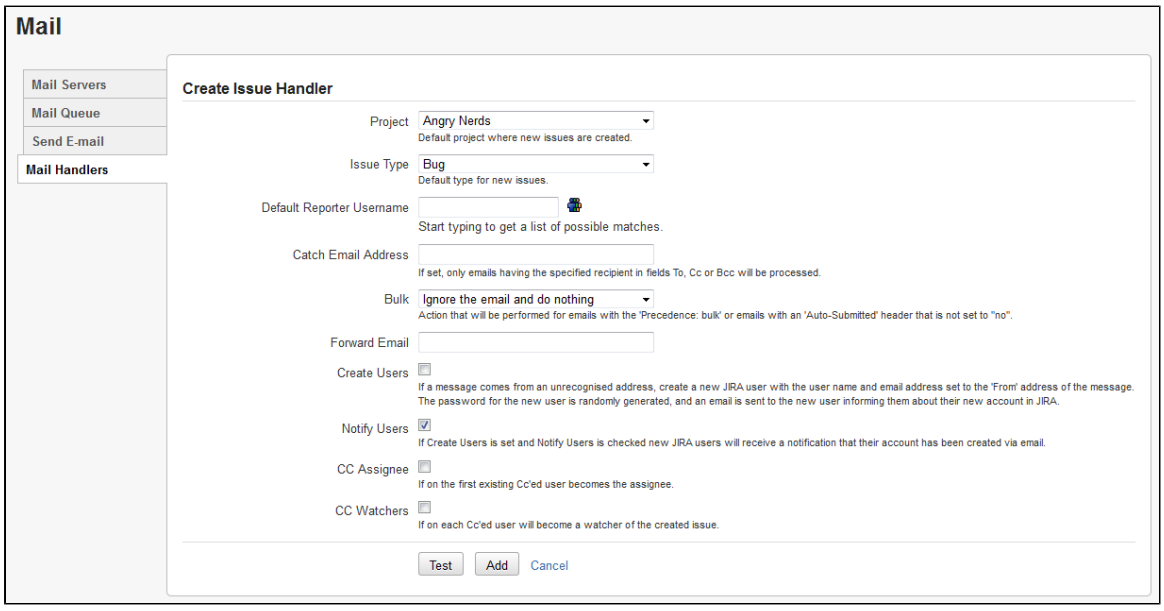
- A number of email debugging tools are now provided to System Administrators in the new 'Logging and Profiling' page under 'Troubleshooting and Support'
    - Enable or disable mail logging
    - Turn debug mail logging on or off
    - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New Email Handler Wizard

JIRA 5.0 incorporates a new email handler wizard that greatly simplifies the process of configuring POP or IMAP mail servers with JIRA for creating issues or comments from email messages.

You no longer have to configure a JIRA service and enter in a complex string of Handler parameters for your email handler. Instead you configure your email handler through a convenient wizard:



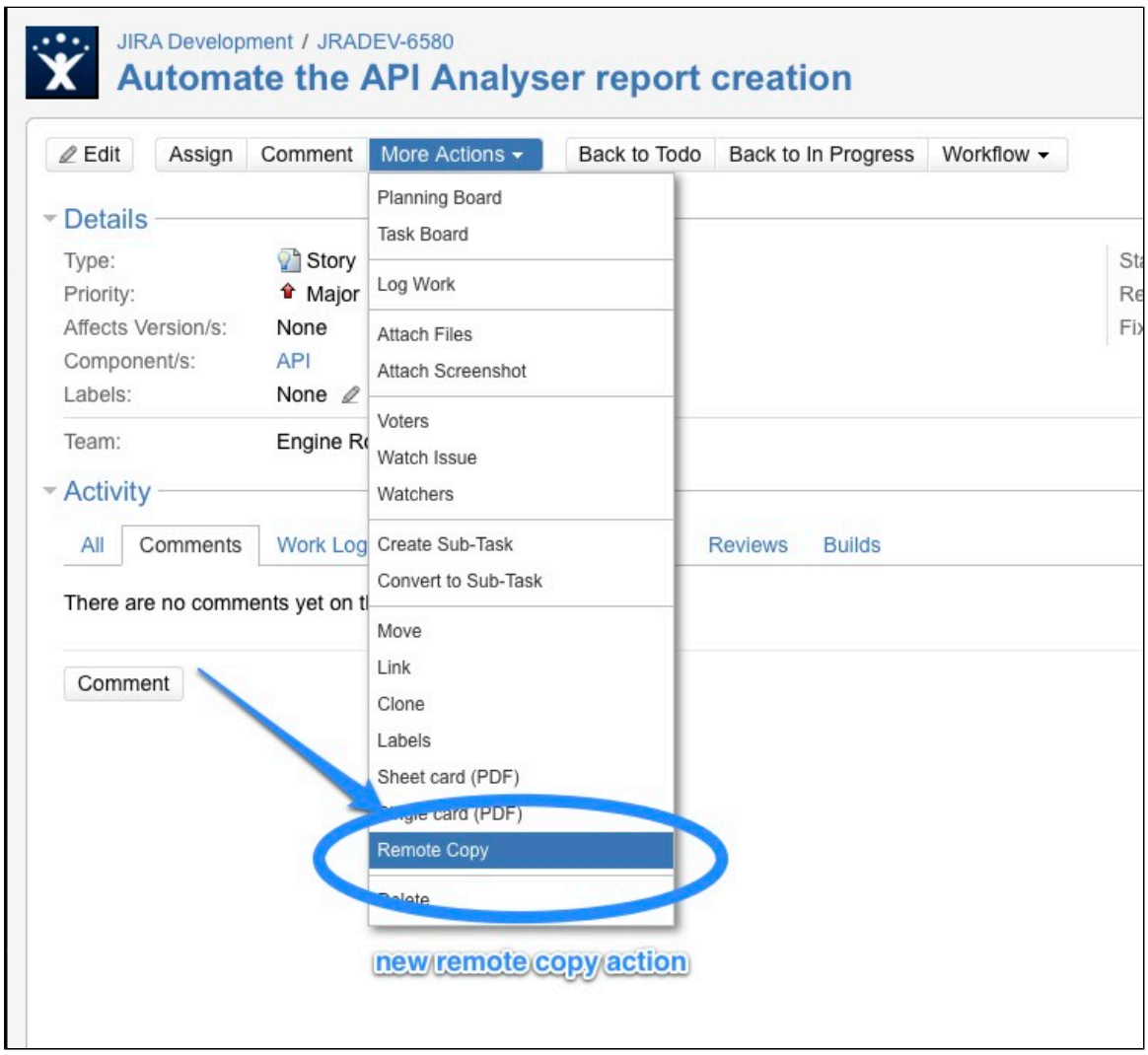
Please be aware that this feature is still undergoing development and as such, the final user experience may change.

^Top

New Plugin to Try Out — Remote Issue Copying

This new JIRA 5.0-compatible feature, currently undergoing development as a plugin, allows you to copy issues from one JIRA site to another.

Once you have an Application Link established between your JIRA site and another, a new issue action 'Remote Copy' will appear in the view issue page. You can limit this action to a particular user group, but by default everyone can use it.



You will be prompted to map field values by field names for JIRA's built-in (system) fields and/or to configure default values for required fields.

You will require the appropriate permissions to set the field value on the target site.

Custom fields are generally supported, although so far, we have only provided a mapper for the SelectCFType custom field type. Supporting more custom fields is a matter of writing more mappers (which we intend to make pluggable for the final JIRA 5.0 release).

The Remote Issue Copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between. You also need to configure the following before you can copy issues between your JIRA sites:

1. An Application Link from the source JIRA site to the target JIRA site — see Adding an Application Link.
2. A Project Link from the source JIRA Project to the target JIRA Project — Adding Project Links between Applications.

The Remote Issue Copying plugin is not yet bundled with JIRA. However, you can download it from from the following link:

- https://maven.atlassian.com/content/groups/public/com/atlassian/cpji/cpji-jira-plugin/2.1-beta1/cpji-jira-plugin-2.1-beta1.jar

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

Known Issues in JIRA 5.0 Beta 1

- If you use a Microsoft SQL Server database with JIRA 5.0 Beta 1, saved filters will not be stored correctly in the database. Hence, whenever JIRA accesses these saved filters, no results will be returned and an error will be generated in JIRA's logs.
- If you add a gadget to a JIRA 5.0 Beta 1 dashboard, which originates from another JIRA installation on version 4.4.x or earlier, the contents of the gadget will not be displayed and will appear to be constantly loading.
- When you use the customisable dialog box for creating an issue and you select the '**Create another**' check box before clicking the '**Create**' button, an error will be generated if you had uploaded attachments to that issue.

JIRA 4.4 RC 1 Release Notes

24 June 2011

JIRA 4.4 RC 1 (a.k.a 4.4 milestone 8 or 'm8') is a public development release leading up to **JIRA 4.4**. A Release Candidate (RC) is a preliminary release leading up to the official release of a JIRA version. RC releases are a fairly stable snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use RC releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 4.4 RC 1 release. Thank you for your feedback during the recent Beta and EAP releases, and please keep providing it here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer, simplified Administration, and user-specific Time Zones. We are also very pleased to announce that the JIRA installation and upgrade processes have been improved and largely automated.

Note to developers: Please see the Plugin Developer Notes for JIRA 4.4.

Highlights of JIRA 4.4 RC 1:

- For Users:
  - User Time Zones
  - New Email Style
  - Issue Linking when Resolving an Issue
  - Workflow Viewer on the 'View Issue' Screen
  - Multiple File Selection and Upload from the 'File Upload' Dialog Box
  - JQL Enhancements ***(expanded since Beta 1)***
  - New-look Activity Stream ***(new since Beta 1)***
  - Graph of Vote History ***(new since Beta 1)***
- For Administrators:
  - Visual Workflow Designer
  - New-Look Administration Area
  - Simplified Project Administration
  - Editable Options for Custom Fields
- Setup and Installation:
  - Improved Setup Wizard with Database Configuration
  - Improved JIRA Standalone Installer/Uninstaller and Automated Upgrade
- Platforms, APIs:
  - IE 9 and Firefox 4 Support
  - REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 RC 1
Download the RC

Upgrading to JIRA 4.4 RC 1

JIRA RC releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

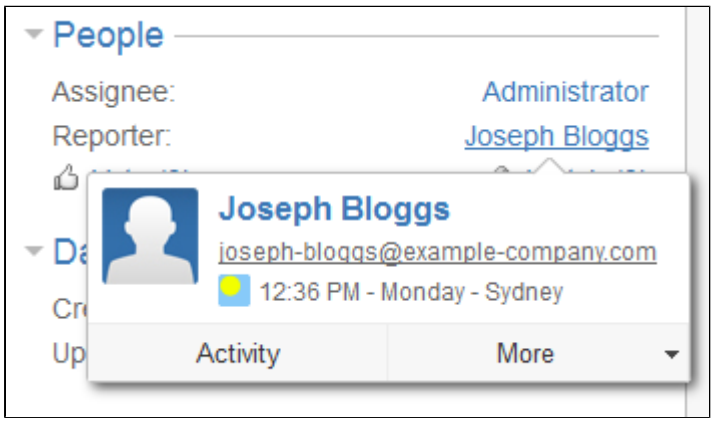
Highlights of JIRA 4.4 RC 1

For Users:

User Time Zones

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: times will be displayed to a user in their local time zone, rather than the server's time zone.

You can set a default user time zone at an administration level, and individual users have the ability to choose their own time zone. Each user's time zone is displayed in their hover profile.

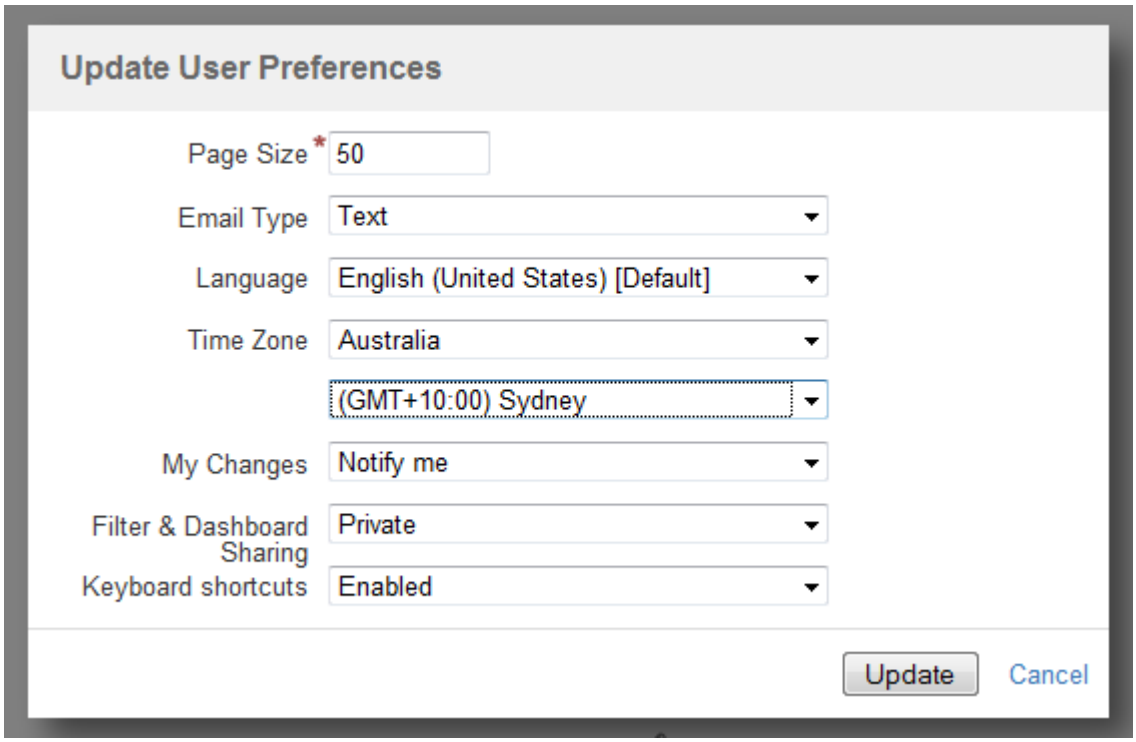


Time zone support has been implemented for quick searching, simple searching and advanced searching, chart and report gadgets, date/time-based custom fields, as well as issue histories, work logs and source code check-ins (via the JIRA FishEye Plugin).

Note: Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information).

JIRA administrators can change the default time zone by going to '**Administration**' => '**General Configuration**' (under '**Global Settings**'), and editing the '**Default user time zone**'.

Users can also change their individual time zone setting via their user profile:

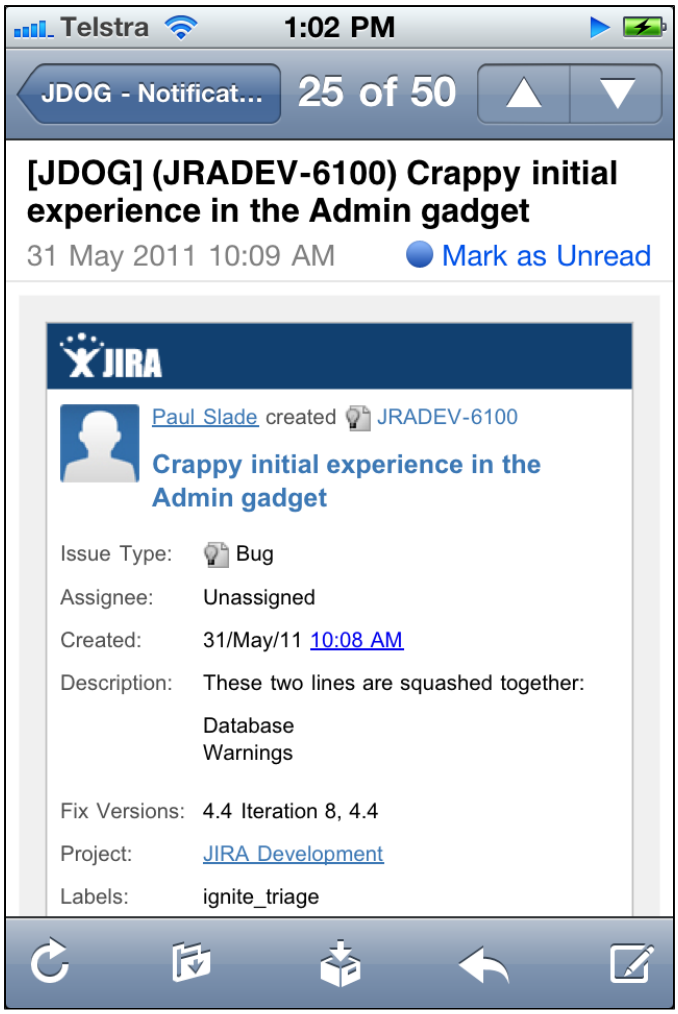


Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

^Top

New Email Style

The HTML email templates have undergone a complete visual refresh. They will also render nicely in mobile mail clients:



They will also thread better in mail clients (such as Gmail), so all emails relating to one issue will thread together.

^Top

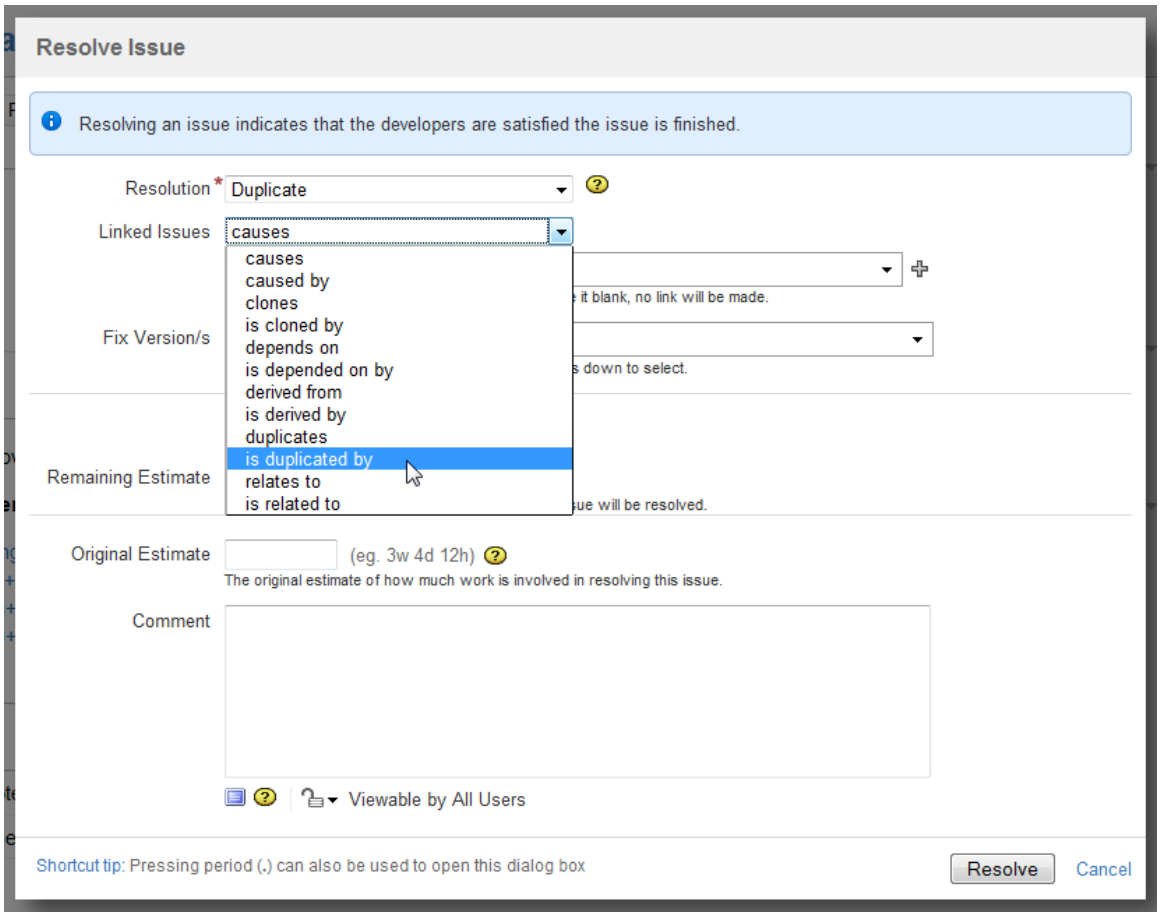
3

Issue Linking when Resolving an Issue

When resolving an issue, you can create links to other issues on an issue resolution screen. This is handy when you want to resolve an issue as a 'duplicate' of another *and at the same time* link to the duplicate issue.

For convenience, your recent issue links and resolutions are readily accessible from the 'Linked Issues' fields (in the screenshot below). The 'Linked Issues' fields can also now be added to any JIRA screen. See Defining a Screen for more information.

If you are upgrading from an earlier version of JIRA, you need to configure this feature manually through JIRA's administration area. Please refer to the Upgrade Notes for details.



^Top

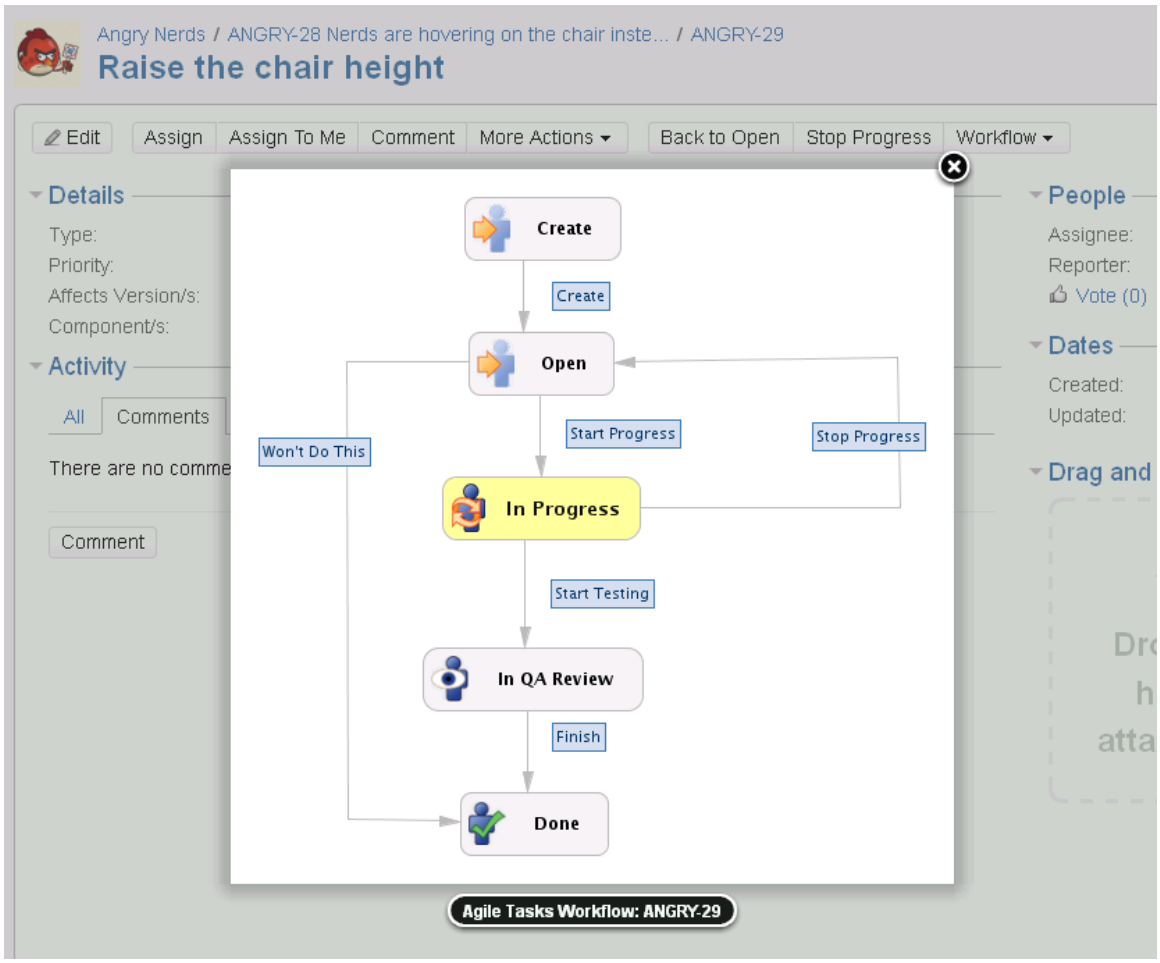
4

Workflow Viewer on the 'View Issue' Screen

You can also see a read-only view of the workflow from the 'View Issue' page — just select '**View Workflow**' from the '**More Actions**' menu. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View Read-Only Workflow' is required to access the workflow viewer feature from the 'View Issue' page.

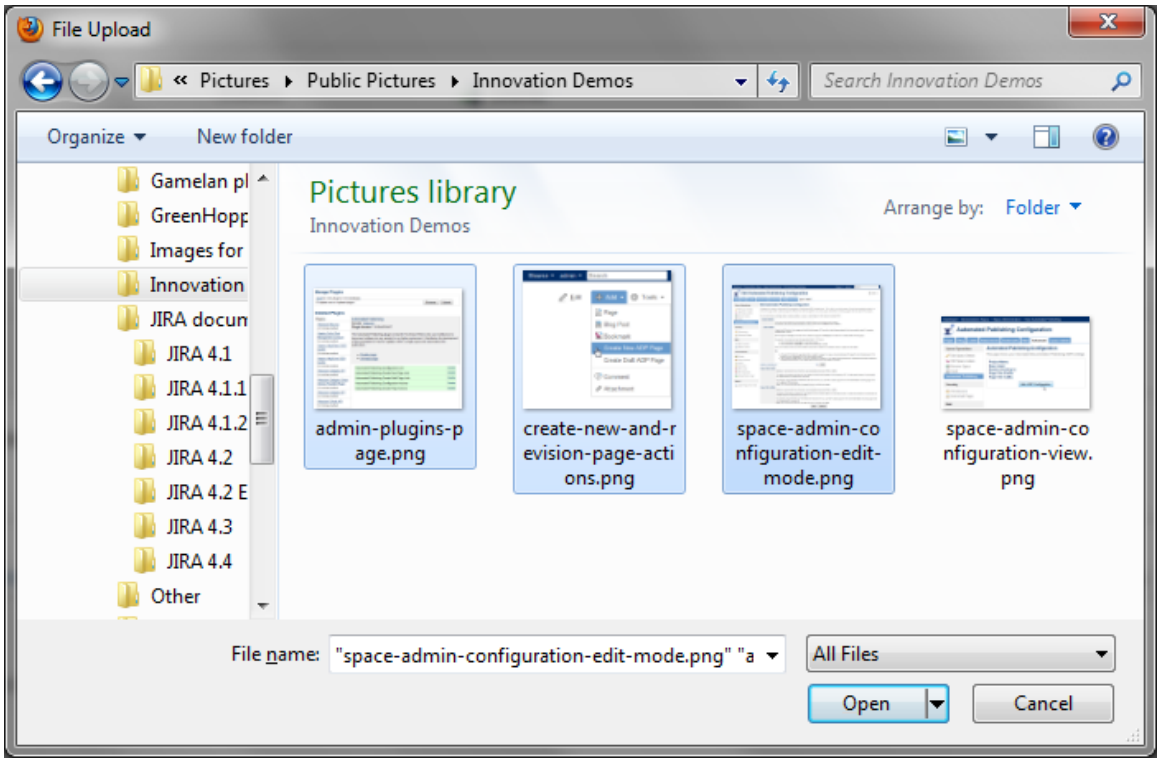


^Top

5

Multiple File Selection and Upload from the 'File Upload' Dialog Box

When using JIRA's 'Attach Files' dialog box, you can now select multiple files in the 'File Upload' dialog box that appears after clicking the 'Browse' button.



^Top

JQL Enhancements (expanded since Beta 1)

Do more with 'WAS'

The ability to search an issue's Change History was introduced in JIRA 4.3 and allowed you to search the history of the **Status** field.

In JIRA 4.4 you can now also search the history of:

- the **Assignee** field, e.g.:

```
assignee WAS "jsmith"
```

- the **Reporter** field, e.g.:

```
reporter WAS "djones"
```

You can also now search for a field that had a particular value:

- **ON** a given date — e.g. find issues that had a status of "closed" on May 31st:

```
status WAS "closed" ON "2011/05/31"
```

- **BEFORE** a given date — e.g. find issues that were assigned to jsmith before May 31st:

```
assignee WAS "jsmith" BEFORE "2011/05/31"
```

- **AFTER** a given date — e.g. find issues that were assigned to me after May 31st:

```
assignee WAS currentUser() AFTER "2011/05/31"
```

- **DURING** a given date range — e.g. find issues that were assigned to me during May:

```
assignee WAS currentUser() DURING ("2011/05/01","2011/05/31")
```

- set **BY** a particular user — e.g. find issues that were assigned to Fred by me:

```
assignee WAS "Fred" BY currentUser()
```

How many Watchers?

The new Watchers field allows you to search for issues with a specified number of watchers, e.g.:

```
watchers > 3
```

For more details please see Advanced Searching.

^Top

New-look Activity Stream (new since Beta 1)

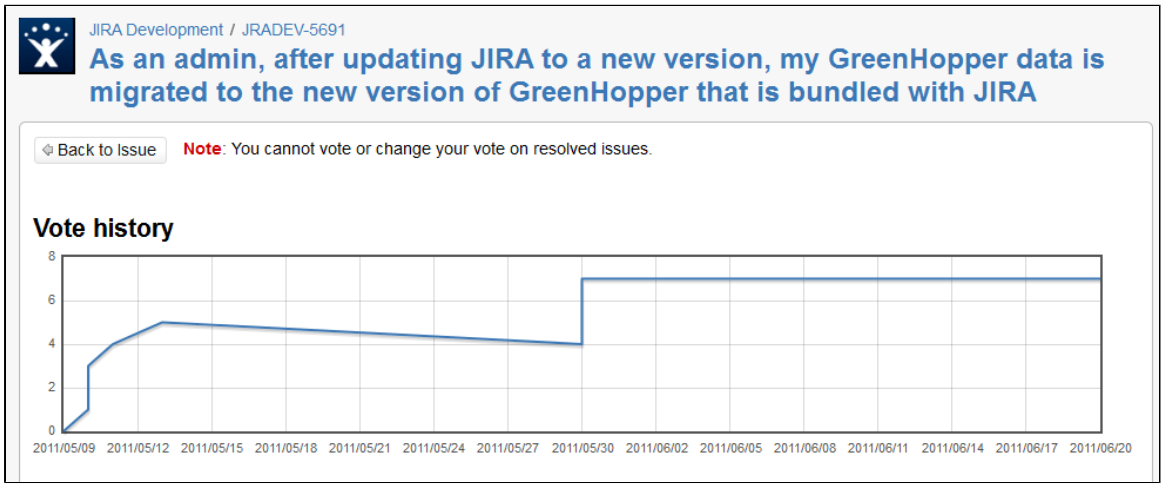
The Activity Stream has had a makeover. As well as looking prettier, it now lets you vote, start watching or comment on an issue with a single click:

Activity Stream
Activity Stream
Tuesday
Andrew Prentice attached one file to ANGRY-73 - bonfire test
iCloud
Tuesday at 5:23 AM Comment Vote Watch
Monday
Mike Cannon-Brookes commented on ANGRY-27 - Nerd flight path does not take into account gravitational pull of moon
Are we building the physics engine for Newtonian or Einsteinian physics?
Monday at 2:15 AM Reply Watch
Sunday
Edwin Wong linked 3 issues
ANGRY-67 - As a nerd, I can flip in a chair while waiting to be shot into the air
is related to
ANGRY-70 - Some graphical glitches and ANGRY-69 - The nerd flies too fast
Sunday at 12:13 AM Vote Watch
Edwin Wong updated 2 fields of ANGRY-67 - As a nerd, I can flip in a chair while waiting to be shot into the air
Logged '2h'
Changed the Remaining Estimate to '0h'
Sunday at 12:13 AM Comment Vote Watch
Edwin Wong attached one file to ANGRY-70 - Some graphical glitches
ANGRY NERDS
Sunday at 12:05 AM Comment Vote Watch
Show more...

^Top

Graph of Vote History (new since Beta 1)

You can now graph an issue's votes over time:



^Top

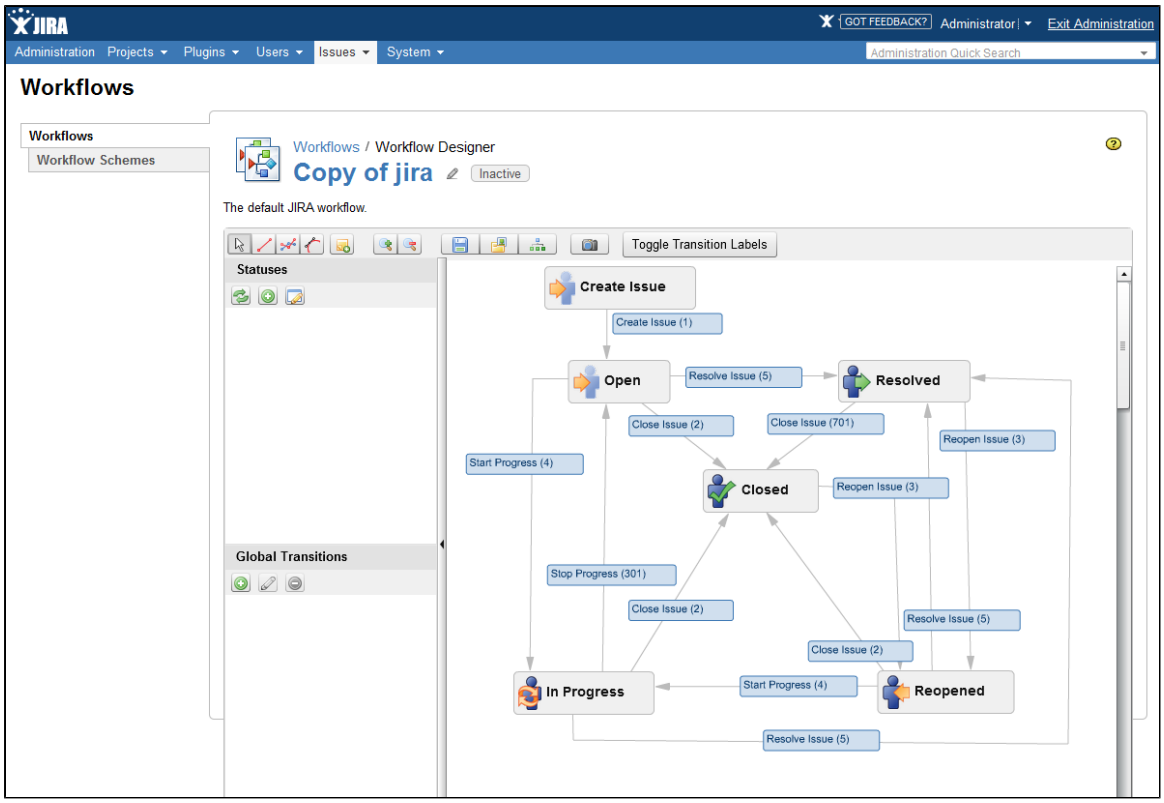
For Administrators:

Visual Workflow Designer

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to '**Workflows**' in JIRA administration as usual, and click the '**Design**' link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.

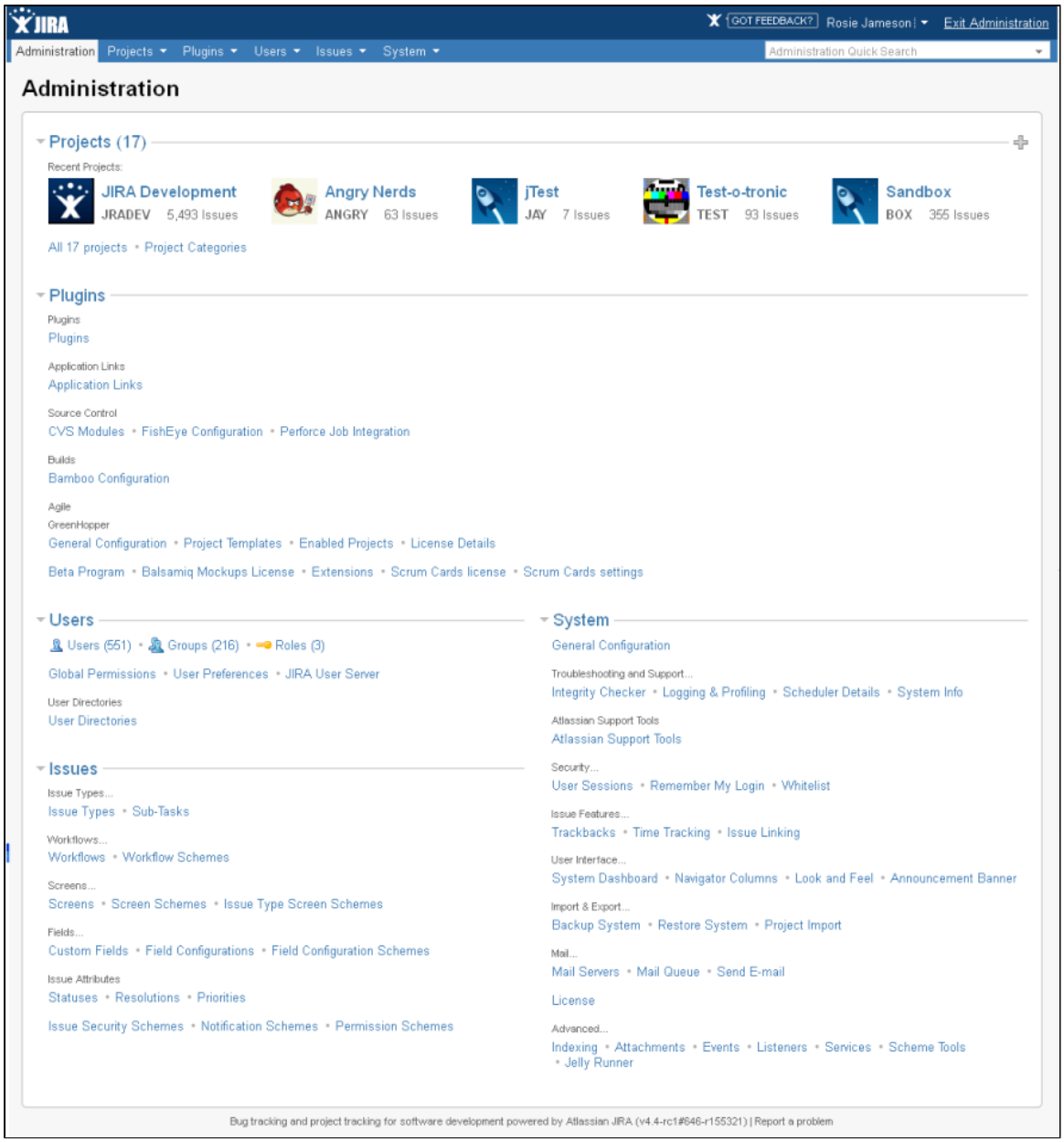
The layout of a workflow is preserved whenever you 'Copy' or 'Create a Draft' of an existing workflow.



^Top

New-Look Administration Area

JIRA 4.4 brings you a dedicated 'administration mode', which replaces the left-hand column of the JIRA's administration console with a series of drop-down menus across the top navigation bar:



To find the new location of a menu item, type it into the **"Administration Quick Search"** box at the top right of the screen — or click the drop-down in the **"Administration Quick Search"** box to get a full list of admin options.

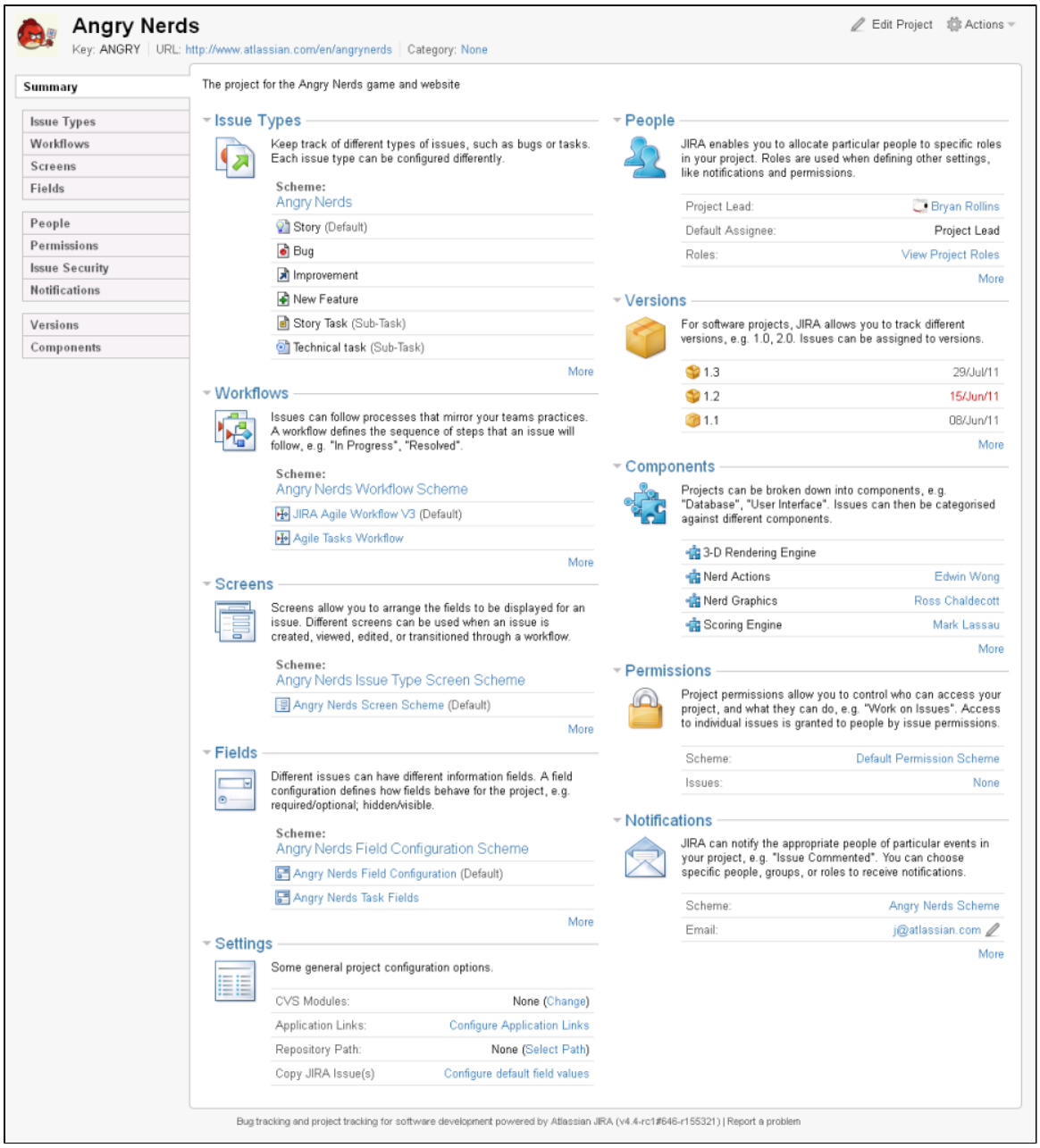
You can bring up the **"Administration Quick Search"** box from anywhere in JIRA by typing **G + G**. This has replaced the **A** keyboard shortcut.

To leave JIRA's 'administration mode', click the '**Exit Administration**' link at the top-right of the screen to return JIRA to its standard user mode.

^Top

Simplified Project Administration

Project administration has become more visual and far simpler:



^Top

Editable Options for Custom Fields

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

^Top

Setup and Installation:

Improved Setup Wizard with Database Configuration

Our trusty setup wizard has had a makeover:

The new Setup Wizard

^Top

Database Configuration Now Part of the Setup Wizard

In JIRA 4.4, configuring a connection to an external database is now part of the standard setup wizard. Upon completing the setup wizard, JIRA will create a direct JDBC connection (whose entire configuration is stored within your JIRA home directory).

Here is the database configuration step of the JIRA Setup Wizard:

^Top

Improved JIRA Standalone Installer/Uninstaller and Automated Upgrade

Linux Installer and Uninstaller

The Linux Installer provides a simple console (shell prompt) wizard that installs JIRA for Linux operating systems. The Linux Installer:

- Installs JIRA under a dedicated user account 'jira' with restricted write access to your JIRA installation directory.
- Can install JIRA as a service, so that JIRA automatically re-starts whenever your Linux operating system must be rebooted.

To install JIRA, simply download the Linux Installer (`.bin`) file and at a shell prompt, execute this file and follow the remaining prompts!

See Installing JIRA on Linux for details.

After using the Linux Installer, an executable file 'uninstall' (located in your JIRA Installation Directory) is available to conveniently uninstall JIRA from your Linux operating system.

See Uninstalling JIRA from Linux for details.

Windows Installer

The installation wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

See Installing JIRA on Windows for details.

Unattended Installation

When installing JIRA using the Linux or Windows Installers (above), a configuration file called `response.varfile` is generated in the `.install4j` subdirectory of your JIRA Installation Directory.

The `response.varfile` file records all configuration options specified during your initial installation. This allows you to reinstall JIRA on multiple server machines based on the same configuration without the need for any user input.

See the 'Performing an Unattended Installation' sections for Linux and Windows for details.

Automated Upgrade

The new Linux and Windows Installers (above) include an option that allows you to upgrade an existing JIRA 4.3.x or later Standalone installation.

This upgrade feature automates the following tasks for you:

1. Backs up the Installation and Home Directories of the existing JIRA installation to be upgraded.
2. Installs **JIRA 5.0.x** whilst migrating the following from your existing JIRA installation to the new **JIRA 5.0.x** installation:
    - Legacy database configurations defined as a datasource within the application server (used in JIRA 4.3.x and earlier) to the new database configuration used in JIRA 4.4 and later. See JIRA 4.4 Upgrade Notes for details.
    - TCP port values in your existing JIRA installation's `server.xml` file.
    - Custom values in your existing JIRA installation's `jira-application.properties`, including key customisations from the `setenv.sh` / `setenv.bat` files.

The upgrade feature detects and notifies you of any other files in the `atlassian-jira` subdirectory of your existing JIRA Installation Directory, which had been deleted, added or modified from a 'default' JIRA installation. This informs you of any customisations you will need to migrate manually over to your upgraded JIRA installation directory.

Also note that the JIRA Configuration Tool (bundled with JIRA Standalone) is now capable of changing JIRA's TCP Ports.

^Top

Platforms, APIs:

IE 9 and Firefox 4 Support

We are very pleased to announce that JIRA 4.4 supports Internet Explorer 9.0 and Firefox 4.0.

^Top

REST API improvements

There are new REST APIs for

- Listing and managing Project Components.
- Listing and managing Project Versions.
- Listing and managing Project Roles.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.4 Beta 1 Release Notes

6 June 2011

JIRA 4.4 Beta 1 (a.k.a 4.4 milestone 7 or 'm7') is a public development release leading up to **JIRA 4.4**. A Beta release is a preliminary release leading up to the official release of a JIRA version. Beta releases are a snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use Beta releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 4.4 Beta 1 release. Thank you for your feedback during the recent EAP releases, and please keep providing it here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer, simplified Administration, and user-specific Time Zones. We are also very pleased to announce that the JIRA installation and upgrade processes have been improved and largely automated.

Note to developers: Please see the Plugin Developer Notes for JIRA 4.4.

Highlights of JIRA 4.4 Beta 1:

- Visual Workflow Designer For Administrators
- Workflow Viewer on the 'View Issue' Screen
- User Time Zones
- Editable Options for Custom Fields
- Simplified Project Administration ***(new since EAP 6)***
- New-Look Administration Area ***(improved since EAP 6)***
- Classy New Email Templates ***(new since EAP 6)***
- Issue Linking when Resolving an Issue
- Multiple File Selection and Upload from the 'File Upload' Dialog Box
- Enhancements to JQL "WAS" Function ***(new since EAP 6)***
- Improved Setup Wizard
- Database Configuration Now Part of the Setup Wizard
- Improved JIRA Standalone Linux Installer with Uninstall and Automated Upgrade ***(improved since EAP 6)***
- Improved JIRA Standalone Windows Installer with Automated Upgrade ***(improved since EAP 6)***
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 Beta 1
Download Beta

Upgrading to JIRA 4.4 Beta 1

JIRA Beta releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.4 Beta 1

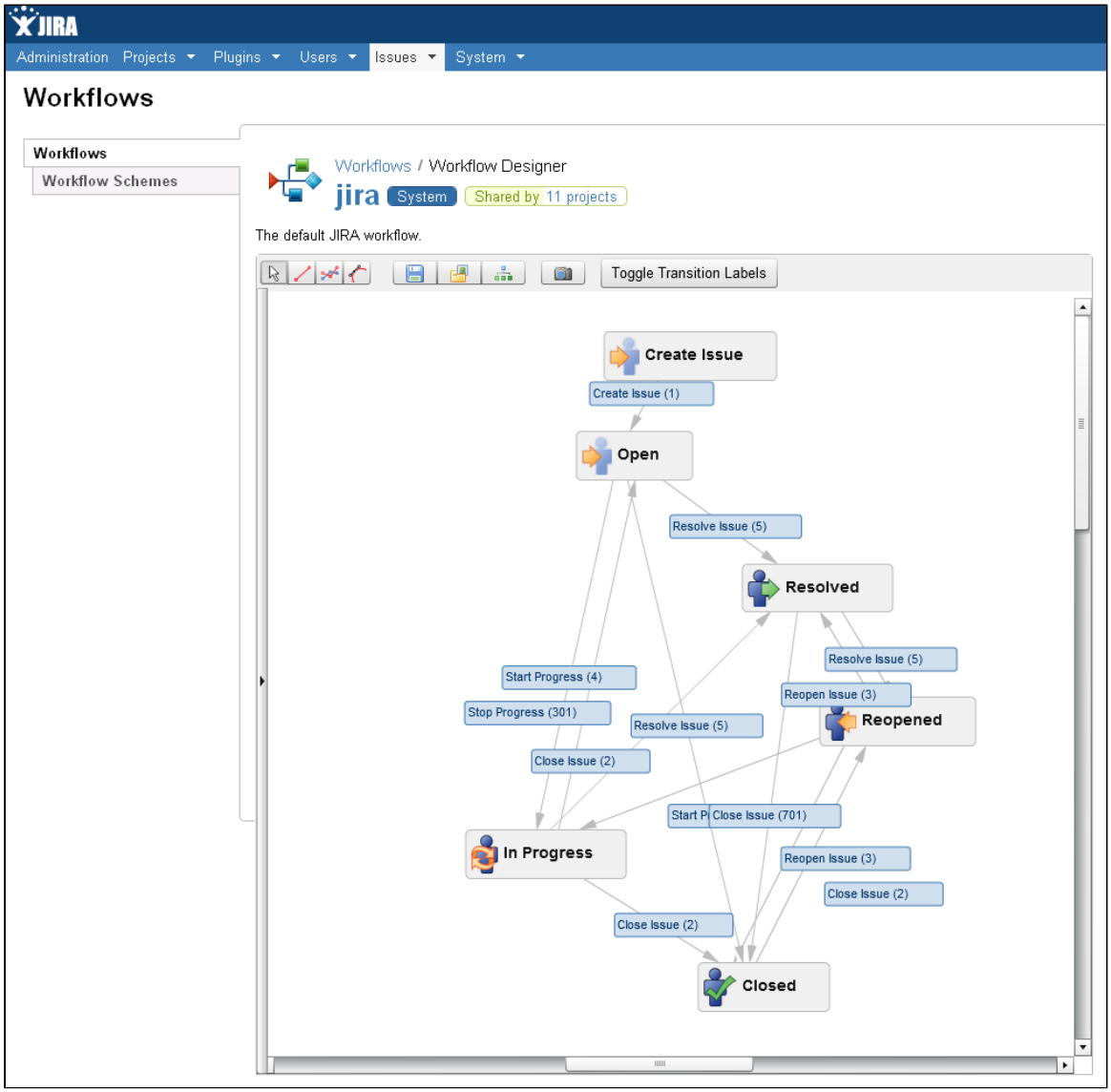
Visual Workflow Designer For Administrators

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to '**Workflows**' in JIRA administration as usual, and click the '**Design**' link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.

In EAP 5, the layout of a workflow is now preserved whenever you 'Copy' or 'Create a Draft' of an existing workflow. We have also

implemented a few improvements to the status editor.



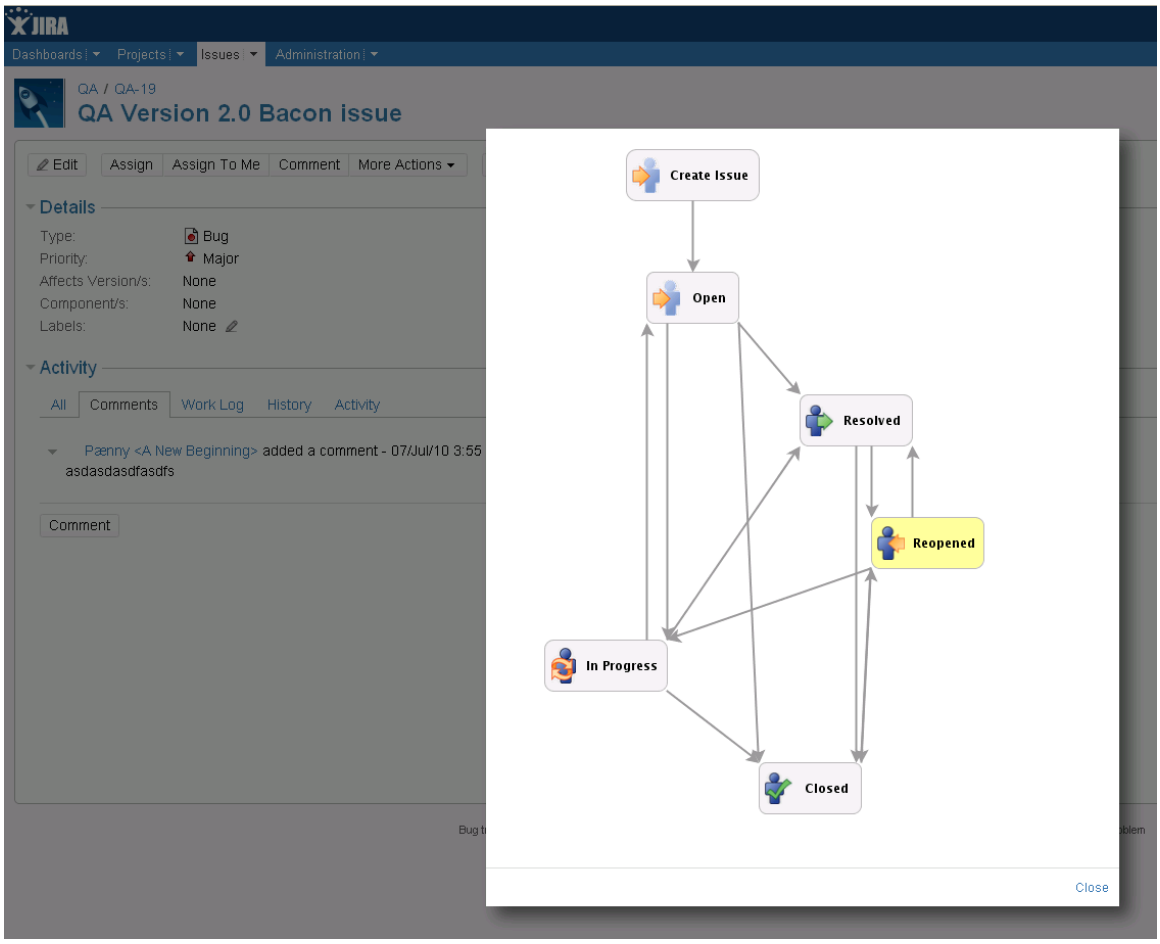
^Top

Workflow Viewer on the 'View Issue' Screen

You can also see a read-only view of the workflow from the 'View Issue' page — just select '**View Workflow**' from the '**More Actions**' menu. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View Read-Only Workflow' is required to access the workflow viewer feature from the 'View Issue' page.



^Top

User Time Zones

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: times will be displayed to a user in their local time zone, rather than the server's time zone.

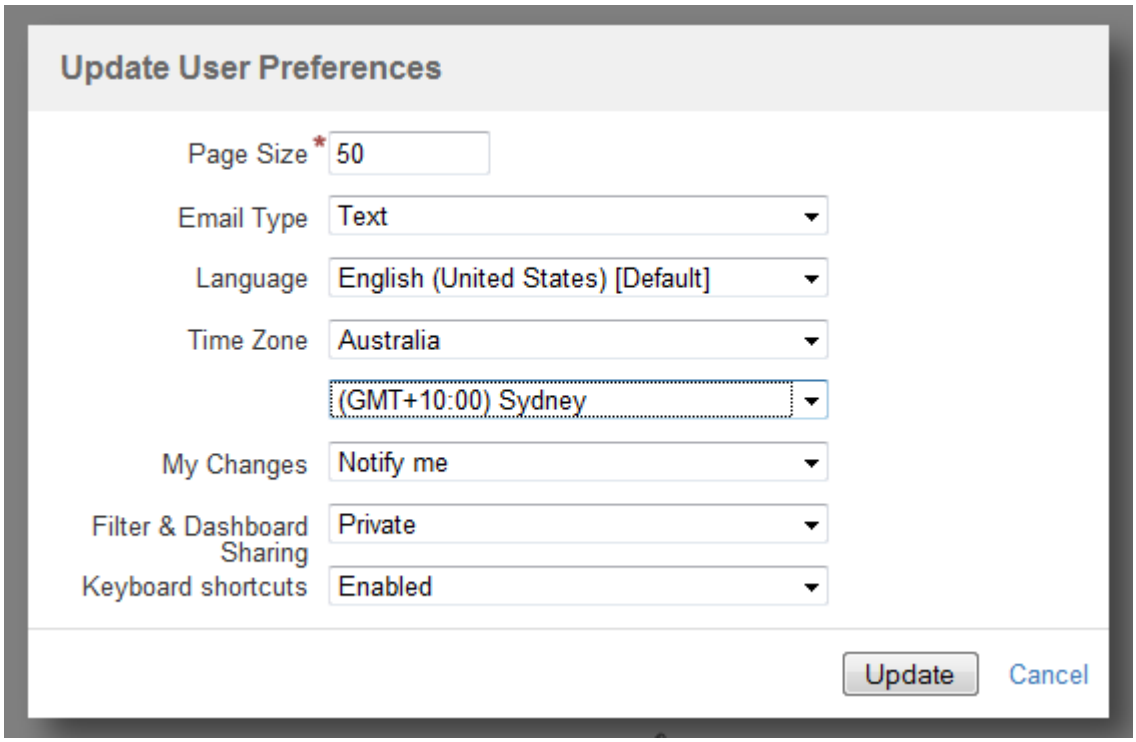
You can set a default user time zone at an administration level, and individual users have the ability to choose their own time zone. Each user's time zone is displayed in their hover profile.

Time zone support has been implemented for quick searching, simple searching and advanced searching, chart and report gadgets, date/time-based custom fields, as well as issue histories, work logs and source code check-ins (via the JIRA FishEye Plugin).

Note: Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information).

JIRA administrators can change the default time zone by going to '**Administration**' => '**General Configuration**' (under '**Global Settings**'), and editing the '**Default user time zone**'.

Users can also change their individual time zone setting via their user profile:



Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

^Top

Editable Options for Custom Fields

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

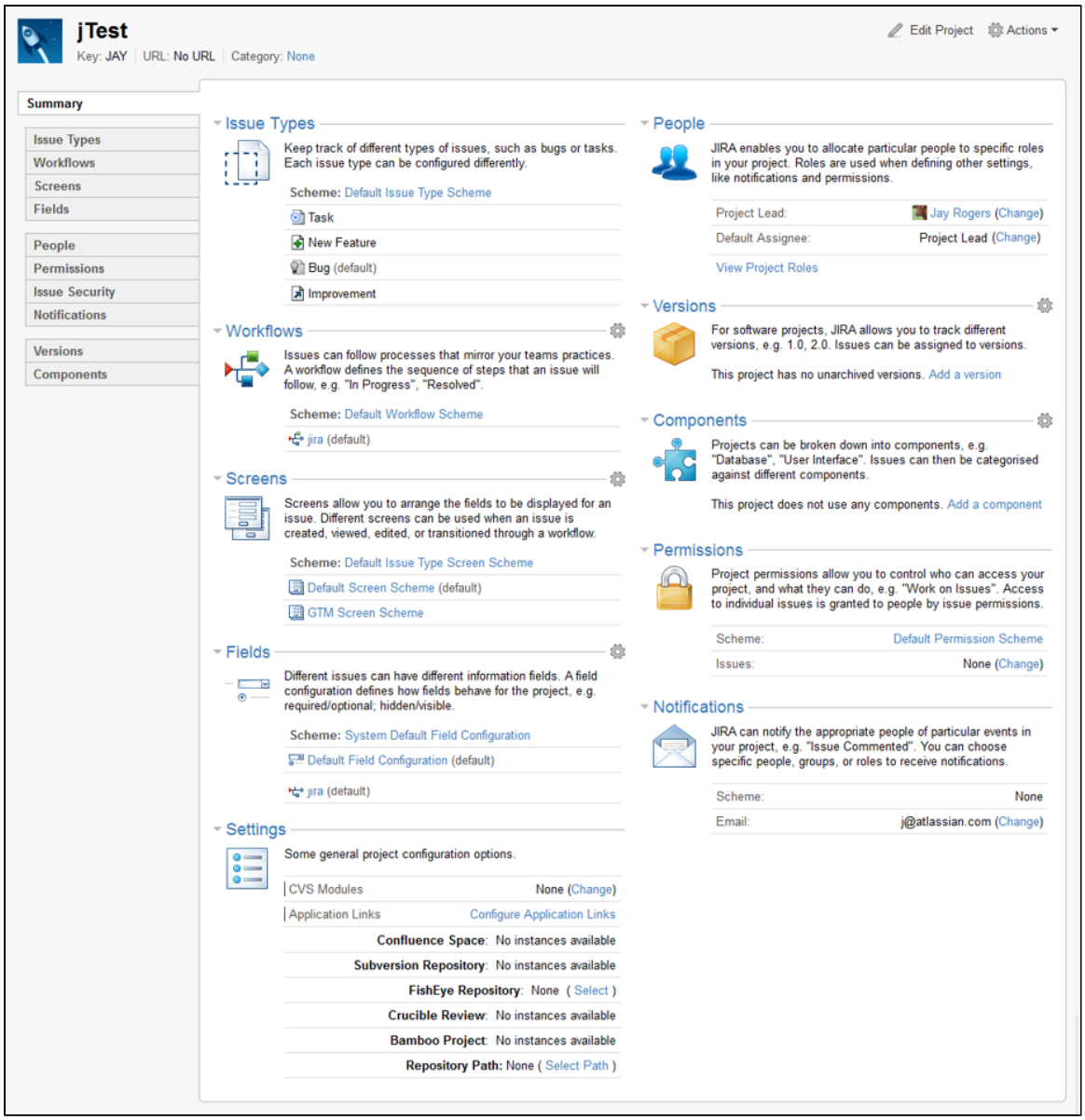
- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

^Top

Simplified Project Administration *(new since EAP 6)*

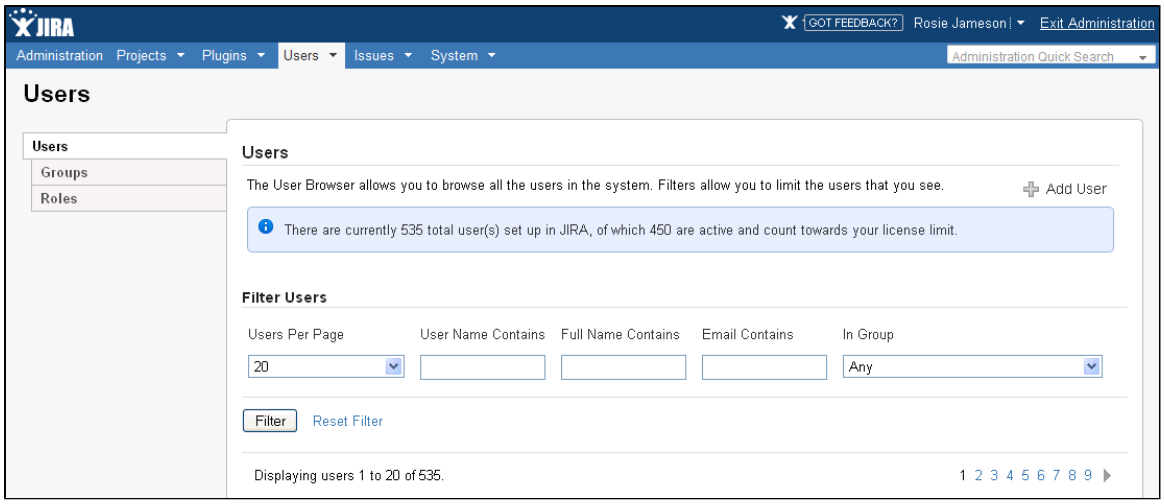
Project administration has become more visual and far simpler:



^Top

New-Look Administration Area *(improved since EAP 6)*

JIRA 4.4 brings you a dedicated 'administration mode', which replaces the left-hand column of the JIRA's administration console with a series of drop-down menus across the top navigation bar:



To find the new location of a menu item, type it into the **"Administration Quick Search"** box at the top right of the screen — or click the drop-down in the **"Administration Quick Search"** box to get a full list of admin options.

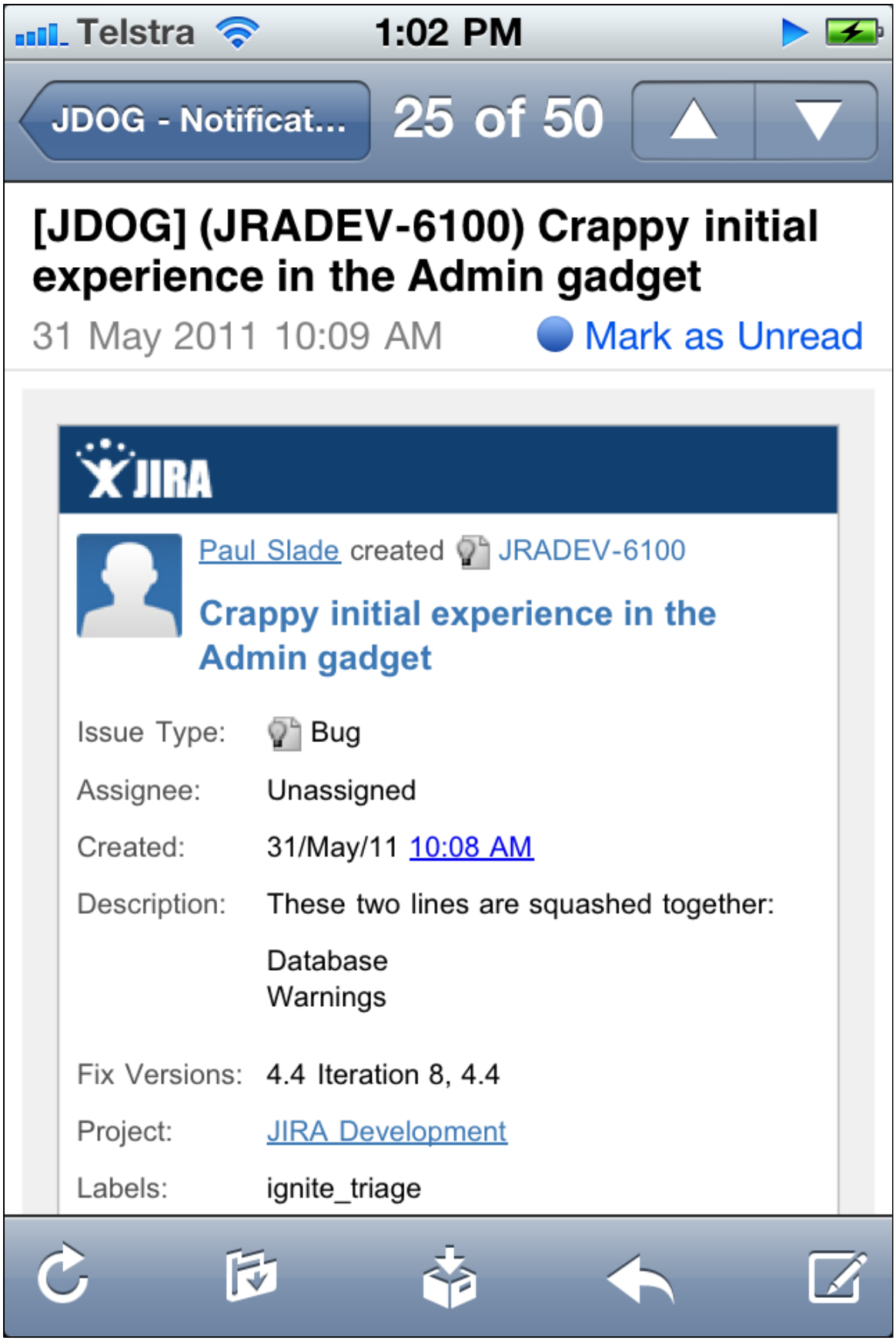
You can bring up the **"Administration Quick Search"** box from anywhere in JIRA by typing **G + A** (note that this will change to **G + G** in the final 4.4 release).

To leave JIRA's 'administration mode', click the '**Exit Administration**' link at the top-right of the screen to return JIRA to its standard user mode.

^Top

Classy New Email Templates *(new since EAP 6)*

The HTML email templates have undergone a complete visual refresh. They will also render nicely in mobile mail clients:



They will also thread better in mail clients (such as Gmail), so all emails relating to one issue will thread together.

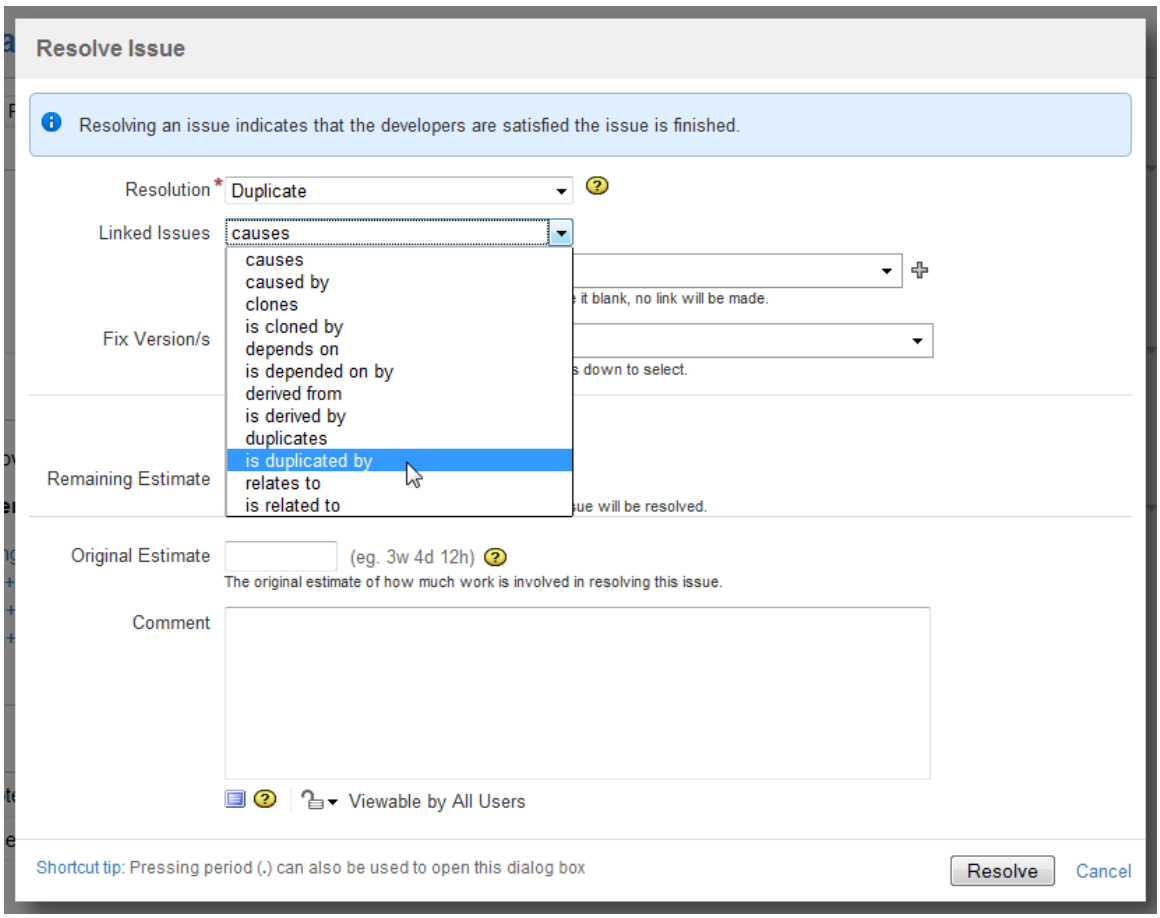
^Top

Issue Linking when Resolving an Issue

When resolving an issue, you can create links to other issues on an issue resolution screen. This is handy when you want to resolve an issue as a 'duplicate' of another *and at the same time* link to the duplicate issue.

For convenience, your last 10 issue links and resolutions are readily accessible from the 'Linked Issues' fields (in the screenshot below). The 'Linked Issues' fields can also be added to any JIRA screen via the new 'Issue Linking' field in JIRA 4.4. See Defining a Screen for more information.

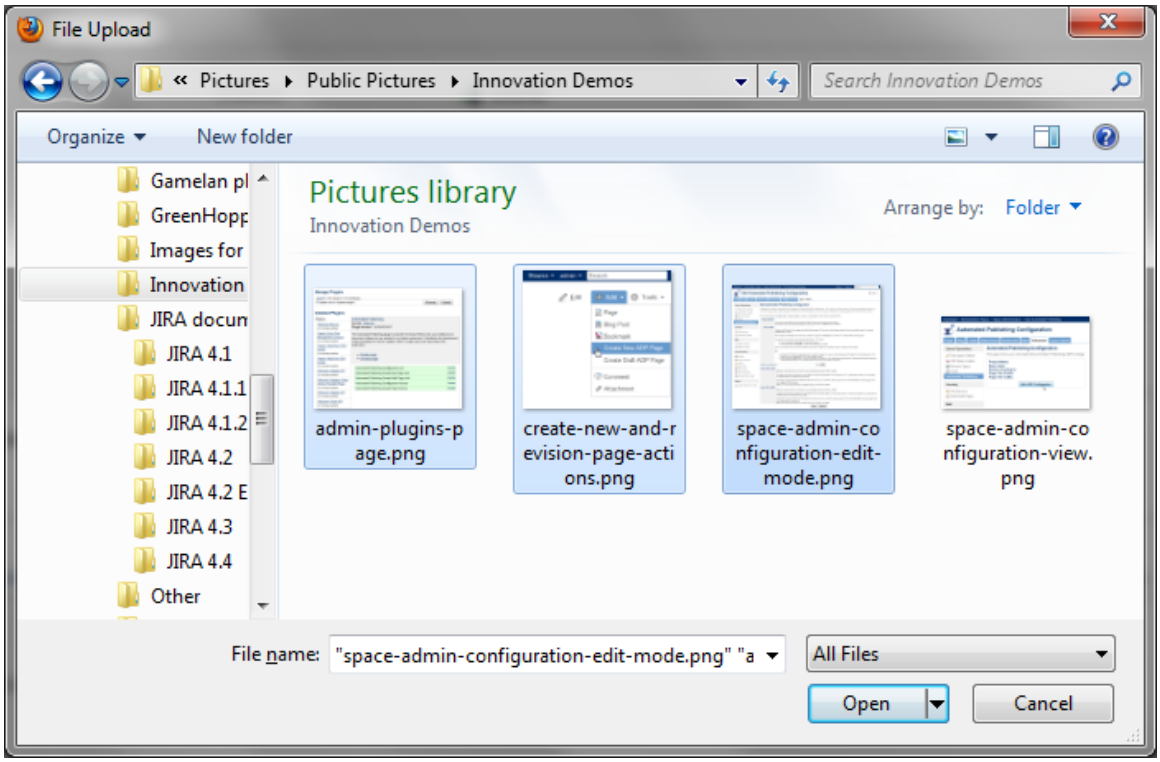
If you are upgrading from an earlier version of JIRA, you need to configure this feature manually through JIRA's administration area. Please refer to the Upgrade Notes for details.



^Top

Multiple File Selection and Upload from the 'File Upload' Dialog Box

When using JIRA's 'Attach Files' dialog box, you can now select multiple files in the 'File Upload' dialog box that appears after clicking the 'Browse' button.



^Top

Enhancements to JQL "WAS" Function *(new since EAP 6)*

The ability to search an issue's Change History was introduced in JIRA 4.3 and allowed you to search the history of the **Status** field.

In JIRA 4.4 you can now also search the history of:

- the **Assignee** field, e.g.:

```
assignee WAS "jsmith"
```

- the **Reporter** field, e.g.:

```
reporter WAS "djones"
```

You can also now search for a field that had a particular value:

- **ON** a given date — e.g. find issues that had a status of "closed" on May 31st:

```
status WAS "closed" ON "2011/05/31"
```

- **BEFORE** a given date — e.g. find issues that were assigned to jsmith before May 31st:

```
assignee WAS "jsmith" BEFORE "2011/05/31"
```

- **AFTER** a given date — e.g. find issues that were assigned to me after May 31st:

```
assignee WAS currentUser() AFTER "2011/05/31"
```

- **DURING** a given date range — e.g. find issues that were assigned to me during May:

```
assignee WAS currentUser() DURING ("2011/05/01","2011/05/31")
```

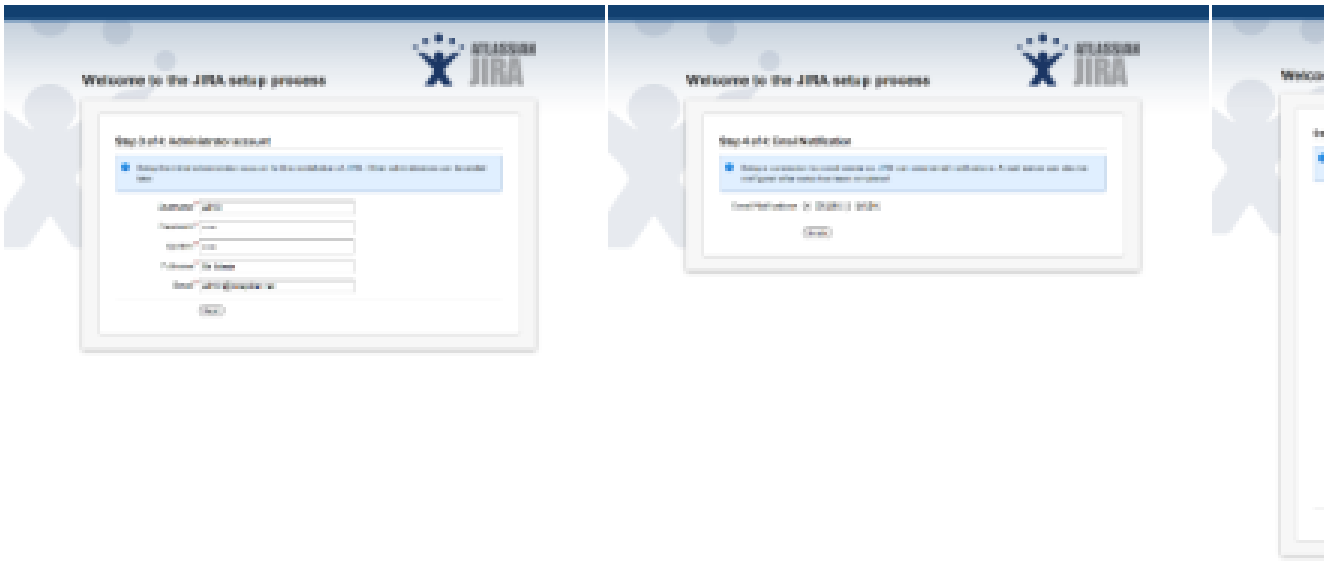
- set **BY** a particular user — e.g. find issues that were assigned to Fred by me:

```
assignee WAS "Fred" BY currentUser()
```

Improved Setup Wizard

Our trusty wizard has had a makeover:

The new Setup Wizard



^Top

12

Database Configuration Now Part of the Setup Wizard

In JIRA 4.4, configuring a connection to an external database is now part of the standard setup wizard. Upon completing the setup wizard, JIRA will create a direct JDBC connection (whose entire configuration is stored within your JIRA home directory).

The database configuration step of the setup wizard will change before the final 4.4 release, but here is how it looks so far:

^Top

Improved JIRA Standalone Linux Installer with Uninstall and Automated Upgrade *(improved since EAP 6)*

Console Installer

A simple console (command line) wizard is now available for Linux operating systems. The console wizard:

- Installs JIRA under a dedicated user account 'jira' with restricted write access to your JIRA installation directory.
- Can install JIRA as a service, so that JIRA automatically re-starts whenever your Linux operating system must be rebooted.

To install JIRA, simply download the Linux `.bin` installer file and at a shell prompt, execute this file and follow the remaining prompts!

The console wizard can install JIRA as either the 'root' user or a non-root user. However, to install JIRA as a service, the console wizard must be executed as the 'root' user.

See Installing JIRA on Linux for details.

Unattended Installation

After installing JIRA on Linux using the `.bin` installer file above, a configuration file called `response.varfile` is generated in the `.install4j` subdirectory of your JIRA Installation Directory.

See Performing an Unattended Installation (on Linux) for details.

Uninstaller

After installing JIRA on Linux using the `.bin` installer file above, an executable file called `uninstall` (located in your JIRA Installation Directory) is available to conveniently uninstall JIRA.

See Uninstalling JIRA from Linux for details.

Automated Upgrade *(improved since EAP 6)*

The console wizard includes an option that allows you to upgrade an existing JIRA Standalone installation from version 4.3.x or later.

While the upgrade feature installs a new version of JIRA, it automates the following tasks for you:

1. Backs up your existing JIRA installation and home directories.
2. Migrates database configurations used in JIRA 4.3.x and earlier to the new database configuration used in JIRA 4.4.
3. Migrates port values in your existing JIRA installation's `server.xml` file to your new version of JIRA.
4. Migrates custom values in your existing JIRA installation's `jira-application.properties` and `setenv.sh` files to your new version of JIRA.
   In the `setenv.sh` file, only the following values are migrated:
   - `JVM_SUPPORT_RECOMMENDED_ARGS`
   - `JVM_MINIMUM_MEMORY`
   - `JVM_MAXIMUM_MEMORY`
   - `JIRA_MAX_PERM_SIZE`

(New in Beta 1) The upgrade feature detects and notifies you of any files (other than `jira-application.properties` and `setenv.sh`) in the `atlassian-jira` subdirectory of your existing JIRA Installation Directory, which had been deleted, added or modified from a 'default' JIRA installation. This informs you of any customisations you will need to migrate manually over to your upgraded JIRA installation directory.

Please Note:

- The upgrade process requests that you conduct a backup of your database using your database's backup utilities. If your database does not support online backups, you can stop the upgrade process, shut down JIRA, perform your database backup and then restart the upgrade process to continue on.
- If you have made customisations to your `seraph-config.xml` file or any other files in your JIRA installation directory, these must be migrated manually.
- If your attachments and index files are located outside your JIRA home directory, then backups of these directories must be performed manually.

^Top

Improved JIRA Standalone Windows Installer with Automated Upgrade *(improved since EAP 6)*

The installation wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

See Installing JIRA on Windows for details.

Unattended Installation

When you execute the installer file at the Windows command prompt and complete your installation of JIRA, a file called `response.varfile` is generated in the `.install4j` subdirectory of your JIRA Installation Directory.

See Performing an Unattended Installation (on Windows) for details.

Automated Upgrade *(improved since EAP 6)*

The installation wizard includes an option that allows you to upgrade an existing JIRA Standalone installation from version 4.3.x or later.

While the upgrade feature installs a new version of JIRA, it automates the following tasks for you:

1. Backs up your existing JIRA installation and home directories.
2. Migrates database configurations used in JIRA 4.3.x and earlier to the new database configuration used in JIRA 4.4.
3. Migrates port values in your existing JIRA installation's `server.xml` file to your new version of JIRA.
4. Migrates custom values in your existing JIRA installation's `jira-application.properties` and `setenv.bat` files to your new version of JIRA.
   In the `setenv.bat` file, only the following values are migrated:
   - `JVM_SUPPORT_RECOMMENDED_ARGS`
   - `JVM_MINIMUM_MEMORY`
   - `JVM_MAXIMUM_MEMORY`
   - `JIRA_MAX_PERM_SIZE`

(New in Beta 1) The upgrade feature detects and notifies you of any files (other than `jira-application.properties` and `setenv.sh`) in the `atlassian-jira` subdirectory of your existing JIRA Installation Directory, which had been deleted, added or modified from a 'default' JIRA installation. This informs you of any customisations you will need to migrate manually over to your upgraded JIRA installation directory.

Please Note:

- The upgrade process requests that you conduct a backup of your database using your database's backup utilities. If your database does not support online backups, you can stop the upgrade process, shut down JIRA, perform your database backup and then restart the upgrade process to continue on.
- If you have made customisations to your `seraph-config.xml` file or any other files in your JIRA installation directory, these must be migrated manually.
- If your attachments and index files are located outside your JIRA home directory, then backups of these directories must be performed manually.

^Top

REST API improvements

There are new REST APIs for

- Listing and managing Project Components.
- Listing and managing Project Versions.
- Listing and managing Project Roles.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.3 RC1 Release Notes

18 February 2011

JIRA 4.3 RC1 (a.k.a 4.3 milestone 7 or 'm7') is a public development release leading up to **JIRA 4.3**. A Release Candidate (RC) is a preliminary release leading up to the official release of a JIRA version. RC releases are a fairly stable snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use RC releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 4.3 RC1 release. As always, we would love to hear your feedback and comments about this release. To keep the discussion centralised, please provide your feedback as a comment on JRA-23131.

Identity management comes of age in JIRA 4.3, with complete LDAP integration. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins, and a new data importer for easier migration from your old systems. Additionally, a raft of new JQL functions give you many powerful new searching options, such as the ability to search an issue's change history.

Dashboard Publish/Subscribe with Confluence (i.e. Gadgets 2.0) is also included, so if you are using Confluence as well as JIRA, your Confluence gadgets will now appear in the JIRA Gadget Directory.

Note to developers: JIRA 4.3 RC1 includes Unified Application Links (UAL), Gadgets 2.0, and the Atlassian Plugin Framework version 2.6. Also please see the Plugin Developer Notes for JIRA 4.3, plus note that the REST API continues to evolve — please watch the Developer blog for the latest news on the REST API.

Highlights of JIRA 4.3 RC1:

- Full integration with LDAP and Active Directory
- Easy management of User Directories via JIRA Admin UI
- New Plugin Management System
- Application Links: Connecting Applications Together
- Add another application's Gadgets to your JIRA Gadget Directory
- Improved Importer ***(new since Beta 1)***
- Dashboard Performance Improvements
- JQL and Quick Search Enhancements
- Revamped User Avatars
- Improvements to Issue Links
- Remembered Assignees
- 'Whitelist' for External URLs ***(new since Beta 1)***
- Mail Server Configuration Improvements
- Security Enhancements
- JIRA now supports 'In-place Database Upgrades'
- Support Tools Plugin now bundled ***(new since Beta 1)***
- Support for Chrome and Safari 5 Browsers
- REST API Improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.3 RC1

Download RC

Upgrading to JIRA 4.3 RC1

JIRA RC releases are available here. When upgrading, please follow the JIRA 4.3 Upgrade Guide.

Do not use in production

RC releases should not be used in production environments as they are not officially supported.

Please also note:

- **RC releases may not be safe** — RC releases are snapshots of the ongoing JIRA development process. As such:
  - While we deem these releases to be fairly stable, they have not undergone sufficient testing to be deemed an official (final) release.
  - While feature development work is complete, some features may be subject to change or may be removed before the next full release.
- **No upgrade path** — Because RC releases still represent work in progress, we **cannot** provide a supported upgrade path between RC releases, from EAP or Beta to RC releases, or from any RC to the eventual final release. Thus, any data you store in a JIRA RC release may not be able to be migrated to a future JIRA release.

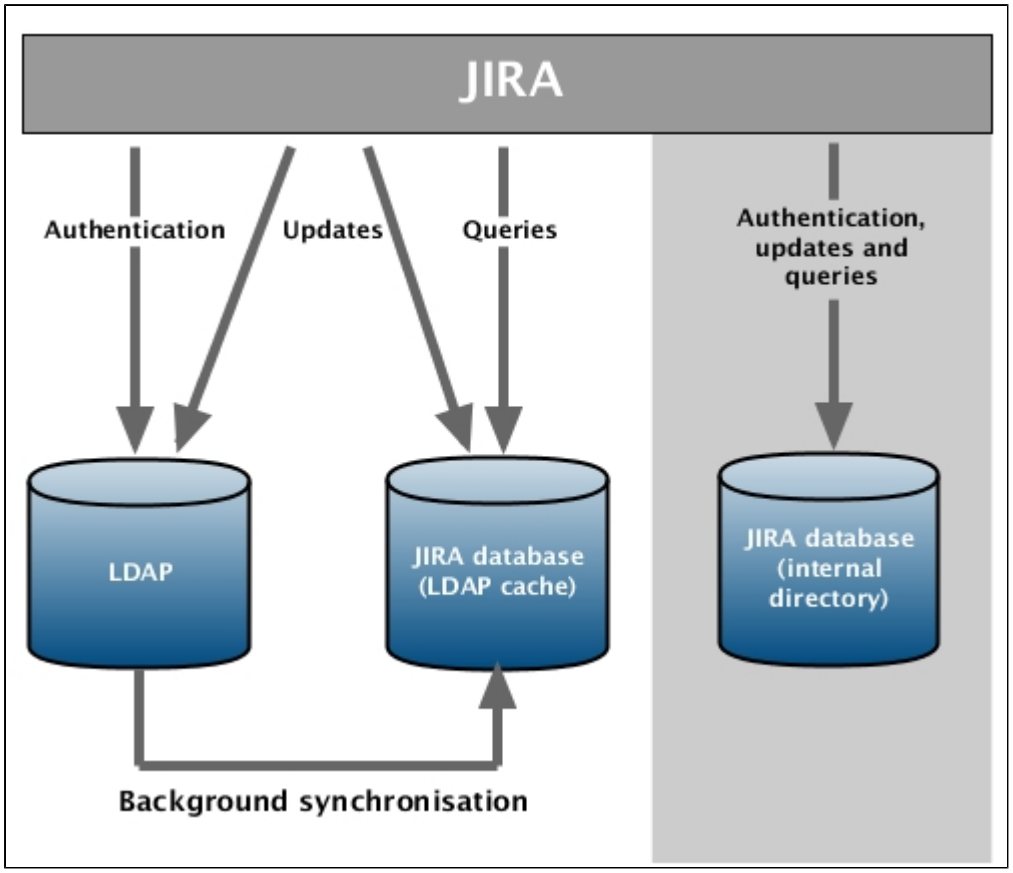
Highlights of JIRA 4.3 RC1

Full integration with LDAP and Active Directory

The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3, giving you the ability to connect to an LDAP server — including Microsoft Active Directory — for *all* user information. Your options include:

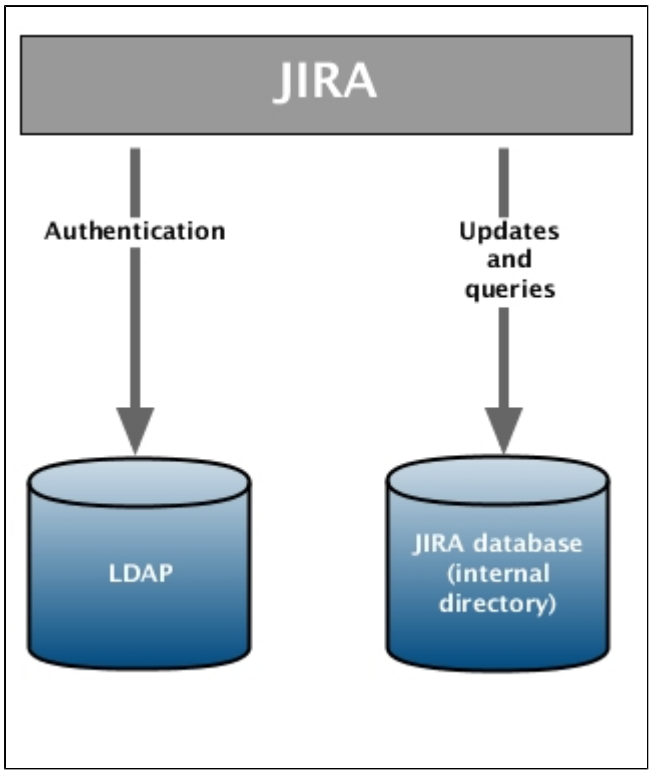
Integrate JIRA with LDAP/Active Directory

Update your user details in either JIRA *or* LDAP/Active Directory and they will automatically populate to the other repository:



Use LDAP for authentication only

This was the old way of integrating JIRA with LDAP, prior to JIRA 4.3. You can still do this, but much more easily (see below).



Connect to a Crowd server for user management

For larger and more complex installations, you may need to install Atlassian Crowd for user management and single sign-on. (See our guide to limitations and recommendations.) When integrating earlier versions of JIRA and Crowd, you had to manually edit a number of configuration files. JIRA 4.3 offers the following new features:

- Simple and quick setup via the JIRA and Crowd administration consoles.
- Clever synchronisation and caching to ensure the best response times.
- More...

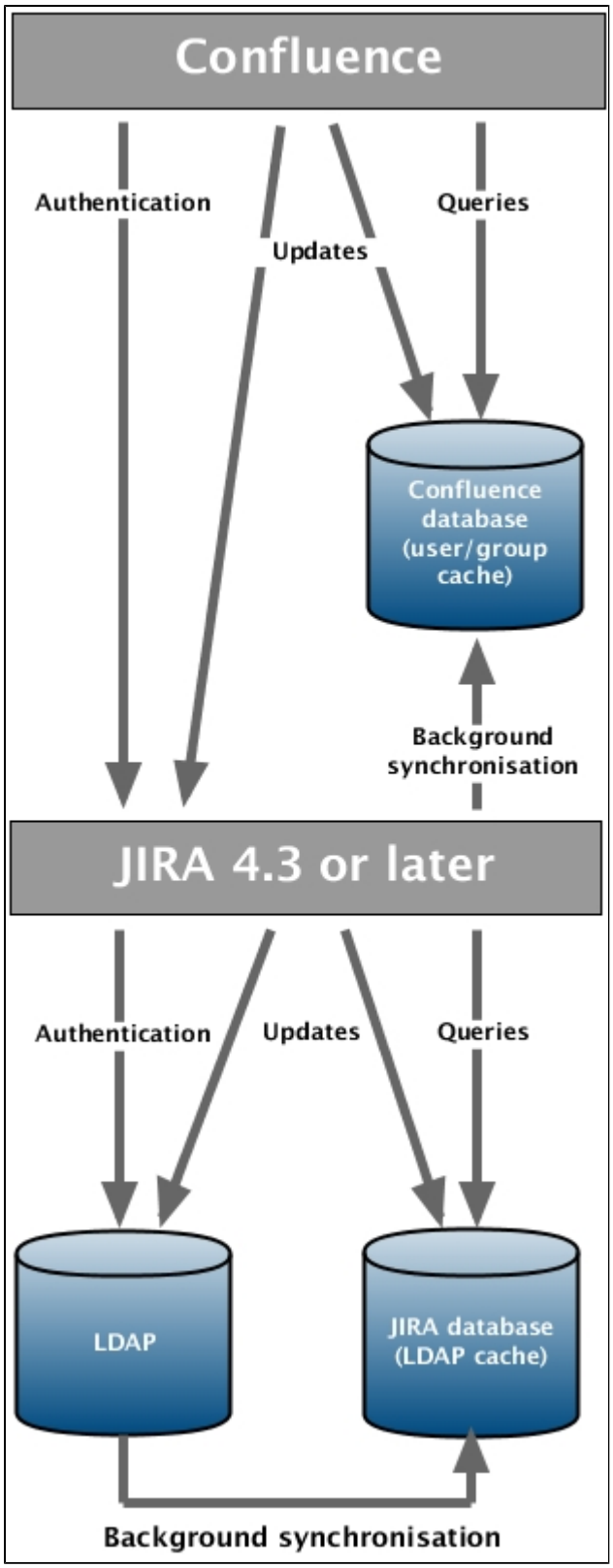
Use multiple LDAP and/or Crowd servers simultaneously

If you have multiple directories, you can now simply connect JIRA to all of them.

Connect Confluence to your JIRA User Directory

JIRA 4.3 can act as the directory manager for your Confluence site, interacting with one or more user directories and ensuring that you have the same set of users and groups across both applications.

Confluence 3.5 can connect directly to JIRA 4.3 via the administration UI. Clever synchronisation and caching ensure the best response times for your directory searches.



Earlier versions of Confluence can continue to use a direct JDBC connection to JIRA 4.2 and earlier.

^Top

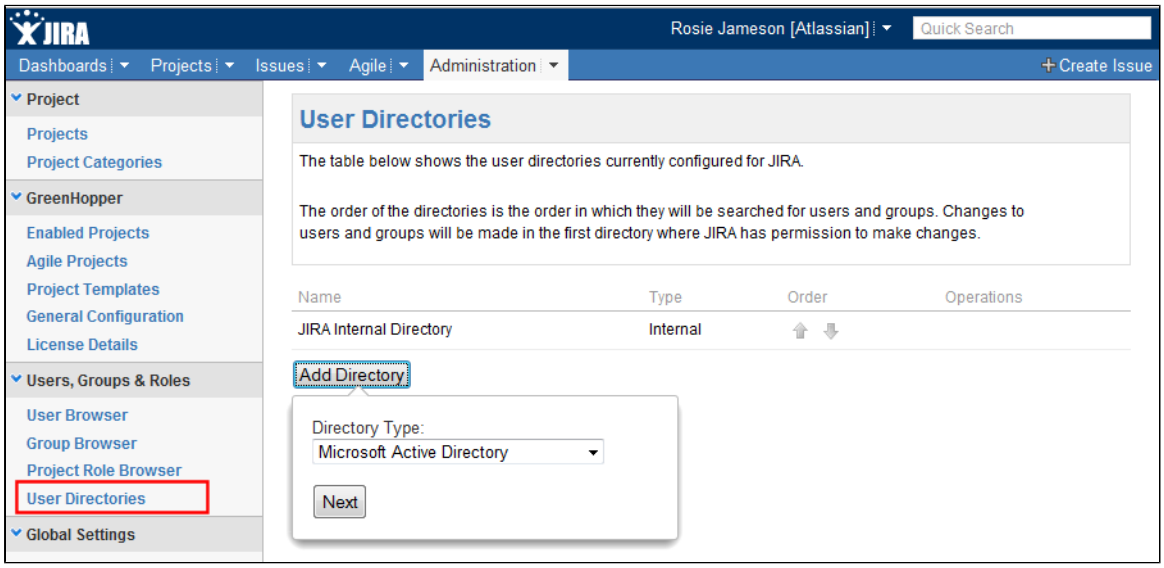
2

Easy management of User Directories via JIRA Admin UI

In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.

JIRA 4.3 brings a simple, powerful and flexible directory management interface:

- Choose from a list of supported directory types, including Microsoft Active Directory and ten other popular LDAP schemas.
- Configure all your LDAP settings via the JIRA Administration interface: permissions, server and schema settings. We pre-populate the fields with default values depending on your choice of directory type.
- Choose the LDAP permissions to suit your needs: read/write, read only, local groups or authentication only.
- Make use of the caching and copy-as-required configurations to optimise the performance of your LDAP searches.
- Add as many directory servers as you need. Connect directory servers of different types, including the JIRA internal directory (default), LDAP, Crowd and/or another JIRA server.
- Turn on support for nested groups.
- Configure your LDAP connection pool.

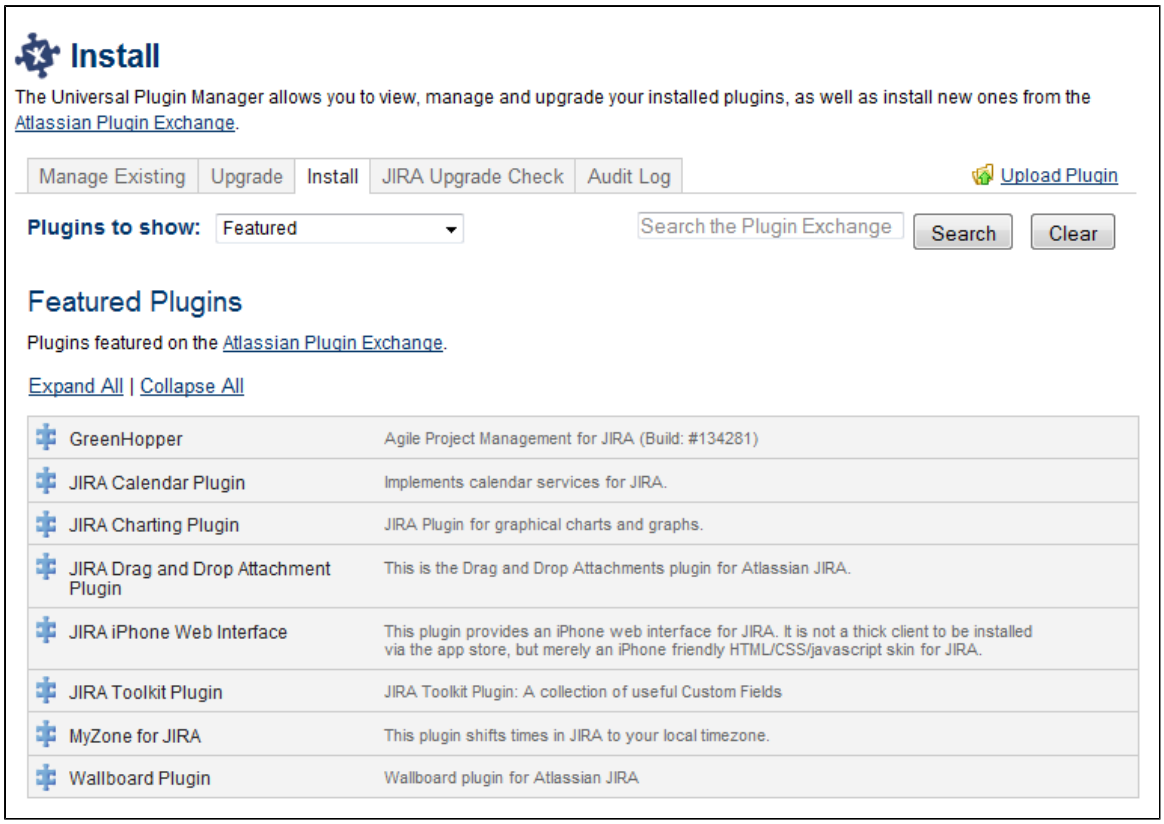


^Top

New Plugin Management System

JIRA 4.3 includes the Universal Plugin Manager (UPM), which provides you with a simple way of adding and managing plugins:

- Auto-discovery of available plugins (see **Featured Plugins** in the screenshot below)
- Point-and-click installation — you no longer need to download JAR files and shut down JIRA to install them.



^Top

Application Links: Connecting Applications Together

JIRA 4.3 includes Application Links, which makes it easy to connect your Atlassian applications together.

Application Links allows you to link your JIRA, Confluence, FishEye, Crucible, Bamboo and Subversion applications. You can even choose to associate individual entities (i.e. JIRA projects, Confluence spaces, FishEye repositories, FishEye projects, Crucible projects, Bamboo projects) with each other. Applications Links is bundled with FishEye 2.4, Confluence 3.5, JIRA 4.3, and all later versions of those applications. In addition, Bamboo 3.1 is compatible with AppLinks. You can configure JIRA-to-Bamboo links via the JIRA administration screens.

Linking two applications allows you to share information and access one application's functions from within the other. For example, if you linked a Confluence server to a JIRA server, you can create, find and insert JIRA issues directly onto a Confluence page or blog post using the new 'Insert JIRA Issue' option in Confluence 3.5.

You also can associate entities of two linked applications. For example, you could associate a JIRA project with a Confluence space. This allows you to take advantage of additional integration features like link rendering, which lets you create links to issues or pages in the project or space using a simple textual reference, e.g. `[JRA-1234], [myConfluenceSpace:Test Page]`.

- In JIRA 4.3 there is now a '**Configure Application Links**' option on the '**Project**' management screen, enabling you to easily configure application links for a given project.

For more about configuring Application Links, please see the Application Links Administrator's Guide.

^Top

Add another application's Gadgets to your JIRA Gadget Directory

JIRA 4.3 includes Atlassian Gadgets 2.0, allowing you to quickly add all gadgets from your Confluence, Bamboo, FishEye or Crucible instance — or from another JIRA instance — to your JIRA Gadget Directory, for easy addition to your JIRA dashboard:

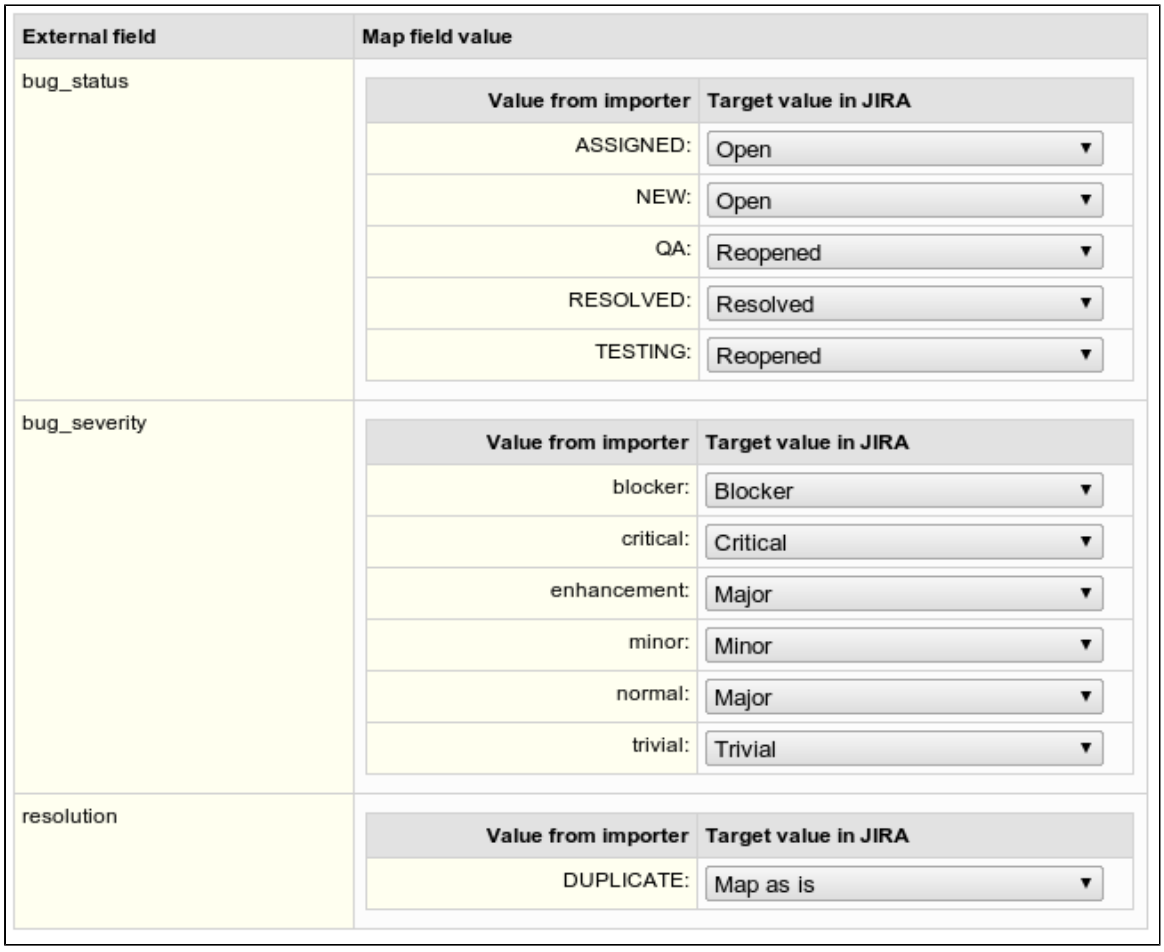
In the JIRA Gadget Directory , you can now click 'Gadget Subscriptions'. There you can provide the URL for the other application (or other JIRA instance), and all the gadgets from that instance will be added to your JIRA Gadget Directory.

^Top

Improved Importer *(new since Beta 1)*

JIRA 4.3 provides a greatly improved Importer for Bugzilla, Mantis, FogBugz and CSV files, by bundling the JIRA Importers Plugin.

The web-based import wizard makes it easy to map fields — and individual field values — from your old bug-tracker to your new JIRA system:



^Top

Dashboard Performance Improvements

We are extremely pleased to announce that you should see a noticeable reduction in the time it takes to load a Dashboard.

- On average, dashboard performance has improved by 29%
- For large dashboards, the performance improvement is 35%

We are also extremely pleased to announce that you should see a noticeable reduction in the time it takes to load the Gadget Directory (depending on how many external gadgets you have configured).

^Top

JQL and Quick Search Enhancements

JIRA 4.3 incorporates several JQL and Quick Search enhancements. You can now:

Search the Change History with JQL 'WAS' Operator

JIRA 4.3 introduces the ability to search the Change History of issues. There is much more functionality to follow, but in this release you can search for changes to the Status field.

For example, the following will return all issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

The `WAS` operator can be used with the `NOT`, `IN` and `NOT IN` operators, e.g. to find issues that have never had a status of 'In Progress' or 'Resolved':

```
status WAS NOT IN ("In Progress","Resolved")
```

Search for relative dates and versions with JQL 'startOfDay', 'endOfDay', 'earliestUnreleasedVersion', 'latestReleasedVersion' (new since Beta 1)

JIRA 4.3 introduces the ability to search for issues relative to the current day, month, week or year. For example, to find issues that have been created today:

```
created > startOfDay()
```

Or to find issues that are due by the end of this month:

```
due < endOfMonth()
```

You can also perform searches based on the earliest unreleased version (i.e. the next version that is due to be released) of a specified project:

```
earliestUnreleasedVersion(project)
```

Or on the most recently released version of a specified project:

```
latestReleasedVersion(project)
```

Use a wild card when searching the 'Fix Version' field with Quick Search

When using Quick Search, you can now use the wildcard symbol: "*" to find issues that matches a core part of a Fix Version. For example, "`ff:3.2*`" will match any issue whose Fix For Version is:

- 3.2
- 3.2-beta
- 3.2.1
- 3.2.x

Use the `r:` prefix with Quick Search to find issues reported by a specific user

With Quick Search, you can find issues reported by you, another user or with no reporter, using the prefix "`r:`" followed by a specific reporter term such as "`me`", a username or "`none`", such that:

- "`r:me`" — finds issues reported by you.
- "`r:samuel`" — finds issues reported by the user whose username is "samuel".
- "`r:none`" — finds issues with no reporter.

^Top

Revamped User Avatars

JIRA 4.3 introduces the new-look Atlassian avatars:

- User avatars are displayed as the icon for your profile, and to illustrate your comments on an issue. See the documentation on Adding a User Avatar.

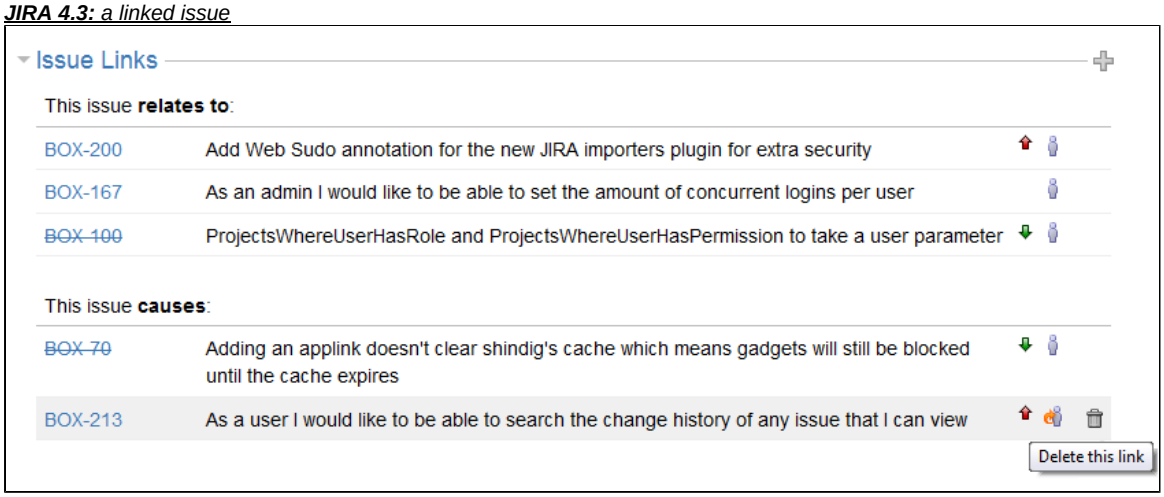
We were fond of the old avatars, but think you'll agree they were looking a little dated by comparison:

^Top

Improvements to Issue Links

The 'Linked Issues' section of the 'View Issue' screen has been streamlined to make it both more compact and quicker to use. You can now delete links directly from this screen upon mouse hover:

JIRA 4.3: *a linked issue*



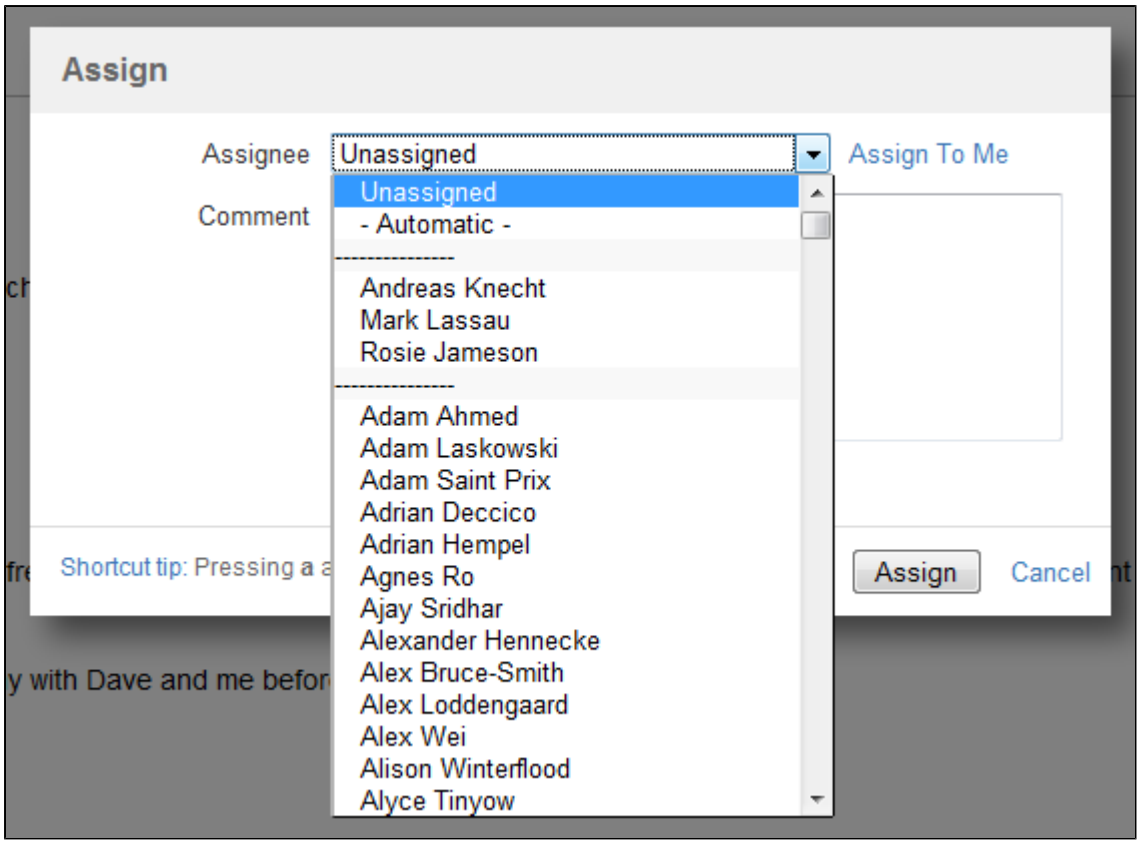
Previously, to delete a link you needed to click the 'cog' icon (which took you to a separate screen):

JIRA 4.2: *a linked issue*

^Top

Remembered Assignees

JIRA now makes it easier to assign users to issues. The top of the assignee dropdown list shows the last five people you assigned issues to, as well as the reporter of the issue and all participants involved in the issue.

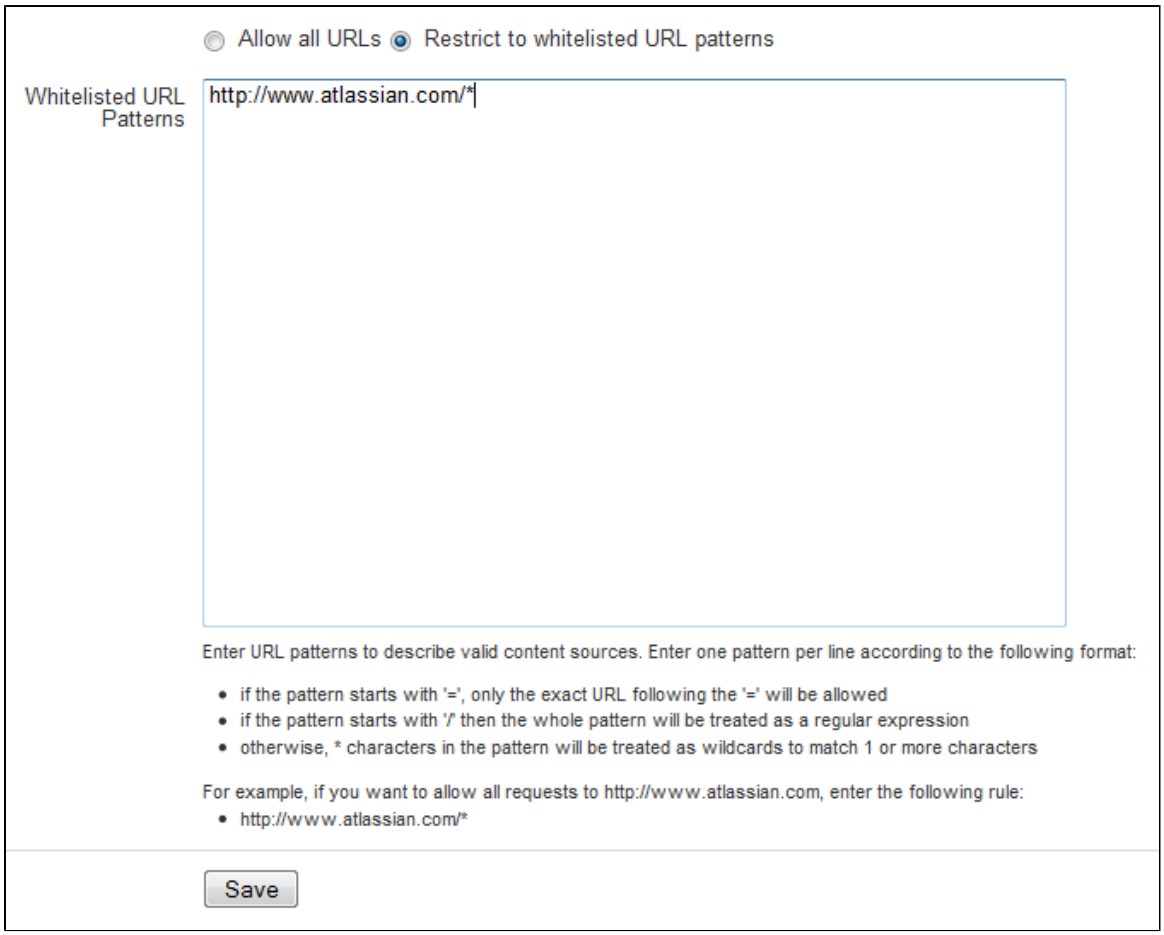


^Top

'Whitelist' for External URLs *(new since Beta 1)*

For security reasons, an administrator may wish to limit the URLs from which users can source content that is displayed on their JIRA site (e.g. via an External Gadget).

JIRA 4.3 allows you to create a 'Whitelist' of URLs — or URL patterns — whose content you wish to make available to users of your JIRA site:

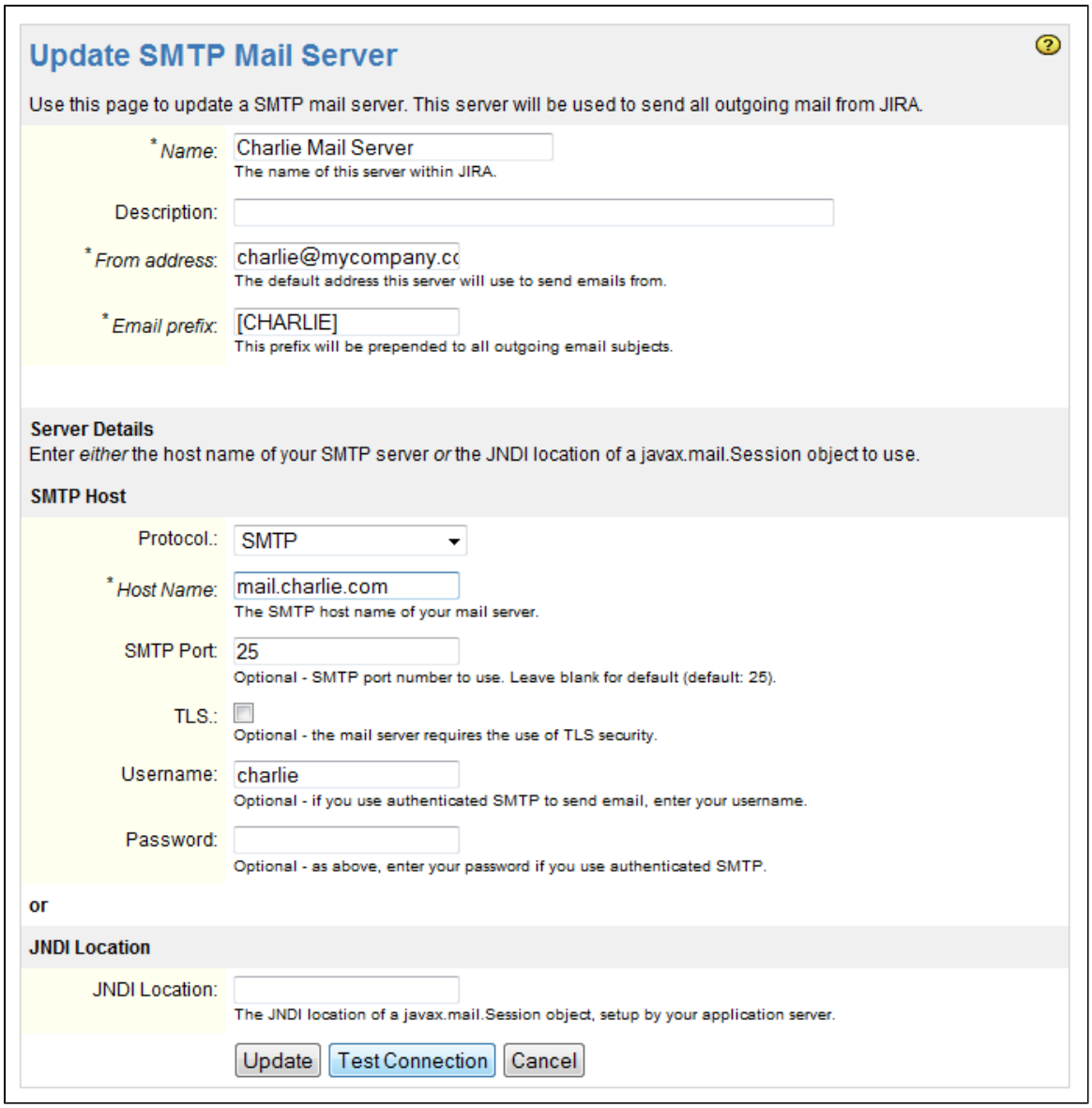


See the documentation for more details.

^Top

Mail Server Configuration Improvements

In JIRA 4.3, administrators can now test their mail server configuration with the "Test Connection" button.



A '**Timeout**' field has also been added.

^Top

Security Enhancements

For enhanced security:

User Email Change is now Password Protected

Users will now be prompted to enter their password when changing their email address.

JIRA provides Secure Administration Sessions (new since Beta 1)

Access to JIRA's administration features and functions is now password-protected.

See the documentation for more details.

^Top

JIRA now supports 'In-place Database Upgrades'

JIRA 4.3 now officially supports 'in-place database upgrades', when upgrading from JIRA 4.0.0 or later.

This method requires much less downtime during the JIRA upgrade process, especially if you operate a large JIRA installation. You no longer need to export your existing JIRA data to an XML backup and then restore this data into your new JIRA version. Instead, we now support simply 'pointing' your new version of JIRA at your existing JIRA database!

See the documentation for more details.

^Top

Support Tools Plugin now bundled *(new since Beta 1)*

We are very pleased to announce that the Atlassian Support Tools Plugin is now bundled with JIRA. The Atlassian Support Tools Plugin proves tools to help you troubleshoot issues with Atlassian products and get help from Atlassian Support.

See the documentation for more details.

^Top

Support for Chrome and Safari 5 Browsers

We are very pleased to announce that Chrome and Safari 5 are supported for use with JIRA 4.3.

^Top

REST API Improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance. Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3 so far, click here.

^Top

JIRA 4.3 Beta 1 Release Notes

1 February 2011

JIRA 4.3 Beta 1 (a.k.a 4.3 milestone 6 or 'm6') is a public development release leading up to **JIRA 4.3**. A Beta release is a preliminary release leading up to the official release of a JIRA version. Beta releases are a snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use Beta releases to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 4.3 Beta 1 release. Thank you for your feedback during the recent EAP releases, and please keep providing it here.

Identity management comes of age in JIRA 4.3, with complete LDAP integration. Additionally, you now have the ability to search an issue's change history. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins.

Dashboard Publish/Subscribe with Confluence (i.e. Gadgets 2.0) is also included, so if you are using Confluence as well as JIRA, your Confluence gadgets will now appear in the JIRA Gadget Directory (and vice versa).

Note to developers: JIRA 4.3 Beta 1 includes Unified Application Links (UAL), Gadgets 2.0, and the Atlassian Plugin Framework version 2.6 . Also please see the Plugin Developer Notes for JIRA 4.3, plus note that the REST API will continue to evolve through the 4.3 Beta releases — please watch the Developer blog for the latest news on the REST API.

Highlights of JIRA 4.3 Beta 1:

- Full LDAP integration
- User Directory management via JIRA Admin UI
- New Plugin Management System
- AppLinks pre-installed and configurable from the JIRA Admin UI
- Add another application's Gadgets to your JIRA Gadget Directory
- Dashboard Performance Improvements *(enhanced since EAP 5)*
- Search the Change History with JQL 'WAS' Function *(improved since EAP 5)*
- Revamped User Avatars
- Enhancements to Quick Search
- Improvements to Issue Links *(new since EAP 5)*
- Remembered Assignees
- Mail Server Configuration Improvements
- User Email Change is now Password Protected
- Support for Chrome and Safari 5 Browsers
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.3 Beta 1
Download Beta

Upgrading to JIRA 4.3 Beta 1

JIRA Beta releases are available here. When upgrading, please follow the JIRA 4.3 Upgrade Guide.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.3 Beta 1

Full LDAP integration

The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3. This provides a number of additional capabilities, mainly the ability to use an LDAP server for all user information. Key features are:

- Ability to connect to an LDAP server, including to Microsoft Active Directory, for user management. This includes:
  - Read-write access
  - Read-only access
  - Read-only access with local groups
  - Use LDAP for authentication only — *Previous functionality*
- Ability to connect to a Crowd server for user management — *Previous functionality*
- Ability to manage users fully within JIRA — *Previous functionality*
- Ability to use 2 or more LDAP and/ or Crowd servers simultaneously for user management.

Note when upgrading from EAP 3

If you are upgrading to Beta 1 from EAP 3, please note that there have been changes to Crowd that require extra data in the cwd_applicatior table.

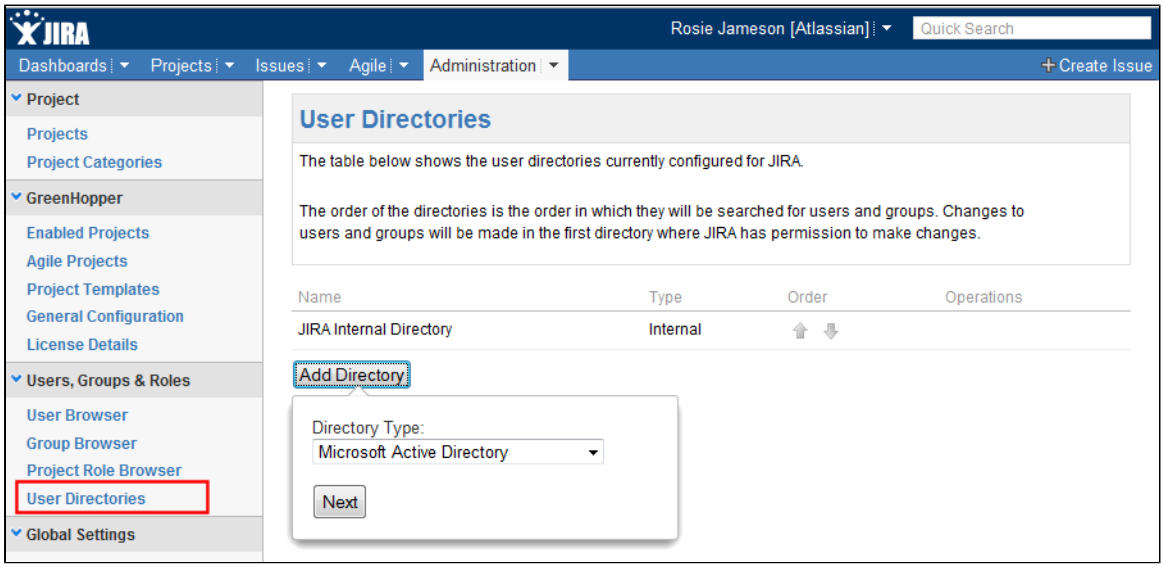
You will need to update the table and add 'CROWD' into the application_type column.

(This will not be necessary when upgrading to the final release of JIRA 4.3.)

^Top

User Directory management via JIRA Admin UI

In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.

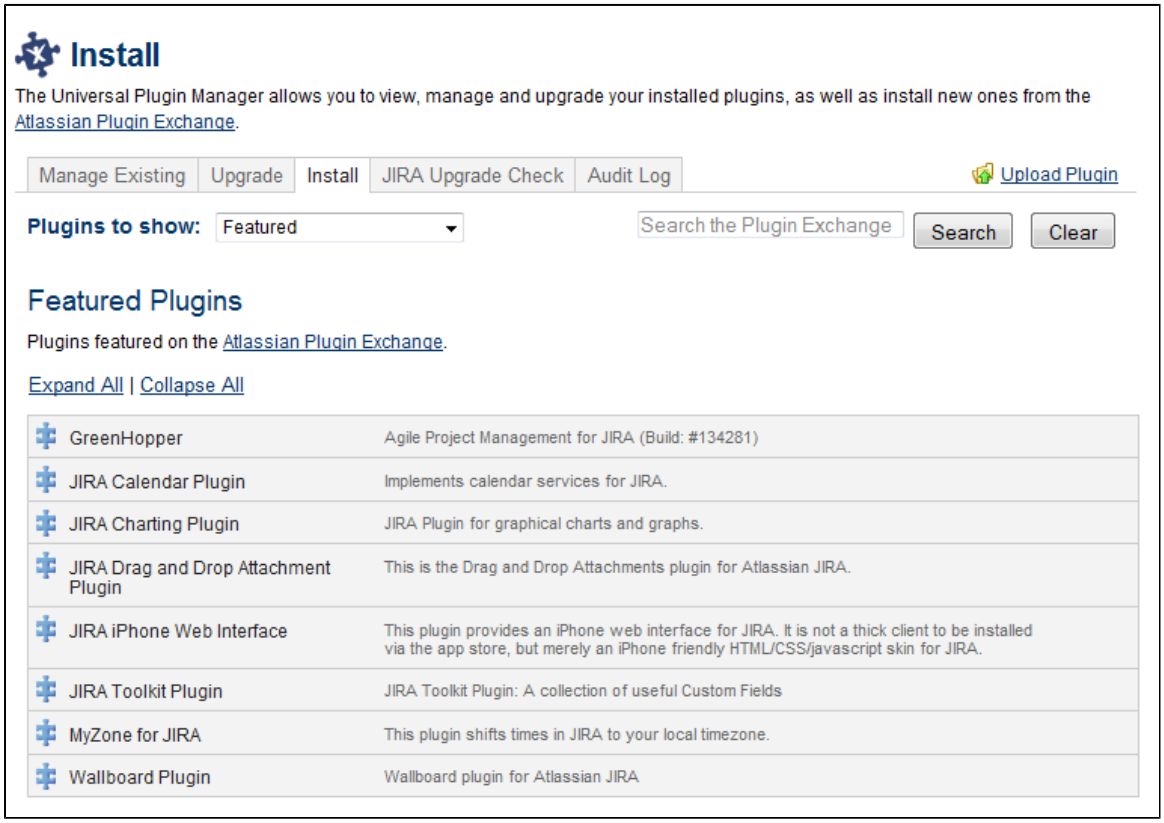


^Top

New Plugin Management System

JIRA 4.3 includes the Universal Plugin Manager (UPM), which provides you with a simple way of adding and managing plugins:

- Auto-discovery of available plugins (see **Featured Plugins** in the screenshot below)
- Point-and-click installation — no more downloading JAR files



^Top

AppLinks pre-installed and configurable from the JIRA Admin UI

JIRA 4.3 includes Unified Application Links (UAL), so you no longer need to install it separately.

Application Links (AppLinks) is a plugin that allows you to link your JIRA, Confluence, FishEye, Crucible, Bamboo and Subversion applications. You can even choose to associate individual entities (i.e. JIRA projects, Confluence spaces, FishEye repositories, FishEye projects, Crucible projects, Bamboo projects) with each other. Applications Links is bundled with FishEye 2.4, Confluence 3.5, JIRA 4.3, and all later versions of those applications. In addition, Bamboo 3.1 is compatible with AppLinks. You can configure JIRA-to-Bamboo links via the JIRA administration screens.

Linking two applications allows you to share information and access one application's functions from within the other. For example, if you linked a JIRA server and a Confluence server, you could view JIRA issues in a Confluence page via the JIRA Issues Macro.

You also can associate entities of two linked applications. For example, you could associate a JIRA project with a Confluence space. This allows you to take advantage of additional integration features like link rendering, which lets you create links to issues or pages in the project or space using a simple textual reference, e.g. `[JRA-1234], [myConfluenceSpace:Test Page]`.

- In JIRA 4.3 there is now a '**Managed Unified Application Links**' option on the '**Manage Project**' screen, enabling you to easily configure application links for a given project.

For more about configuring Application Links, please see the Application Links Administrator's Guide.

^Top

Add another application's Gadgets to your JIRA Gadget Directory

JIRA 4.3 includes Atlassian Gadgets 2.0, allowing you to quickly add all gadgets from your Confluence, Bamboo, FishEye or Crucible instance — or from another JIRA instance — to your JIRA Gadget Directory, for easy addition to your JIRA dashboard:

In the JIRA Gadget Directory , you can now click 'Gadget Subscriptions'. There you can provide the URL for the other application (or other JIRA instance), and all the gadgets from that instance will be added to your JIRA Gadget Directory.

^Top

Dashboard Performance Improvements *(enhanced since EAP 5)*

We are extremely pleased to announce that you should see a noticeable reduction in the time it takes to load a Dashboard.

- On average, dashboard performance has improved by 29%
- For large dashboards, the performance improvement is 35%

In Beta 1 we are also extremely pleased to announce that you should see a noticeable reduction in the time it takes to load the Gadget Directory (depending on how many external gadgets you have configured).

^Top

Search the Change History with JQL 'WAS' Function *(improved since EAP 5)*

JIRA 4.3 introduces the ability to search the Change History of issues. There is much more functionality to follow, but in this release you can search for changes to the Status field.

For example, the following will return all issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

In Beta 1, support for the `NOT`, `IN` and `NOT IN` operators has been added.

E.g. to find issues that have never had a status of 'In Progress':

```
status WAS NOT "In Progress"
```

To find issues that currently have, or previously had, a status of 'In Progress' or 'Resolved':

```
status WAS IN "In Progress,Resolved"
```

^Top

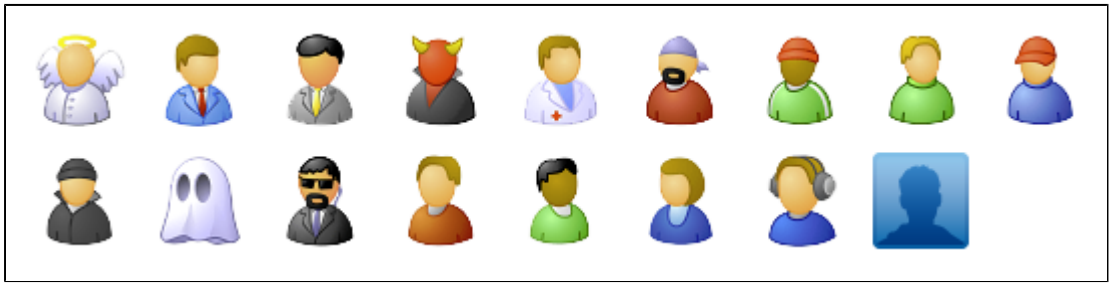
8

Revamped User Avatars

JIRA 4.3 introduces the new-look Atlassian avatars:

- User avatars are displayed as the icon for your profile, and to illustrate your comments on an issue. See the documentation on Adding a User Avatar.

We were fond of the old avatars, but think you'll agree they were looking a little dated by comparison:



^Top

9

Enhancements to Quick Search

When using Quick Search to find issues with a particular version, you can now use the wildcard symbol: "`*`". For example, "`ff:3.2*`" will match any issue whose Fix For Version is:

- 3.2
- 3.2-beta
- 3.2.1
- 3.2.x

You can also find issues reported by you, another user or with no reporter, using the prefix "`r:`" followed by a specific reporter term such as '`me`", a username or "`none`", such that:

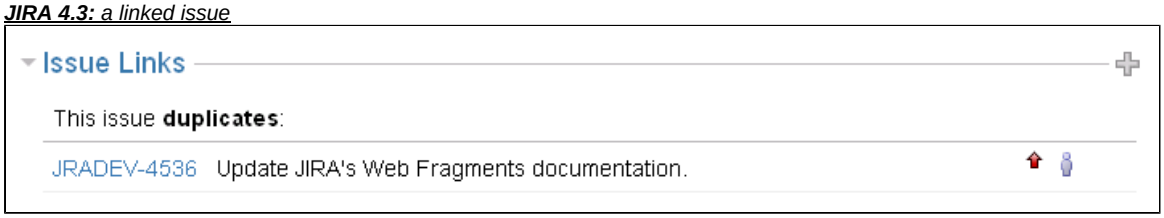
- "`r:me`" — finds issues reported by you.
- "`r:samuel`" — finds issues reported by the user whose username is "samuel".
- "`r:none`" — finds issues with no reporter.

^Top

Improvements to Issue Links *(new since EAP 5)*

The 'Linked Issues' section of the 'View Issue' screen has been streamlined to make it both more compact and quicker to use. You can now delete links directly from this screen:

JIRA 4.3: *a linked issue*



Previously, to delete a link you needed to click the 'cog' icon (which took you to a separate screen):

JIRA 4.2: *a linked issue*

^Top

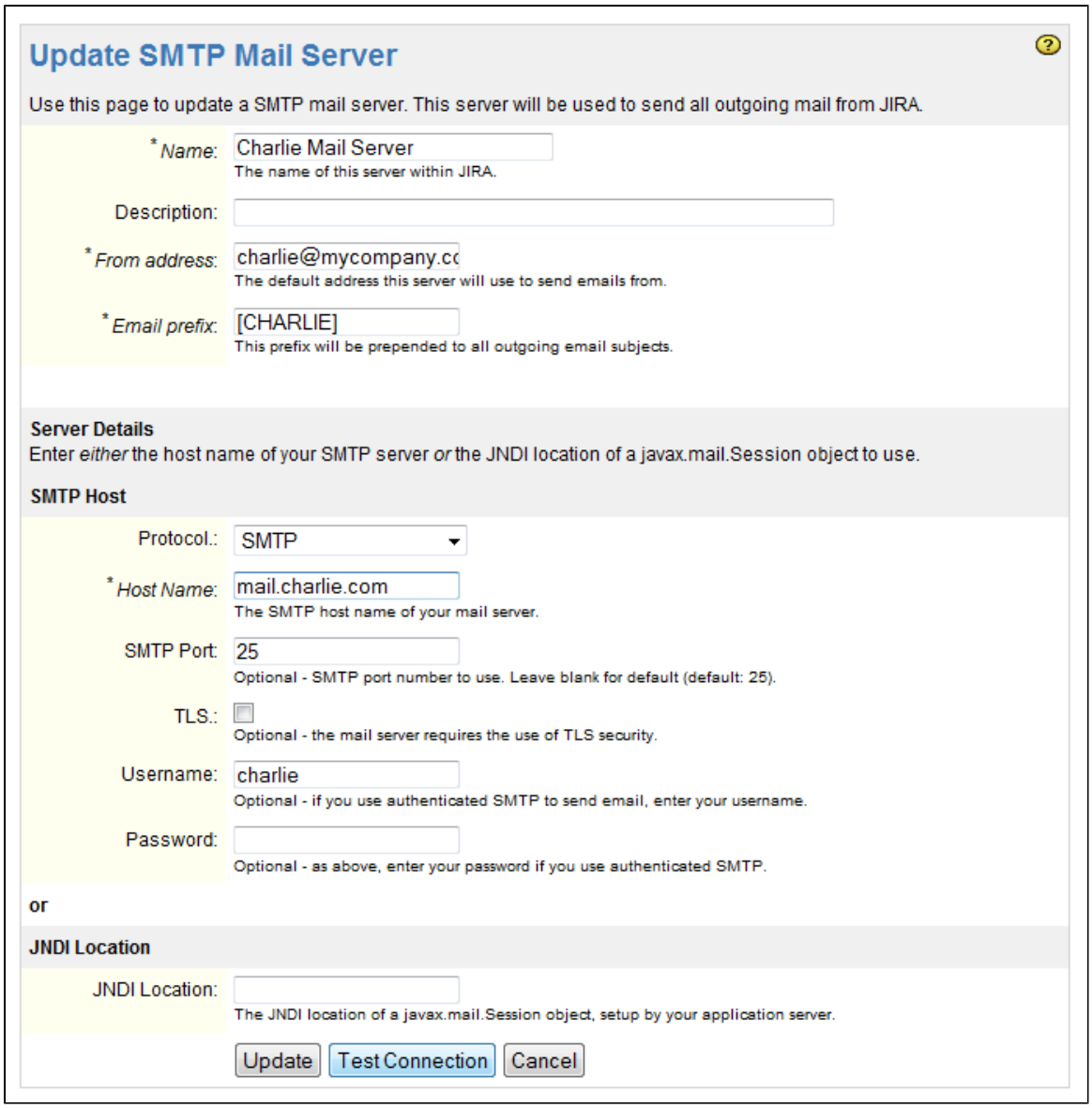
Remembered Assignees

JIRA now makes it easier to assign users to issues. The top of the assignee dropdown list shows the last five people you assigned issues to, as well as the reporter of the issue and all participants involved in the issue.

^Top

Mail Server Configuration Improvements

In JIRA 4.3, administrators can now test their mail server configuration with the "Test Connection" button.



A '**Timeout**' field has also been added.

^Top

User Email Change is now Password Protected

For enhanced security, users will now be prompted to enter their password when changing their email address.

^Top

Support for Chrome and Safari 5 Browsers

We are very pleased to announce that Chrome and Safari 5 are supported for use with JIRA 4.3.

^Top

REST API improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance. Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3 so far, click here.

^Top

JIRA 4.2 RC1 Release Notes

12 October 2010

JIRA 4.2 RC1 is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

A **Release Candidate (RC)** is a preliminary release leading up to the official release of a JIRA version. RC releases are a fairly stable snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use RC releases to test and fix their plugins in advance of an official release.

Do not use in production

RC releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **RC releases may not be safe** — RC releases are snapshots of the ongoing JIRA development process. As such:
  - While we deem these releases to be fairly stable, they have not undergone sufficient testing to be deemed an official (final) release.
  - While feature development work is complete, some features may be subject to change or be removed before the next full release.
- **No upgrade path** — Because RC releases still represent work in progress, we **cannot** provide a supported upgrade path between RC releases, from Beta to RC releases, or from any RC to the eventual final release. Thus, any data you store in a JIRA RC release may not be able to be migrated to a future JIRA release.

The Atlassian team is proud to bring you the JIRA 4.2 RC1 release. As always, we would love to hear your feedback and comments about this release. To keep the discussion centralised, please provide your feedback as a comment on JRA-22093.

JIRA 4.2 gives you a few extra minutes of precious time every day, by providing the ability to triage issues directly from the Issue Navigator without you having to open each issue. For the mouse-averse, the new 'Operations Dialog' box provides access to all menu options via the keyboard.

To help speed up your ability to work on issues, we've also made common actions and workflow operations accessible from convenient dialog boxes.

Time-tracking has become much more flexible: you can now edit the Original Estimate and set the Remaining Estimate to zero when resolving an issue. You can also log work via workflow 'transition' (or any other) screens — now accessible via speedy dialog boxes.

The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users and administrators — and everyone else who likes to label their JIRA issues.

GreenHopper 5.3 RC1 is also available for download, and is compatible with JIRA 4.2 RC1. A feature highlight coming up in GreenHopper 5.3 are GreenHopper-specific keyboard shortcuts.

JIRA 4.2 RC1 differs from the Beta 3 release in the following respects:

- We've fixed 209 bugs, including 56 relating to Internet Explorer and 10 relating to the REST API
- Improved flow and navigation through issue actions — users are more likely to land on the page from which they performed an action.
- More performance improvements have been incorporated into the auto-complete feature for multi-select fields (such as the Component and Versions fields) and the JQL auto-complete feature.

Note to developers: JIRA 4.2 includes the Atlassian Plugin Framework version 2.5, and an alpha release of the JIRA REST API. We are very keen to hear your feedback on the REST API — please try it out and add your comments to JRA-22139.

Supported Platforms: Please note from our announcements in late 2009/early 2010 that JIRA 4.2 will not support Oracle WebLogic, IBM WebSphere, Java Platform 5 and Internet Explorer 6.

Highlights of JIRA 4.2:

- Dialogs for Common Actions and Workflow Operations
- Keyboard Shortcuts and 'Operations Dialog'
- Issue Labelling
- Improvements to the 'View Issue' Screen
- Auto-complete for Versions and Components
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- 'Filter' Gadget creation via the Issue Navigator
- User Avatars and Hover Profile
- New JQL Functions
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.2 RC1
Download RC1

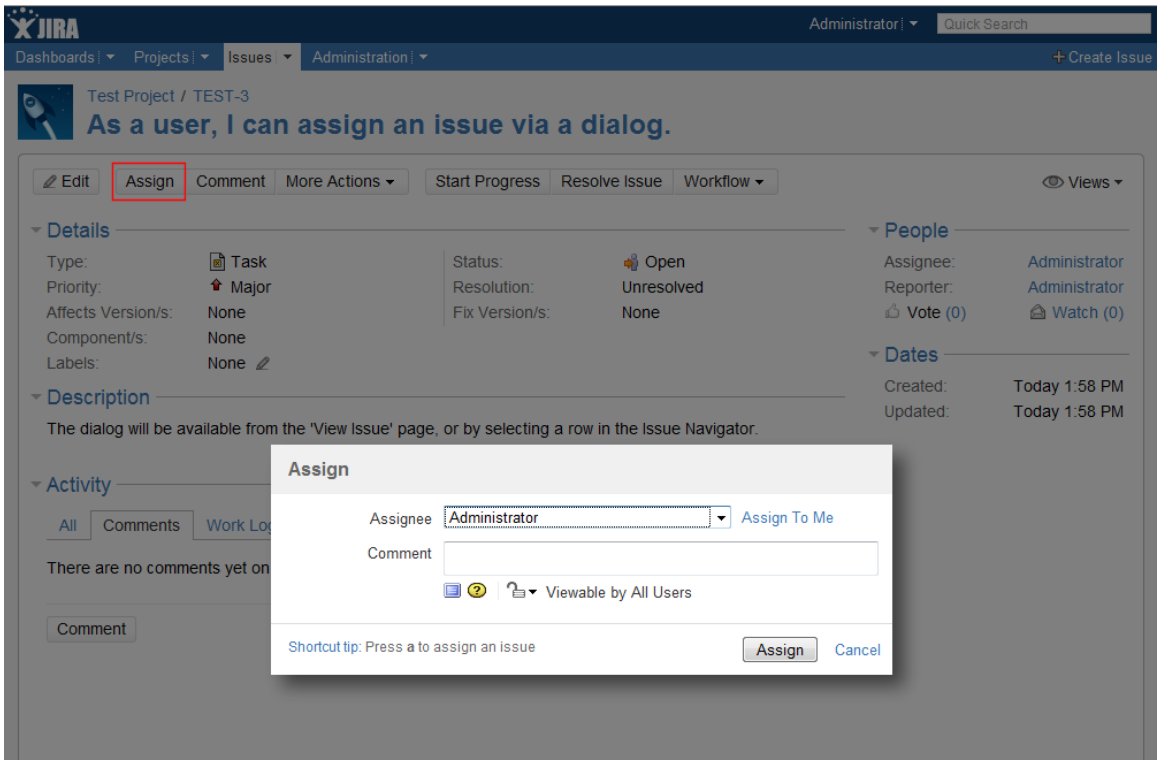
Upgrading to JIRA 4.2 RC1

Development releases of JIRA are available here. Before upgrading, please refer to the JIRA 4.2 Upgrade Guide.

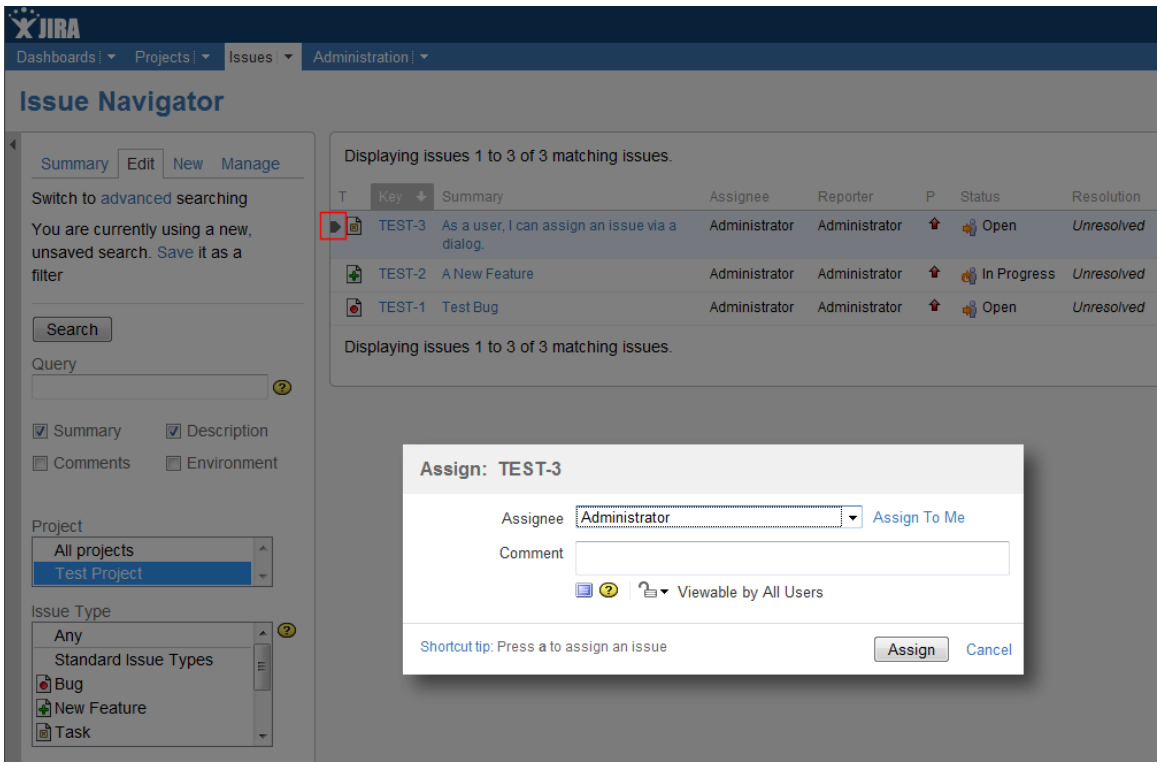
Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

For faster edits to an issue, pop-up dialogs have replaced screen changes for common actions and workflow operations.



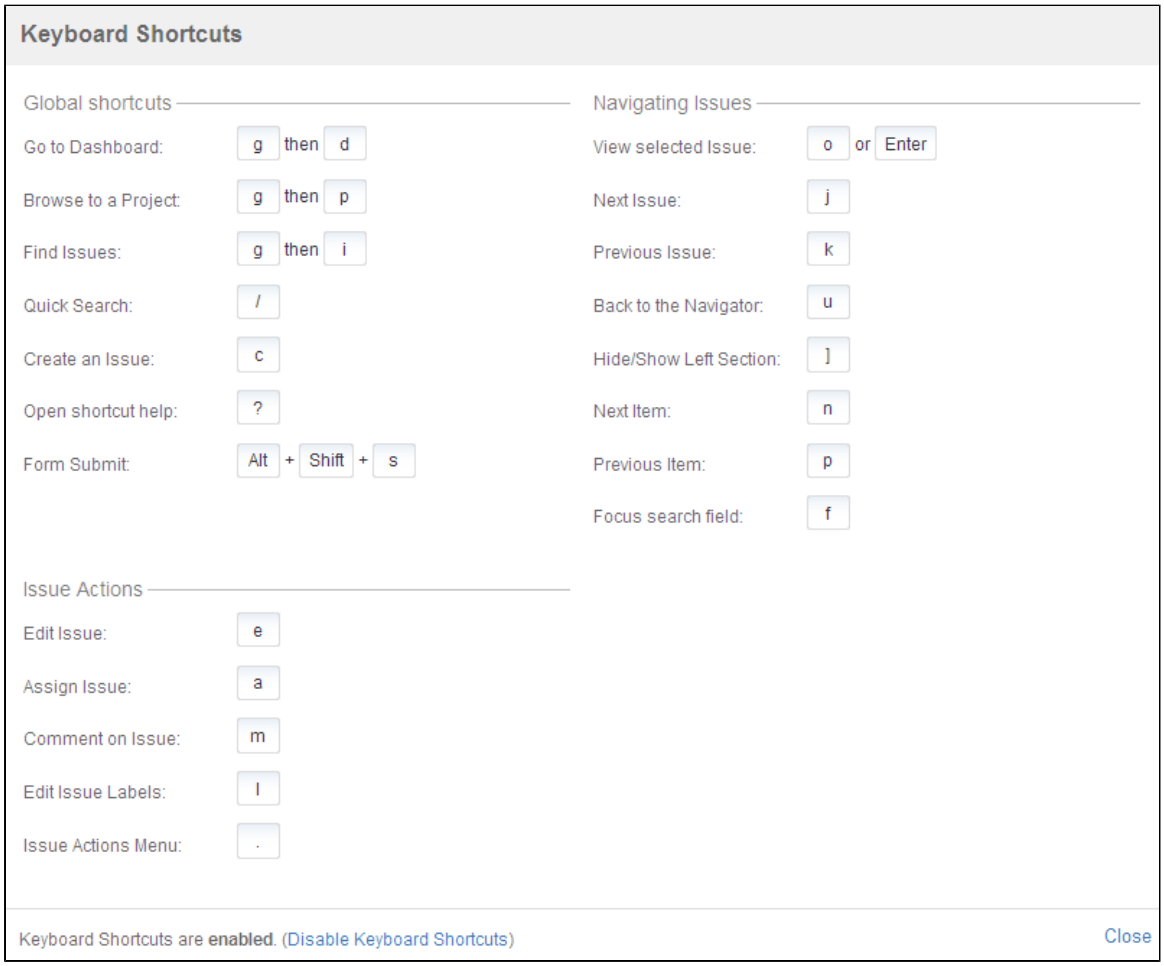
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



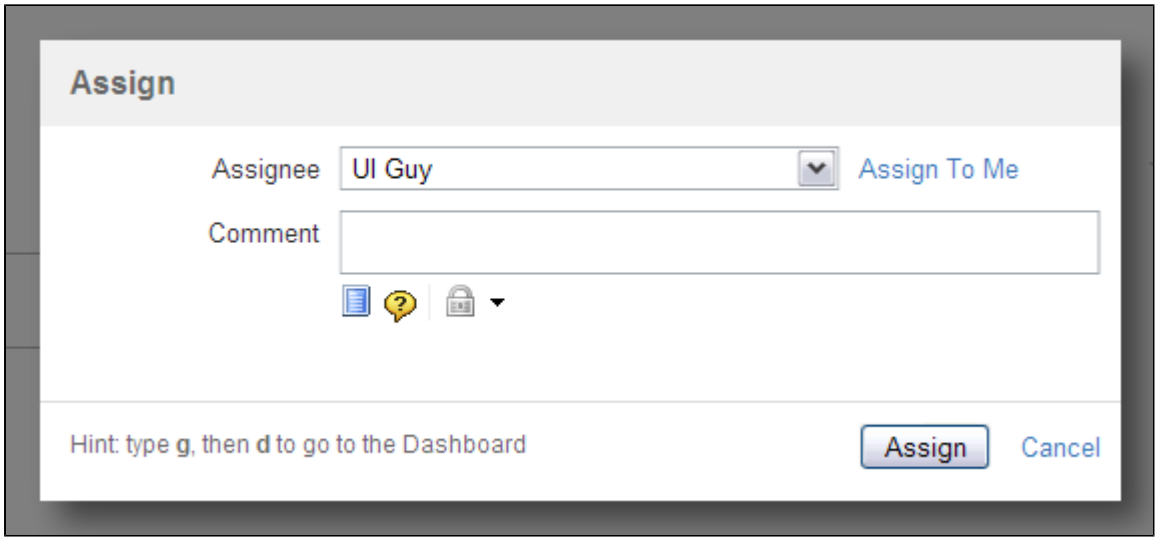
^Top

Keyboard Shortcuts and 'Operations Dialog'

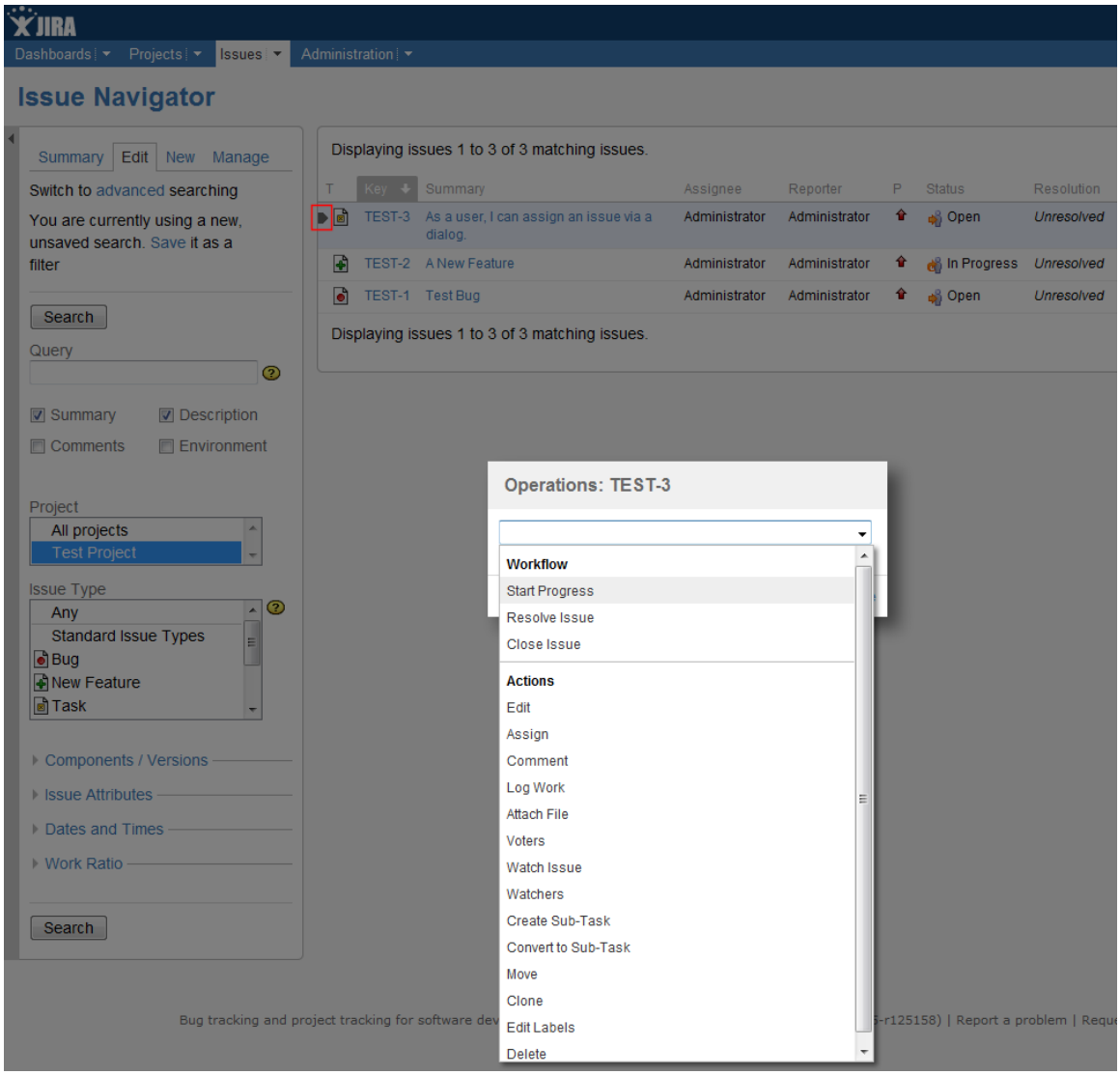
JIRA users can now do more without a mouse — perfect for power users:



Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly:



The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.

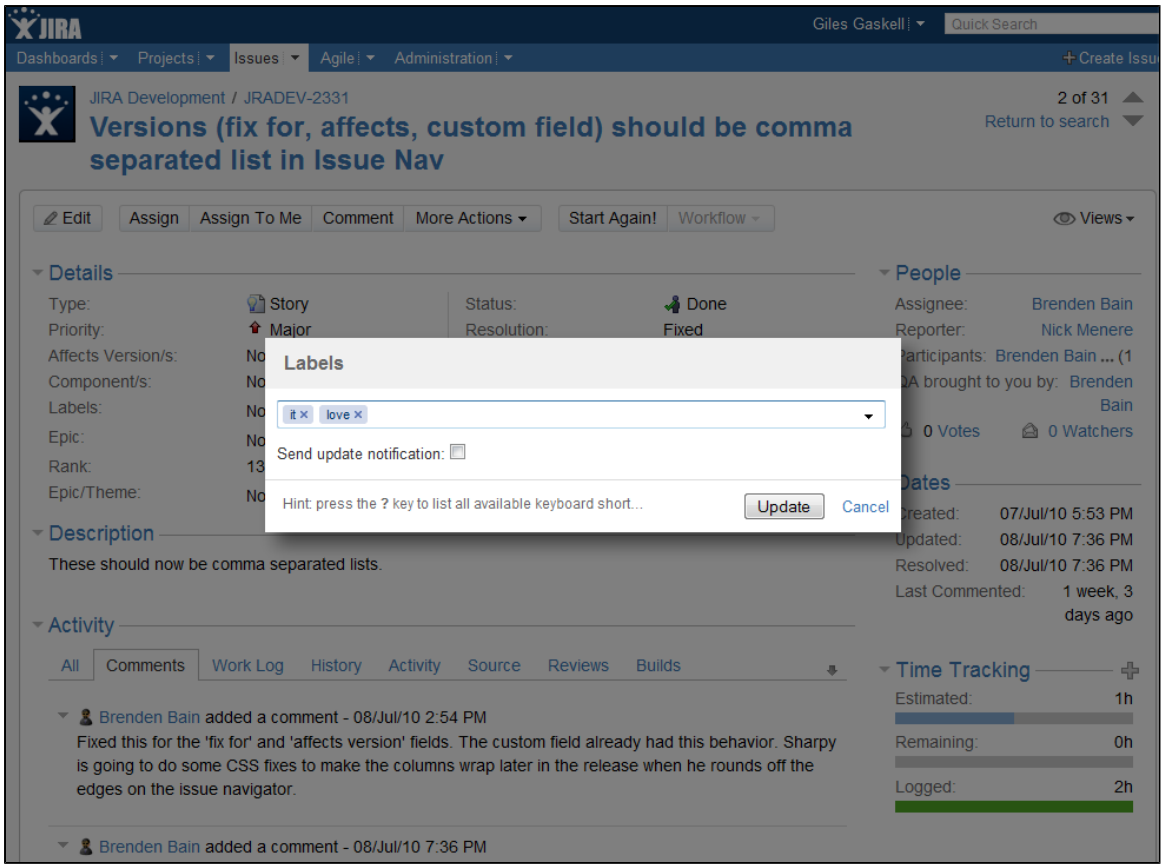


See Using Keyboard Shortcuts for more information.

^Top

Issue Labelling

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.



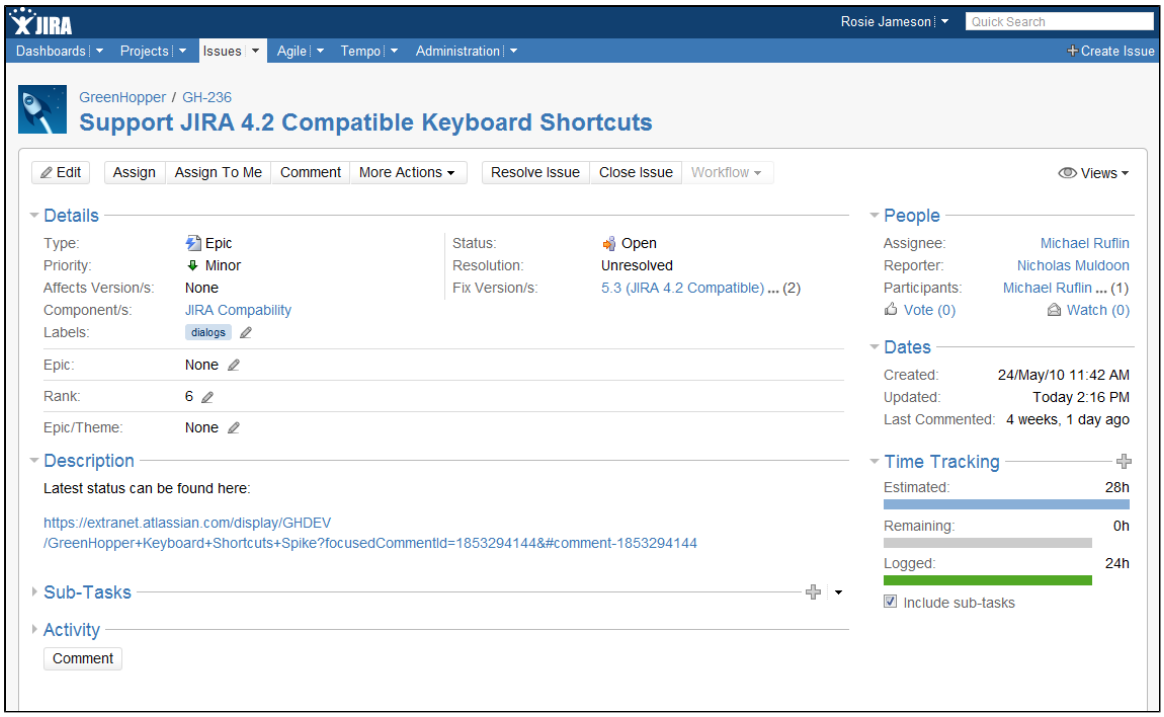
See Labelling an Issue for more information.

^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the 'View Issue' screen even more. Some improvements include:

- Tighter spacing between lines to reduce space between elements
- Stronger section headings
- The 'Description' field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



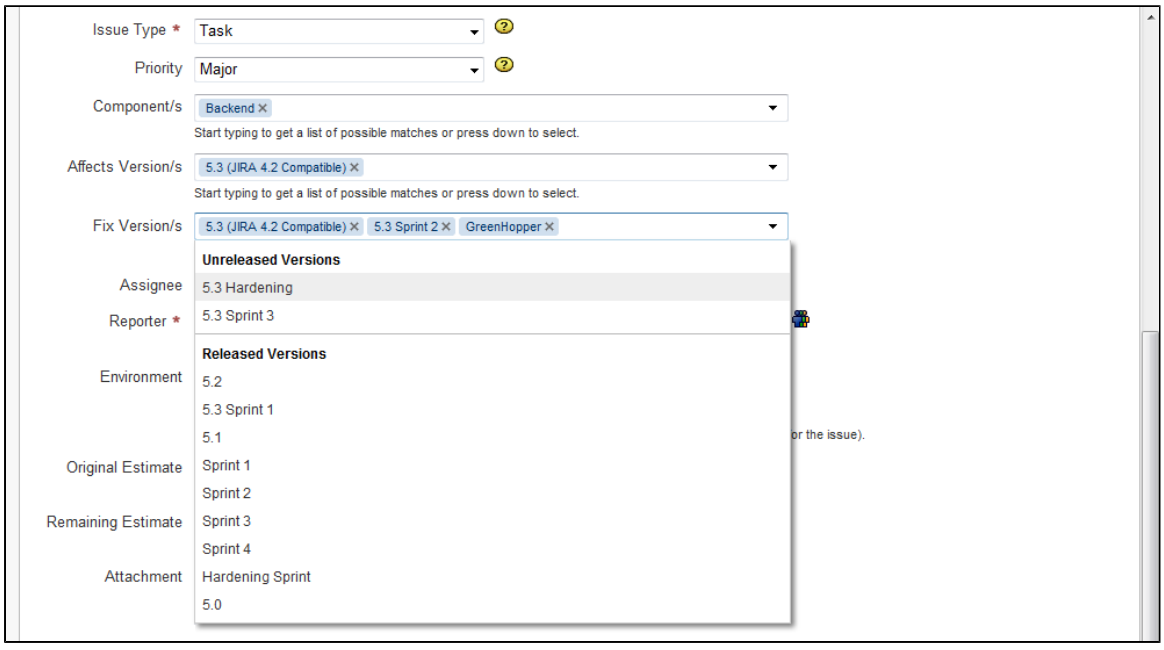
Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the '**Comments**' section.

^Top

Auto-complete for Versions and Components

Editing the **Component**, **Affects Version(s)** and **Fix Version(s)** is now quicker and easier. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

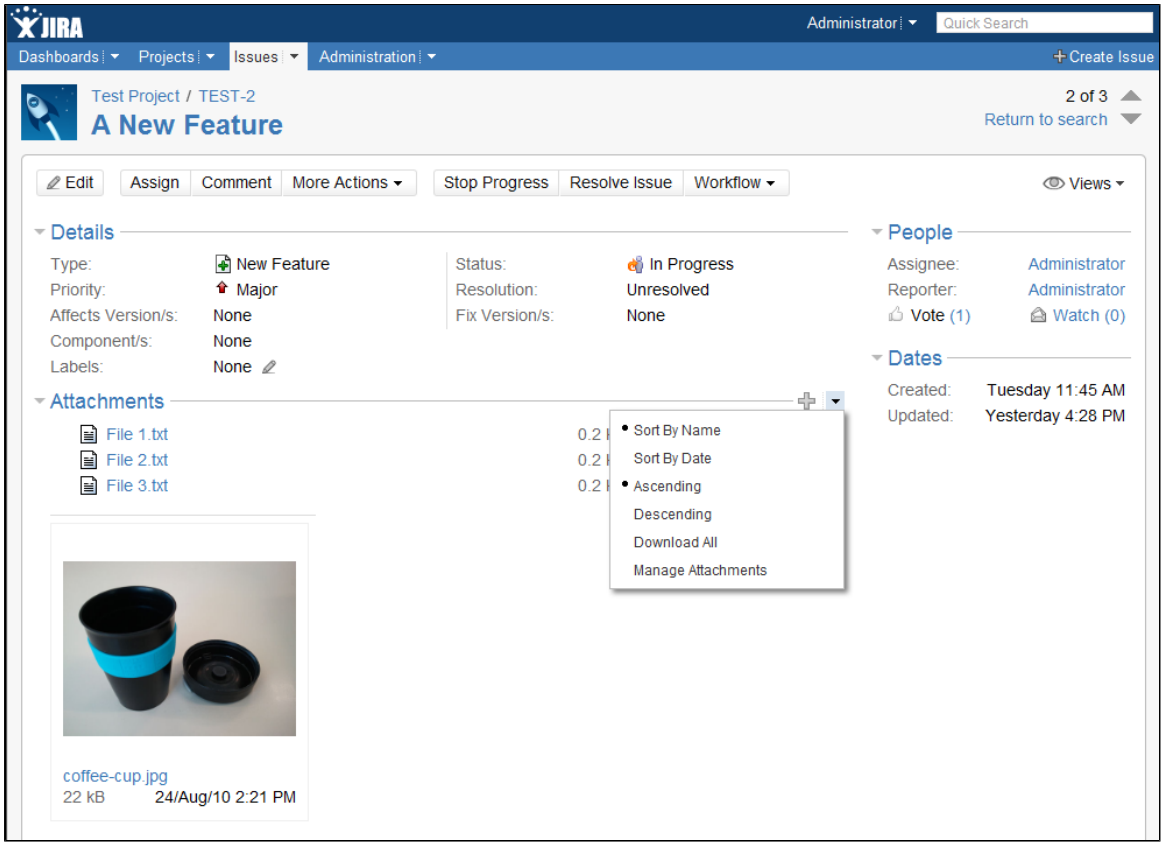
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

Attachments can now be sorted by date or by name, in ascending or descending order.

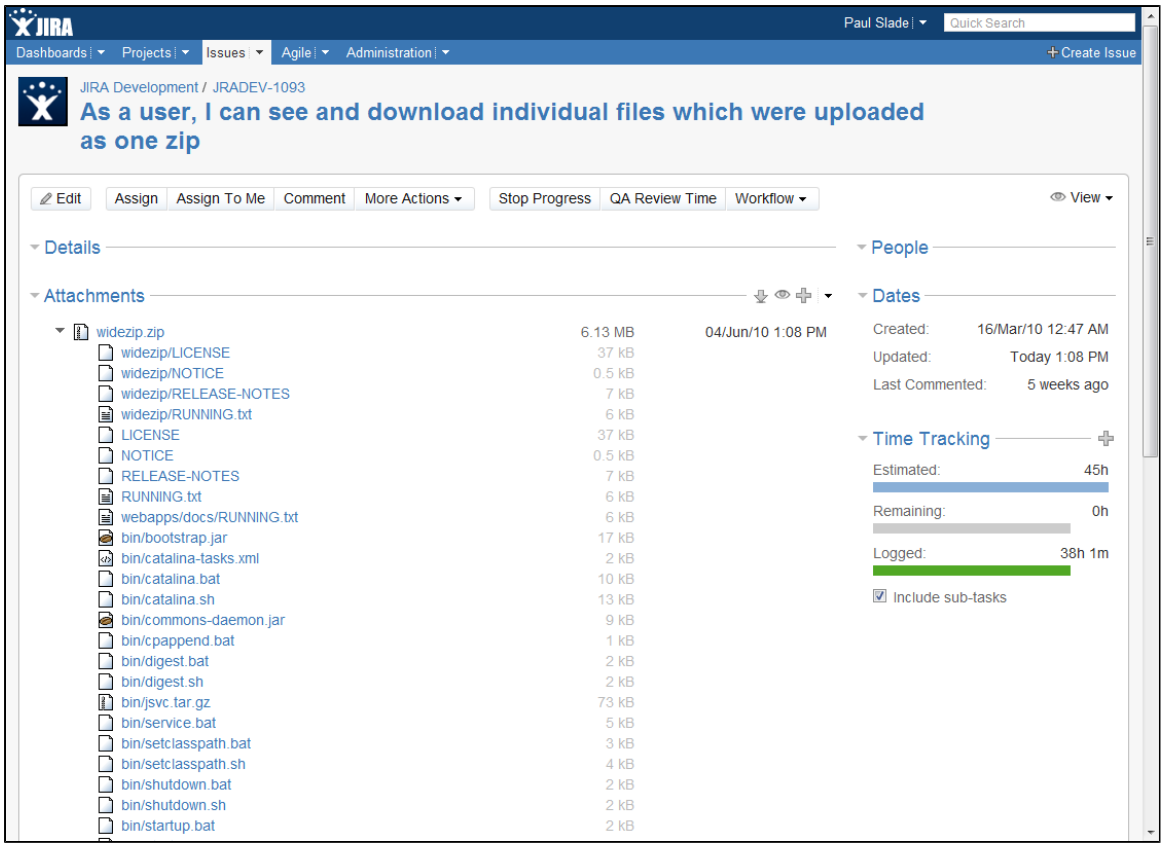


See Attaching a File for more information.

^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



See Accessing ZIP-format File Contents for more information.

^Top

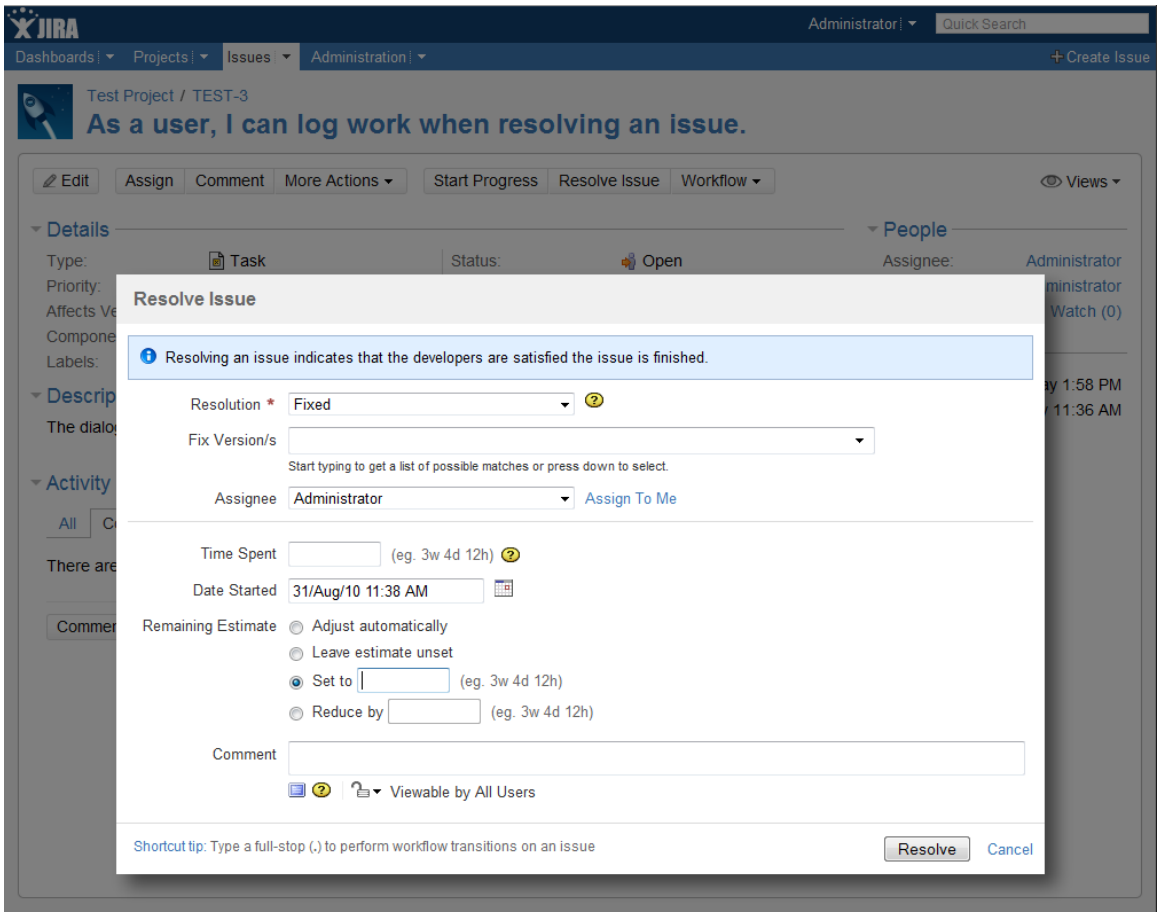
'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).

Lastly you can add the 'Log Work' field to any screen in JIRA (not shown here).



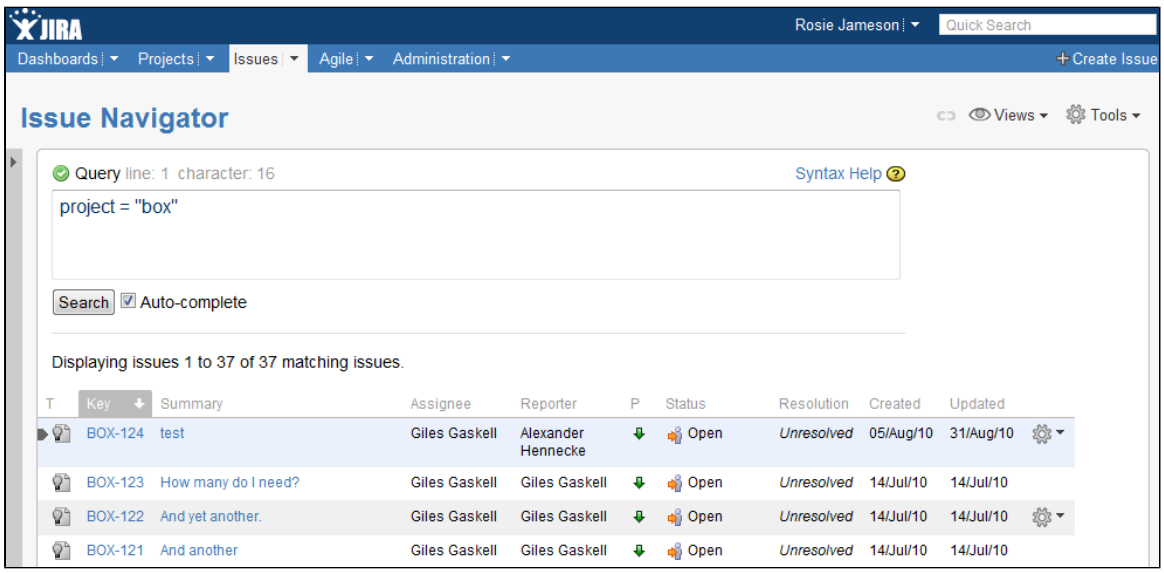
See Logging Work on an Issue for more information.

^Top

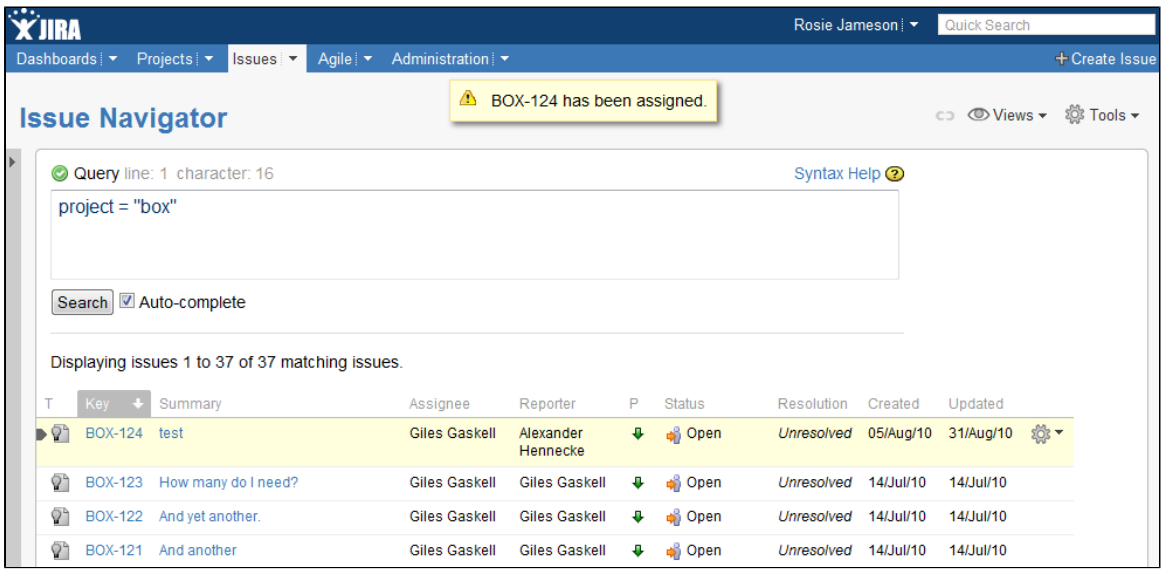
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



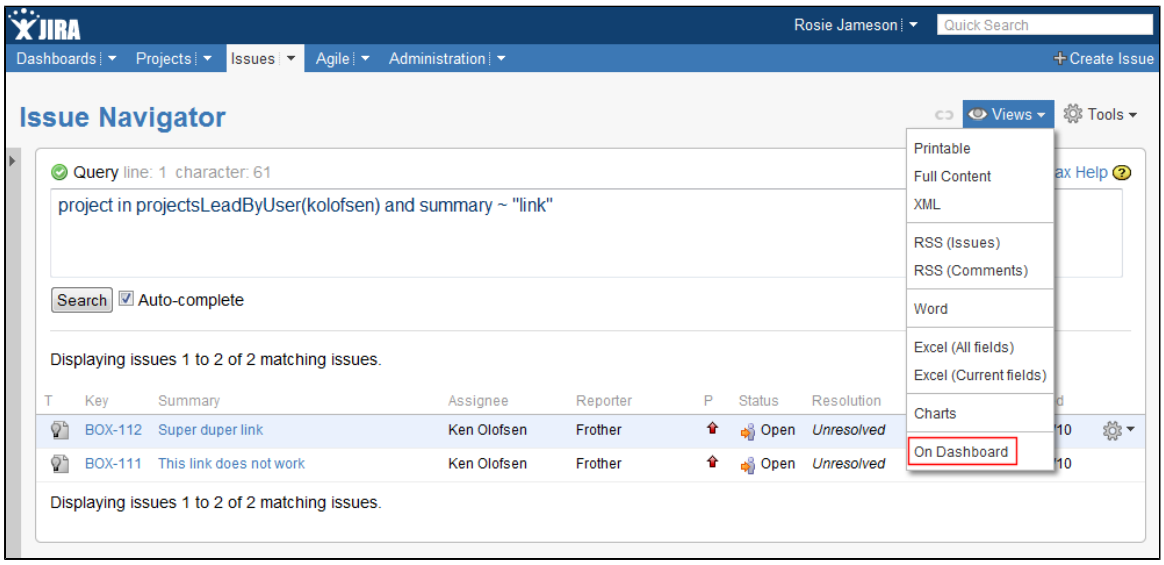
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

'Filter' Gadget creation via the Issue Navigator

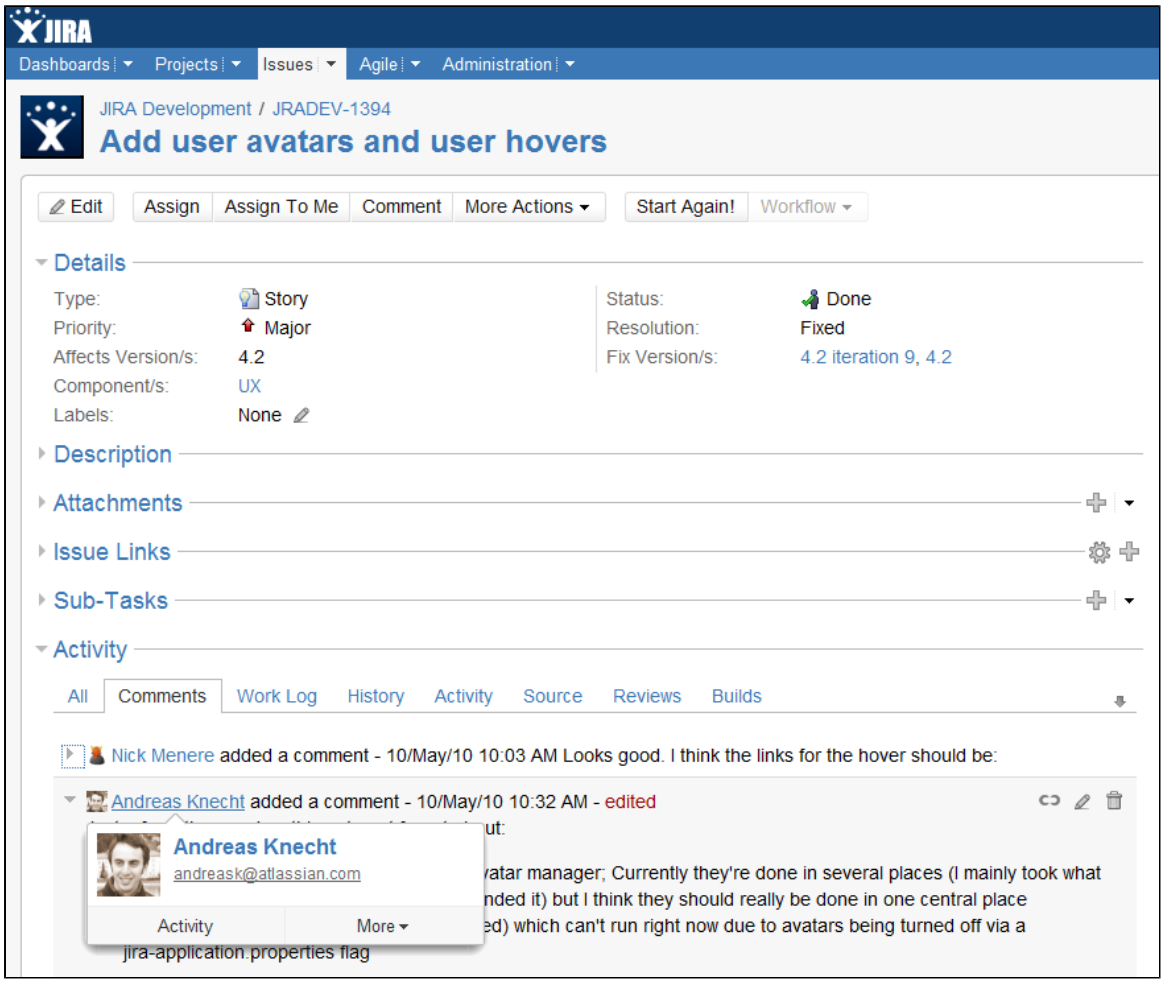
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



^Top

User Avatars and Hover Profile

JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.

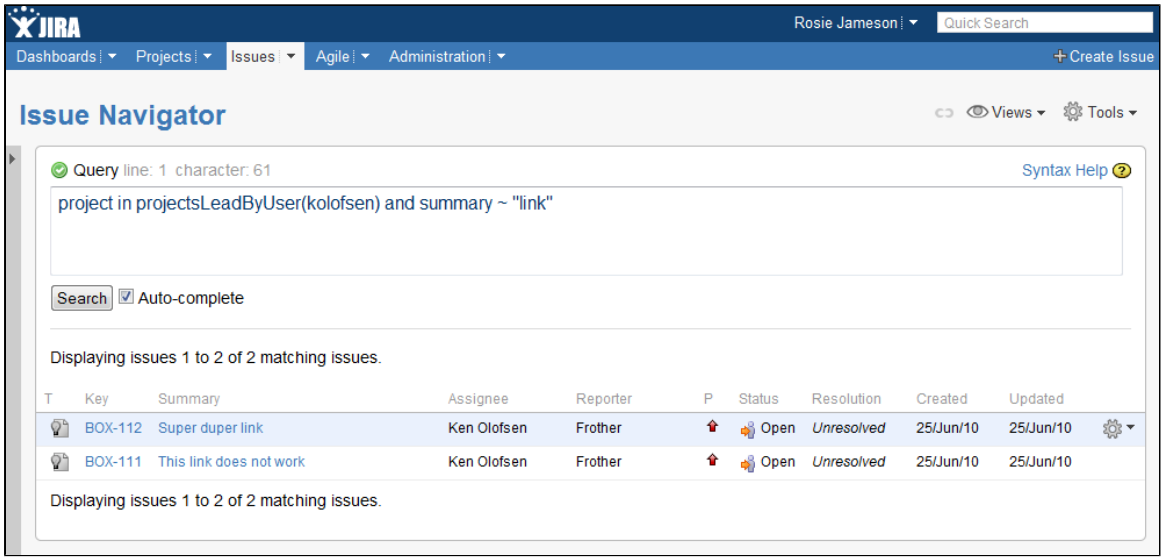


See Adding a User Avatar and Using Hover Profile for more information.

^Top

New JQL Functions

Additional JQL functions for projectsLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



See Advanced Searching for more information.

^Top

Other Enhancements and Fixes

- If you use Firefox or IE, you can now search JIRA issues from the convenience of your browser's search box. Just add your JIRA site as a search engine/provider, via the dropdown menu next to the browser's search box. This is because JIRA now supports the autodiscovery part of the OpenSearch standard.
- Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.2 Beta 3 Release Notes

29 September 2010

JIRA 4.2 Beta 3 is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

The Atlassian team is proud to bring you the JIRA 4.2 Beta 3 release. As always, we would love to hear your feedback and comments about this release. To keep the discussion centralised, please provide your feedback as a comment on JRA-22093.

JIRA 4.2 gives you a few extra minutes of precious time every day, by providing the ability to triage issues directly from the Issue Navigator without you having to open each issue. For the mouse-averse, the new 'Operations Dialog' box provides access to all menu options via the keyboard.

To help speed up your ability to work on issues, we've also made common actions and workflow operations accessible from convenient dialog boxes.

Time-tracking has become much more flexible: you can now edit the Original Estimate and set the Remaining Estimate to zero when resolving an issue. You can also log work via workflow 'transition' (or any other) screens — now accessible via speedy dialog boxes.

The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users and administrators — and everyone else who likes to label their JIRA issues.

GreenHopper 5.3 Beta 3 is also available for download, and is compatible with JIRA 4.2 Beta 3.

JIRA 4.2 Beta 3 differs from the Beta 2 release in the following respects:

- The FishEye plugin was upgraded
- The auto-complete feature for multi-select fields (such as the Component and Versions fields) contains several performance improvements and bug fixes
- The JQL auto-complete feature is now much faster and more responsive
- Several fixes were made to the '*Are you sure you want to navigate away from this page?*' confirmation message to make it less obtrusive
- The REST API contains several bug fixes

- Several web browser-specific bugs were fixed

Note to developers: JIRA 4.2 includes the Atlassian Plugin Framework version 2.5, and an alpha release of the JIRA REST API. We are very keen to hear your feedback on the REST API — please try it out and add your comments to JRA-22139.

Supported Platforms: Please note from our announcements in late 2009/early 2010 that JIRA 4.2 will not support Oracle WebLogic, IBM WebSphere, Java Platform 5 and Internet Explorer 6.

Highlights of JIRA 4.2 :

- Dialogs for Common Actions and Workflow Operations
- Keyboard Shortcuts and 'Operations Dialog'
- Issue Labelling
- Improvements to the 'View Issue' Screen
- Auto-complete for Versions and Components
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- 'Filter' Gadget creation via the Issue Navigator
- User Avatars and Hover Profile
- New JQL Functions
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.2 Beta 3
Download Beta

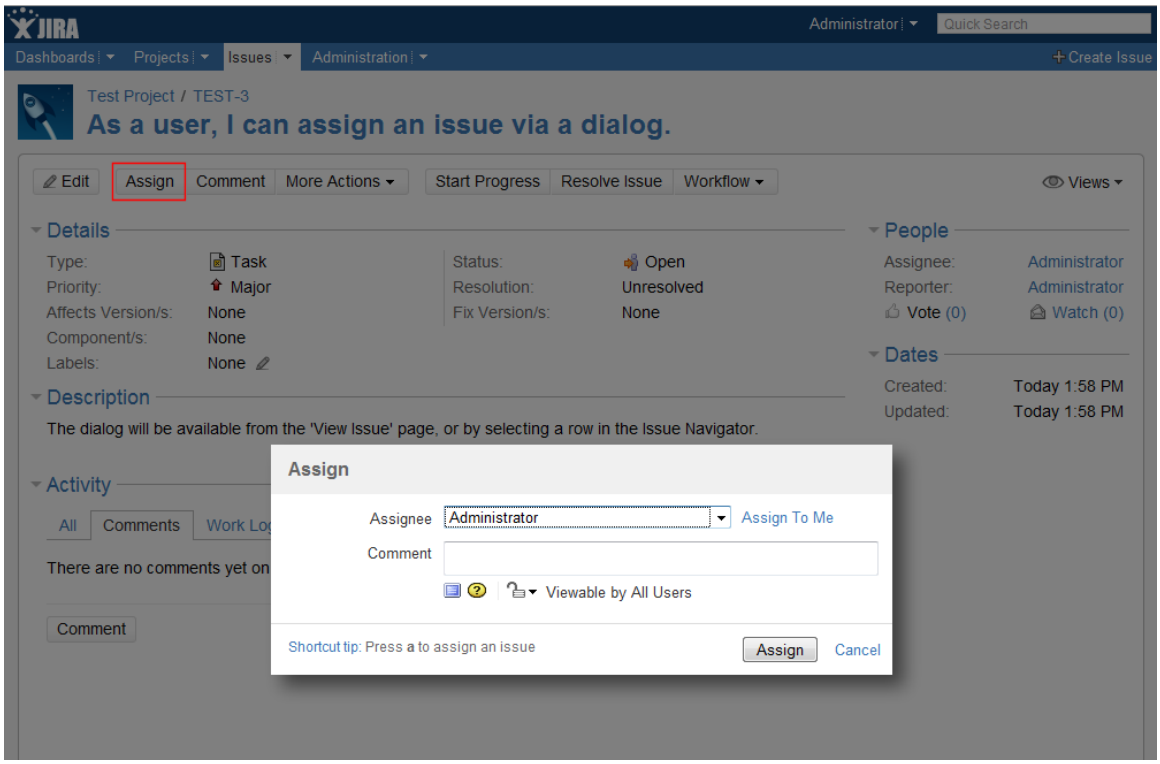
Upgrading to JIRA 4.2 Beta 3

JIRA Beta releases are available here. Before upgrading, please refer to the JIRA 4.2 Upgrade Guide.

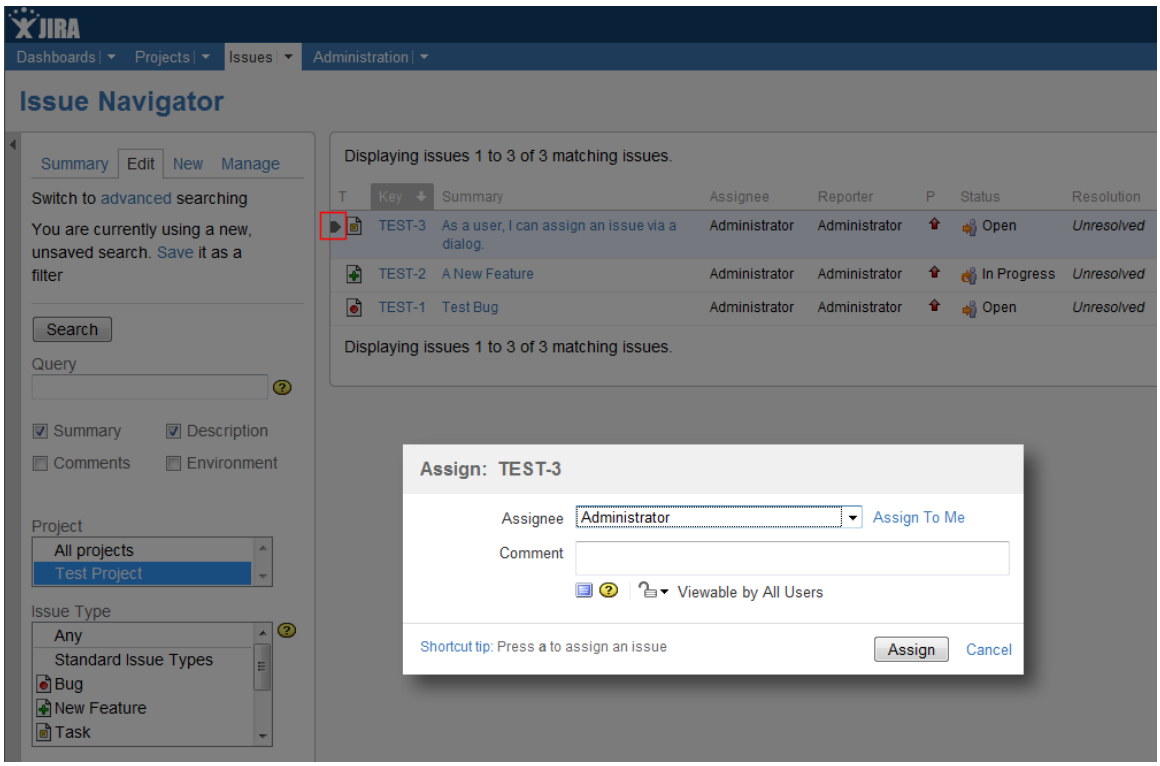
Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

For faster edits to an issue, pop-up dialogs have replaced screen changes for common actions and workflow operations.



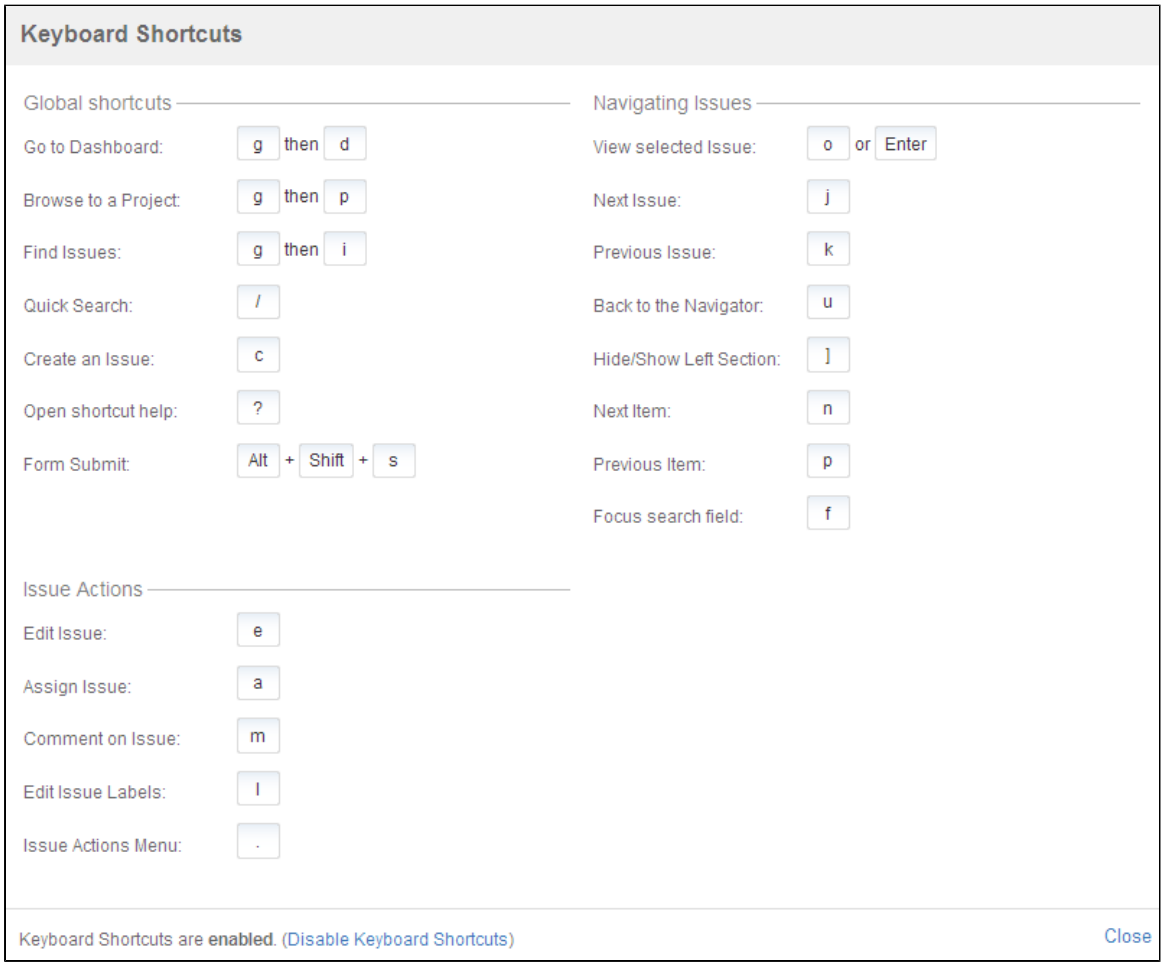
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



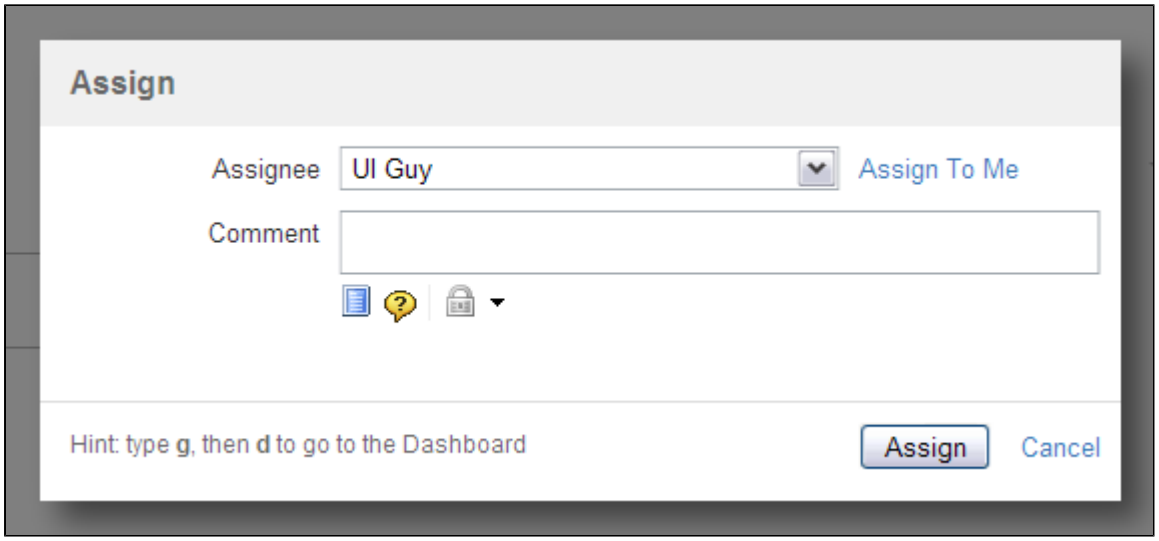
^Top

Keyboard Shortcuts and 'Operations Dialog'

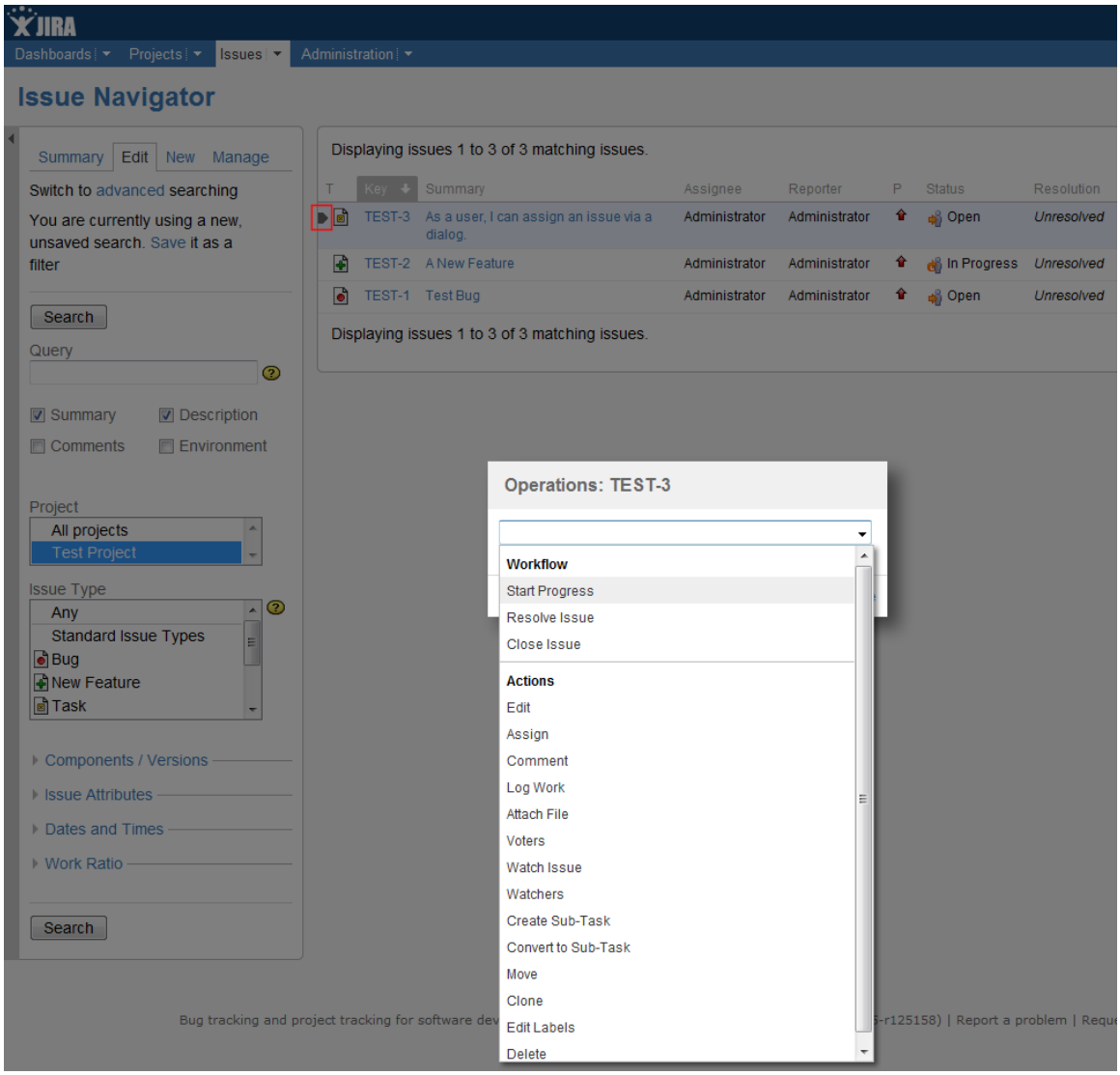
JIRA users can now do more without a mouse — perfect for power users:



Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly:



The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.

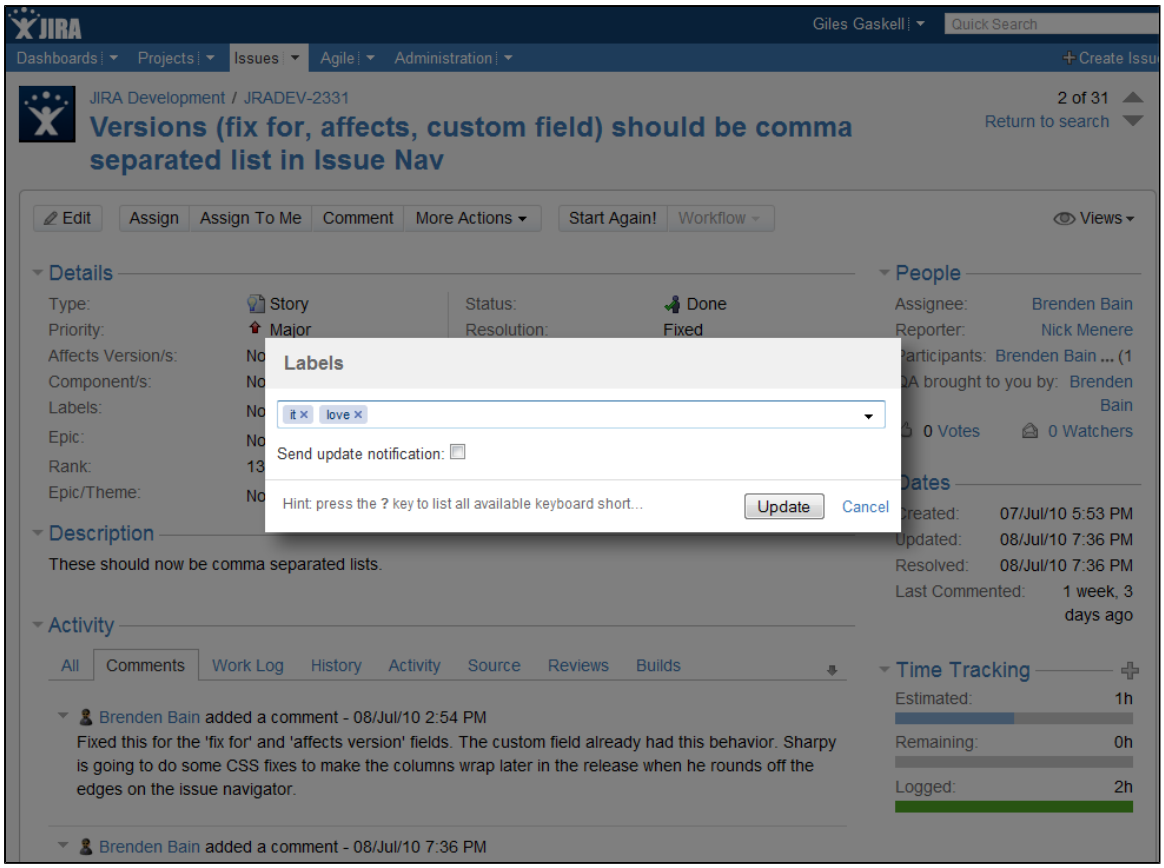


See Using Keyboard Shortcuts for more information.

^Top

Issue Labelling

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.



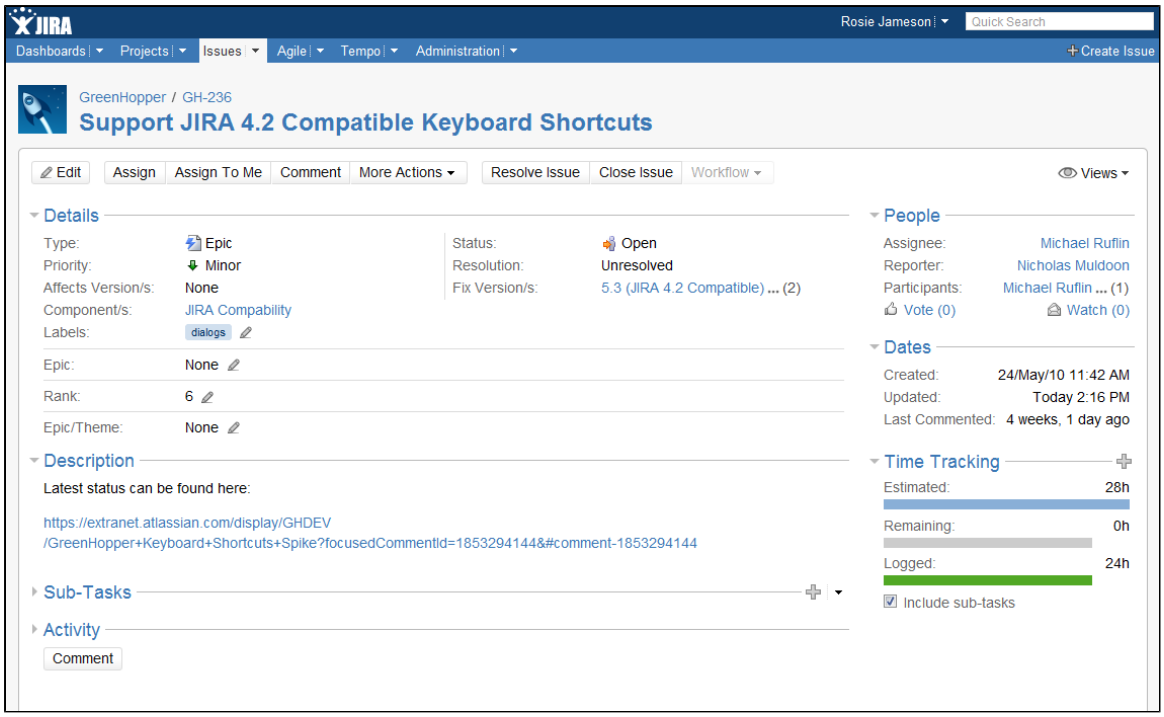
See Labelling an Issue for more information.

^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the 'View Issue' screen even more. Some improvements include:

- Tighter spacing between lines to reduce space between elements
- Stronger section headings
- The 'Description' field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



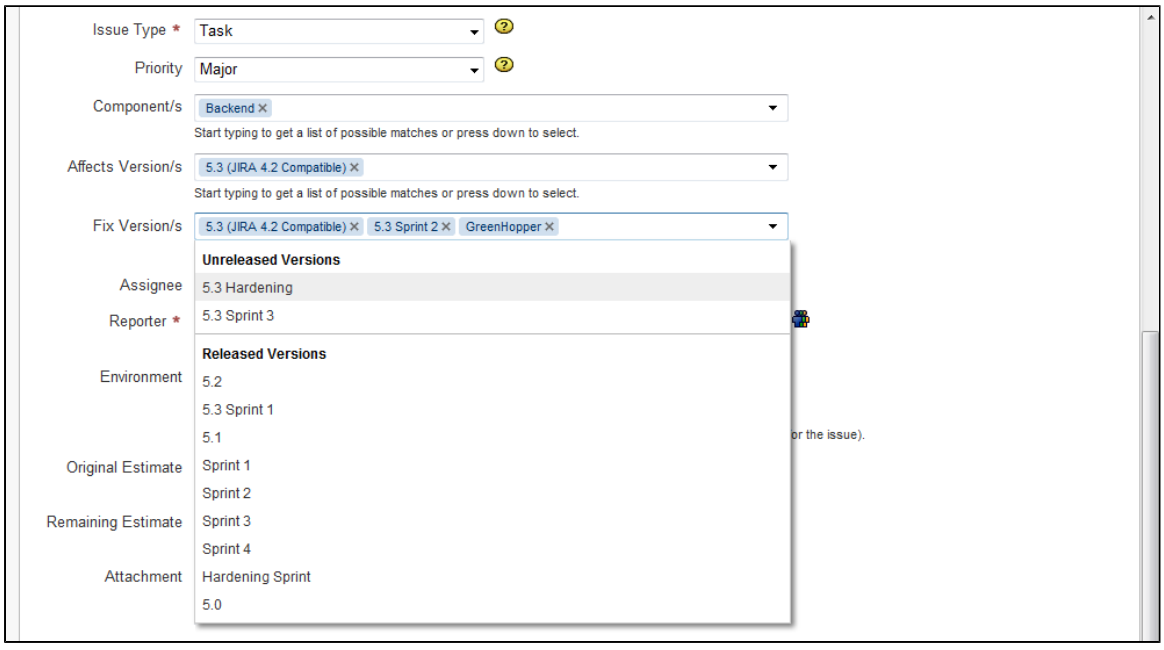
Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the '**Comments**' section.

^Top

Auto-complete for Versions and Components

Editing the **Component**, **Affects Version(s)** and **Fix Version(s)** is now quicker and easier. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

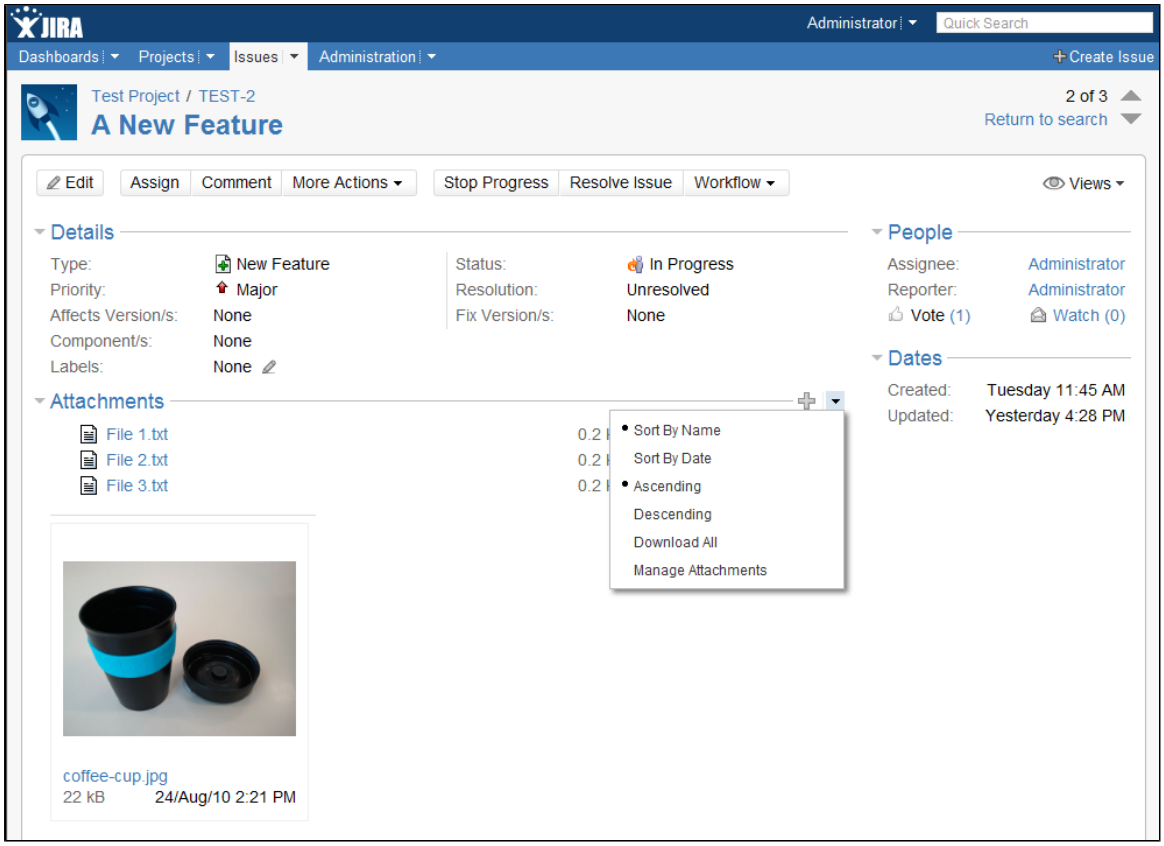
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

Attachments can now be sorted by date or by name, in ascending or descending order.

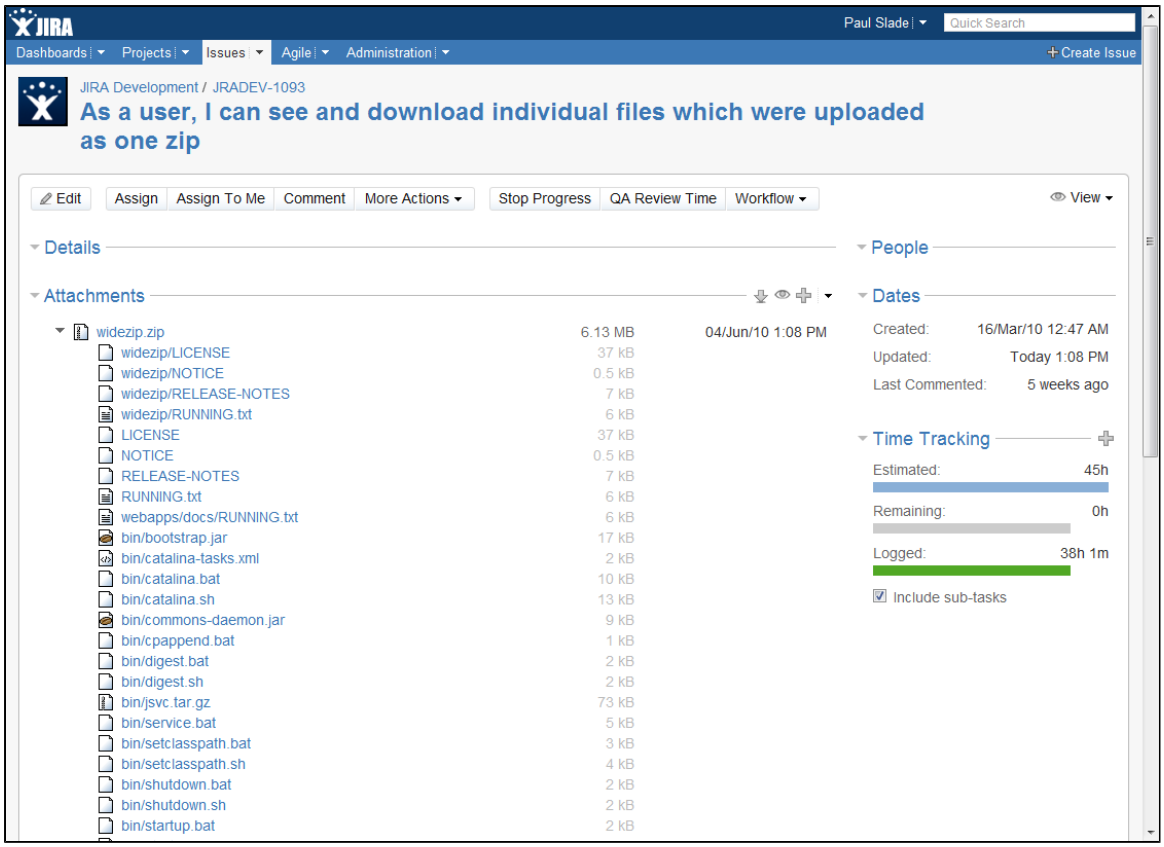


See Attaching a File for more information.

^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



See Accessing ZIP-format File Contents for more information.

^Top

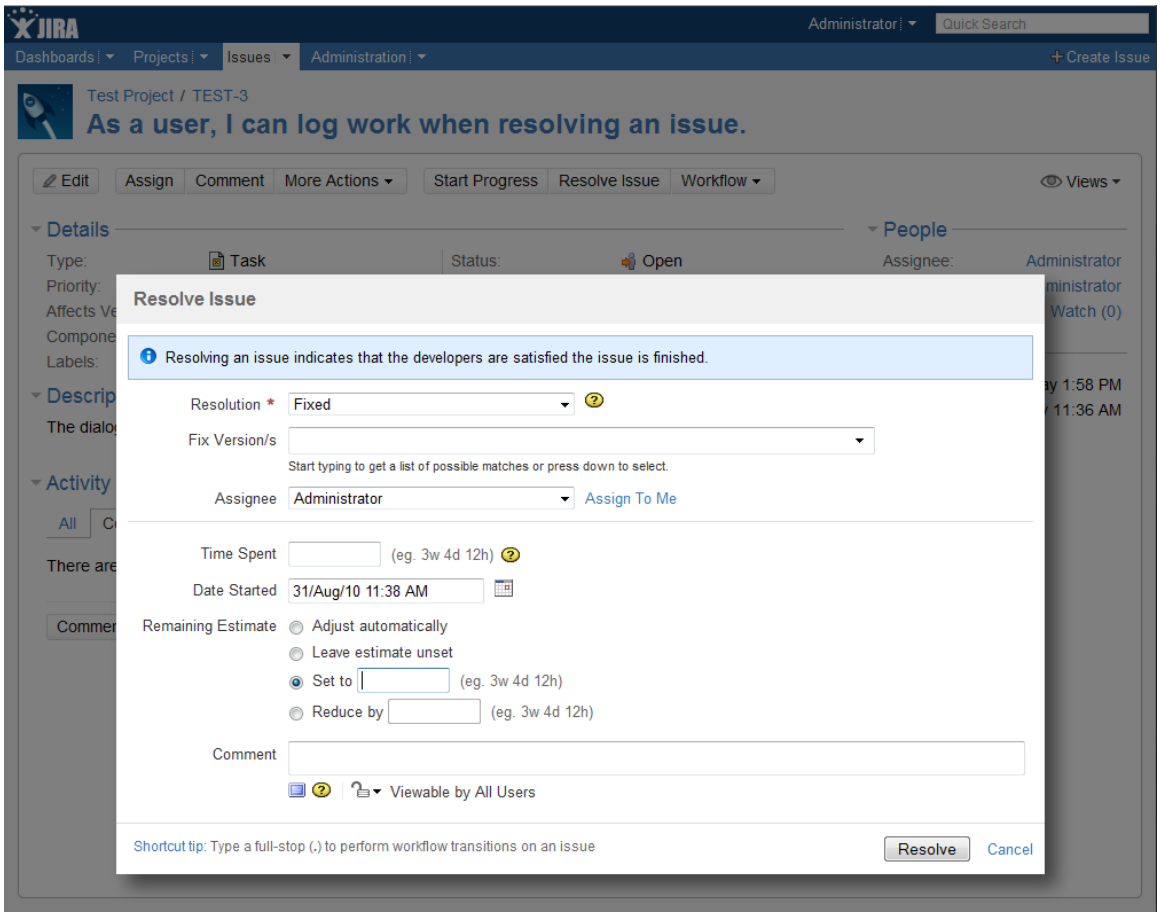
'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).

Lastly you can add the 'Log Work' field to any screen in JIRA (not shown here).



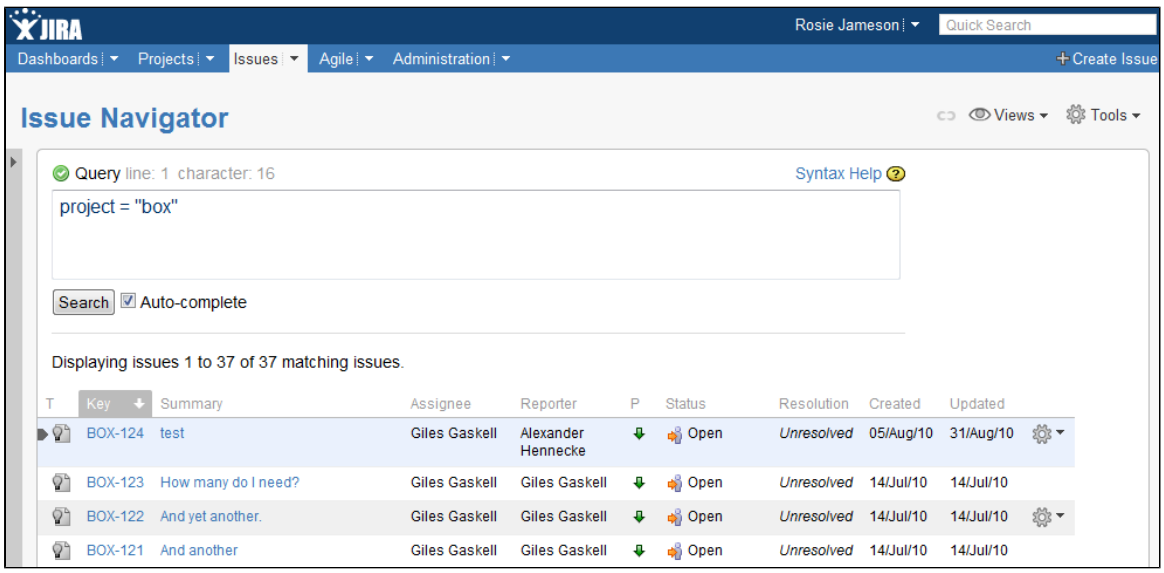
See Logging Work on an Issue for more information.

^Top

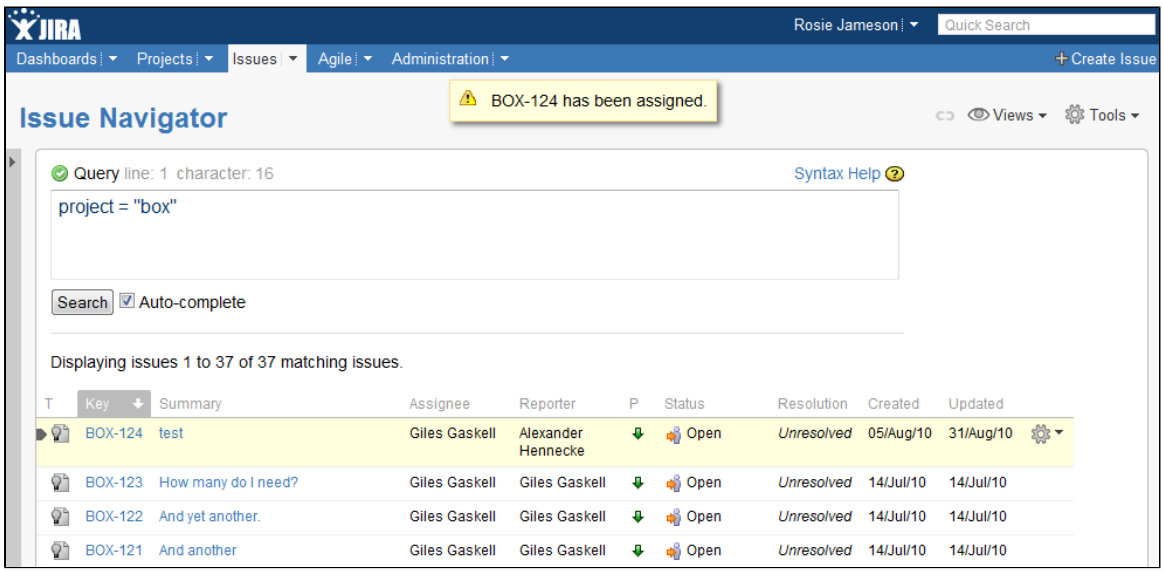
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



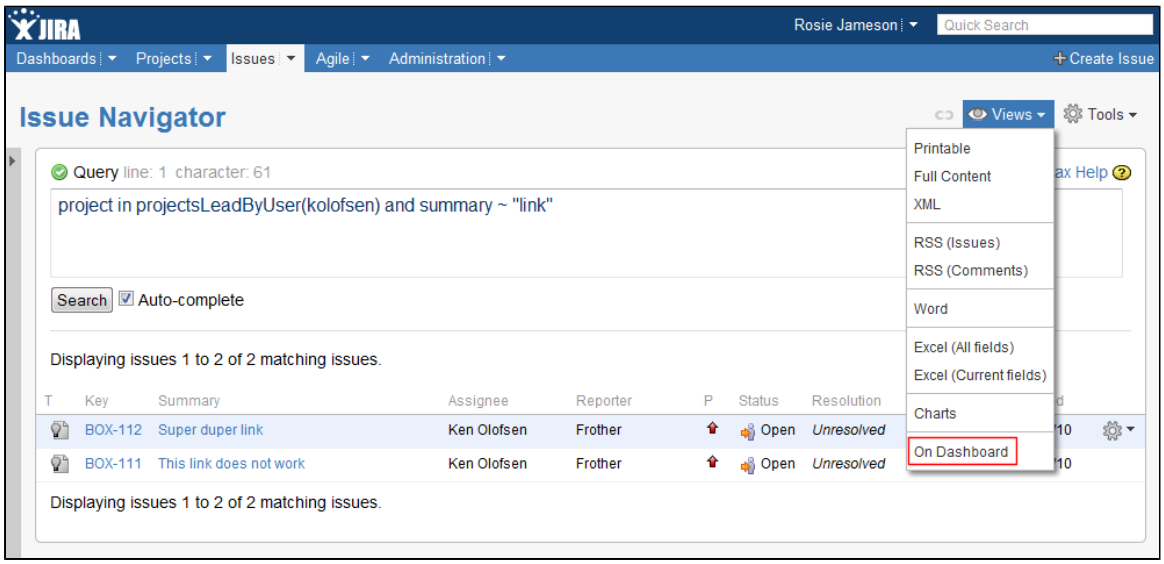
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

'Filter' Gadget creation via the Issue Navigator

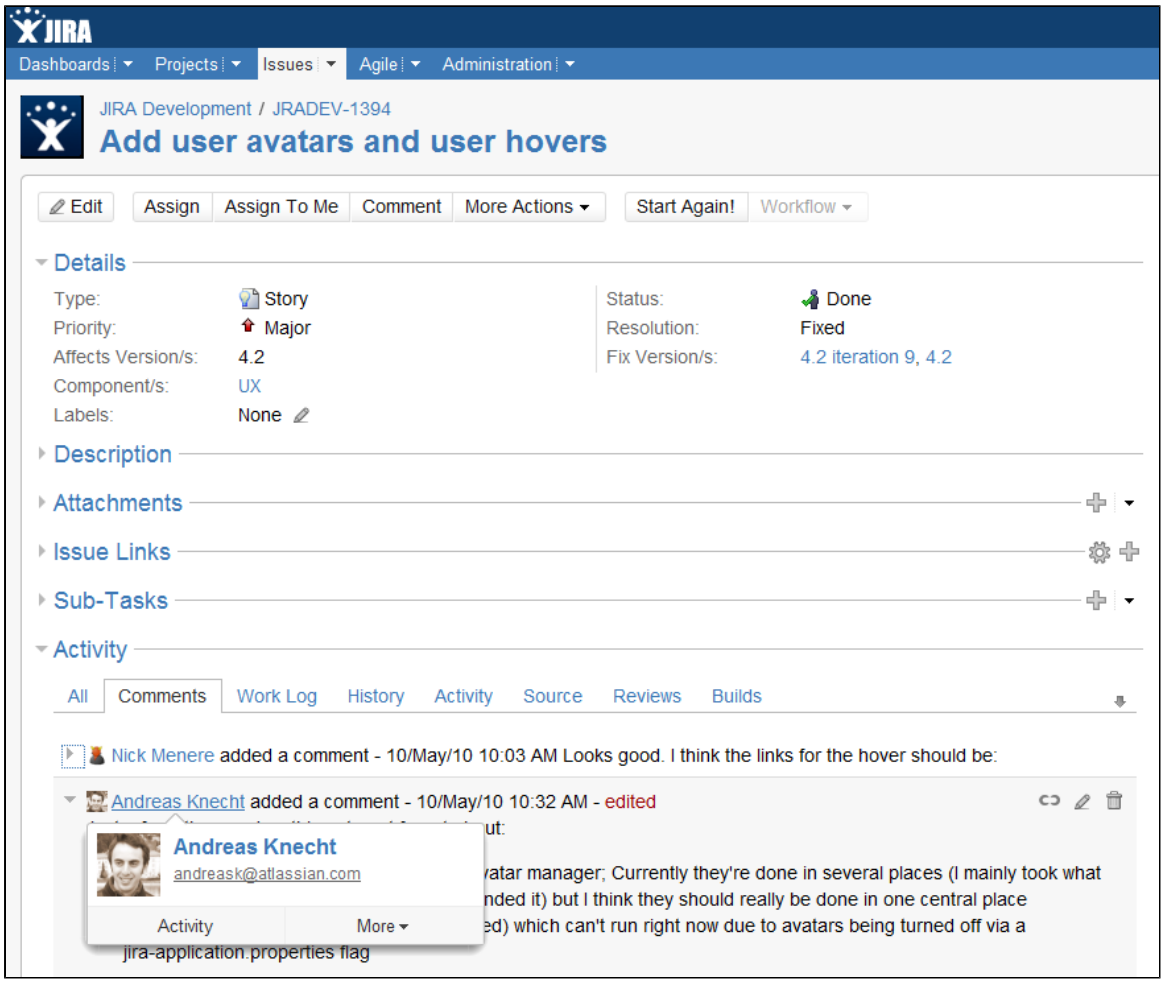
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



^Top

User Avatars and Hover Profile

JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.

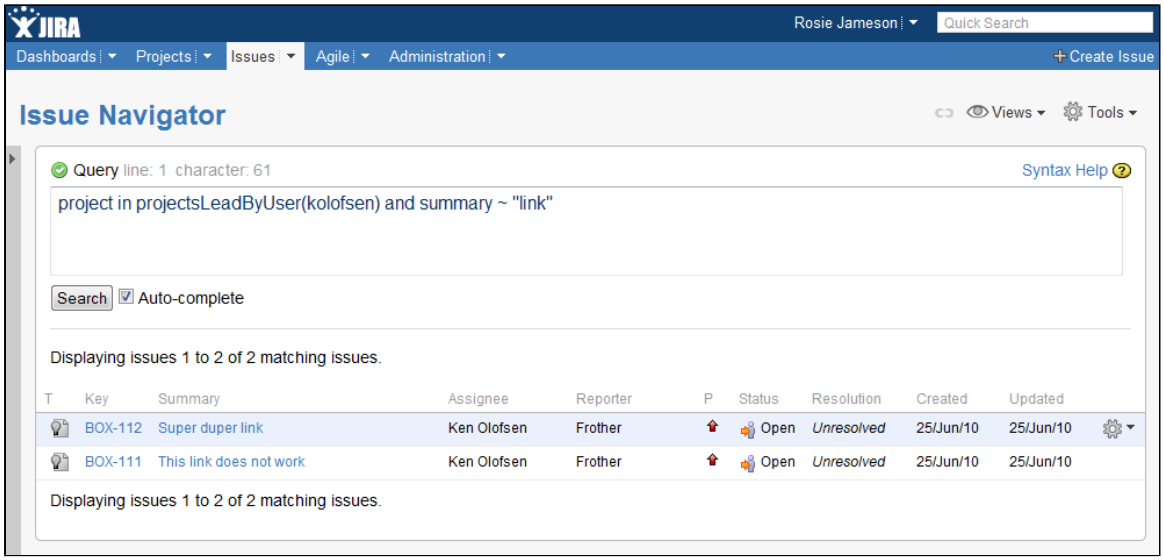


See Adding a User Avatar and Using Hover Profile for more information.

^Top

New JQL Functions

Additional JQL functions for projectsLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



See Advanced Searching for more information.

^Top

Other Enhancements and Fixes

- If you use Firefox or IE, you can now search JIRA issues from the convenience of your browser's search box. Just add your JIRA site as a search engine/provider, via the dropdown menu next to the browser's search box. This is because JIRA now supports the autodiscovery part of the OpenSearch standard.
- Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.2 Beta 2 Release Notes

15 September 2010

JIRA 4.2 Beta 2 is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

The Atlassian team is proud to bring you the JIRA 4.2 Beta 2 release. As always, we would love to hear your feedback and comments about this release. To keep the discussion centralised, please provide your feedback as a comment on JRA-22093.

JIRA 4.2 gives you a few extra minutes of precious time every day, by providing the ability to triage issues directly from the Issue Navigator without you having to open each issue. For the mouse-averse, the new 'Operations Dialog' box provides access to all menu options via the keyboard.

To help speed up your ability to work on issues, we've also made common actions and workflow operations accessible from convenient dialog boxes.

Time-tracking has become much more flexible: you can now edit the Original Estimate and set the Remaining Estimate to zero when resolving an issue. You can also log work via workflow 'transition' (or any other) screens — now accessible via speedy dialog boxes.

The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users and administrators — and everyone else who likes to label their JIRA issues.

GreenHopper 5.3 Beta 2 is also available for download, and is compatible with JIRA 4.2 Beta 2.

Differences between JIRA 4.2 Beta 1 and Beta 2:

- JIRA 4.2 Beta 2 includes a number of bug fixes, including an issue with the dialog boxes and mod_proxy.
- The JIRA 4.2 documentation is being released with Beta 2. We would be very grateful for your feedback — please leave comments on individual pages.

Note to developers: JIRA 4.2 includes the Atlassian Plugin Framework version 2.5, and an alpha release of the JIRA REST API. We are very keen to hear your feedback on the REST API — please try it out and add your comments to JRA-22139.

Highlights of JIRA 4.2 :

- Dialogs for Common Actions and Workflow Operations
- Keyboard Shortcuts and 'Operations Dialog'
- Issue Labelling
- Improvements to the 'View Issue' Screen
- Auto-complete for Versions and Components
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- 'Filter' Gadget creation via the Issue Navigator
- User Avatars and Hover Profile
- New JQL Functions
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.2 Beta 2
Download Beta

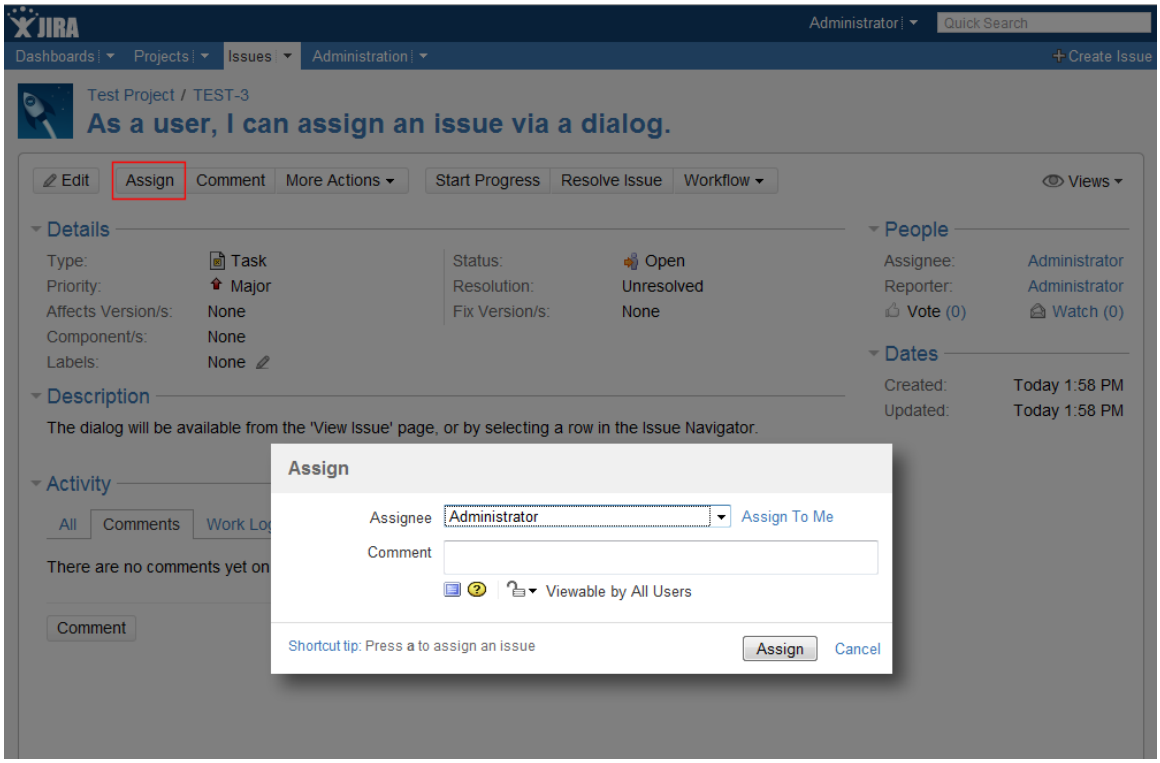
Upgrading to JIRA 4.2 Beta 2

JIRA Beta releases are available here. Before upgrading, please refer to the JIRA 4.2 Upgrade Guide.

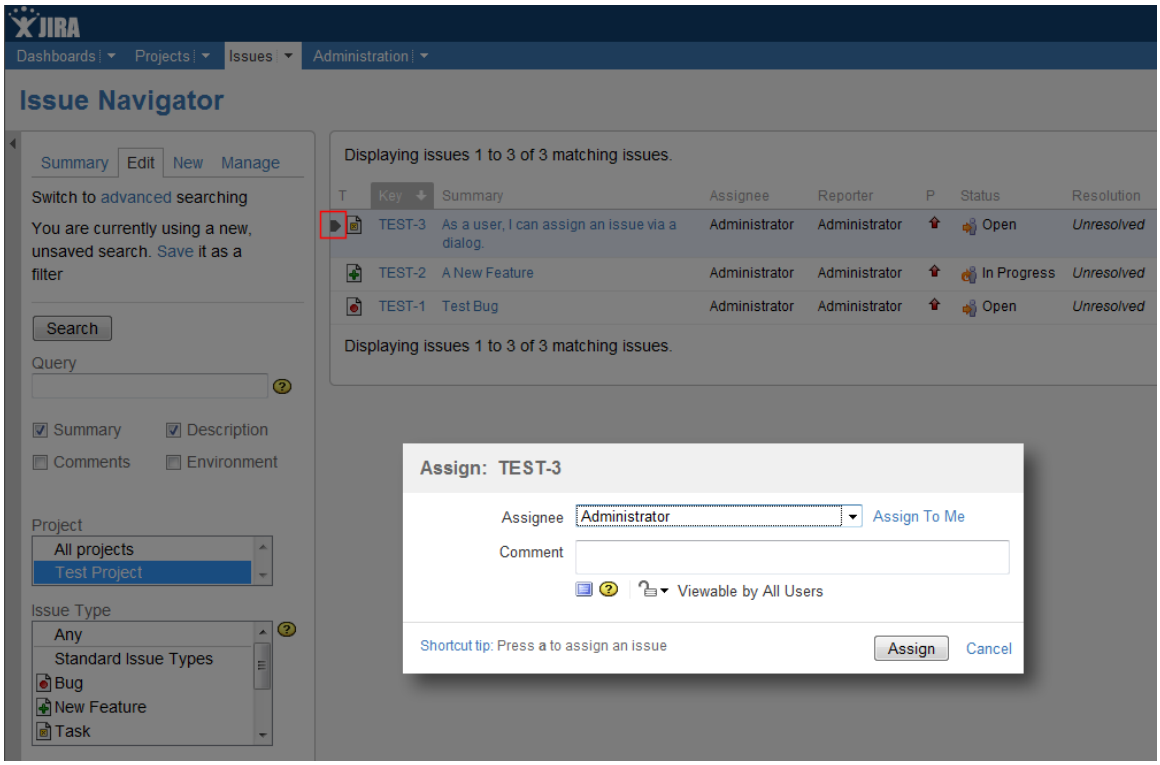
Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

For faster edits to an issue, pop-up dialogs have replaced screen changes for common actions and workflow operations.



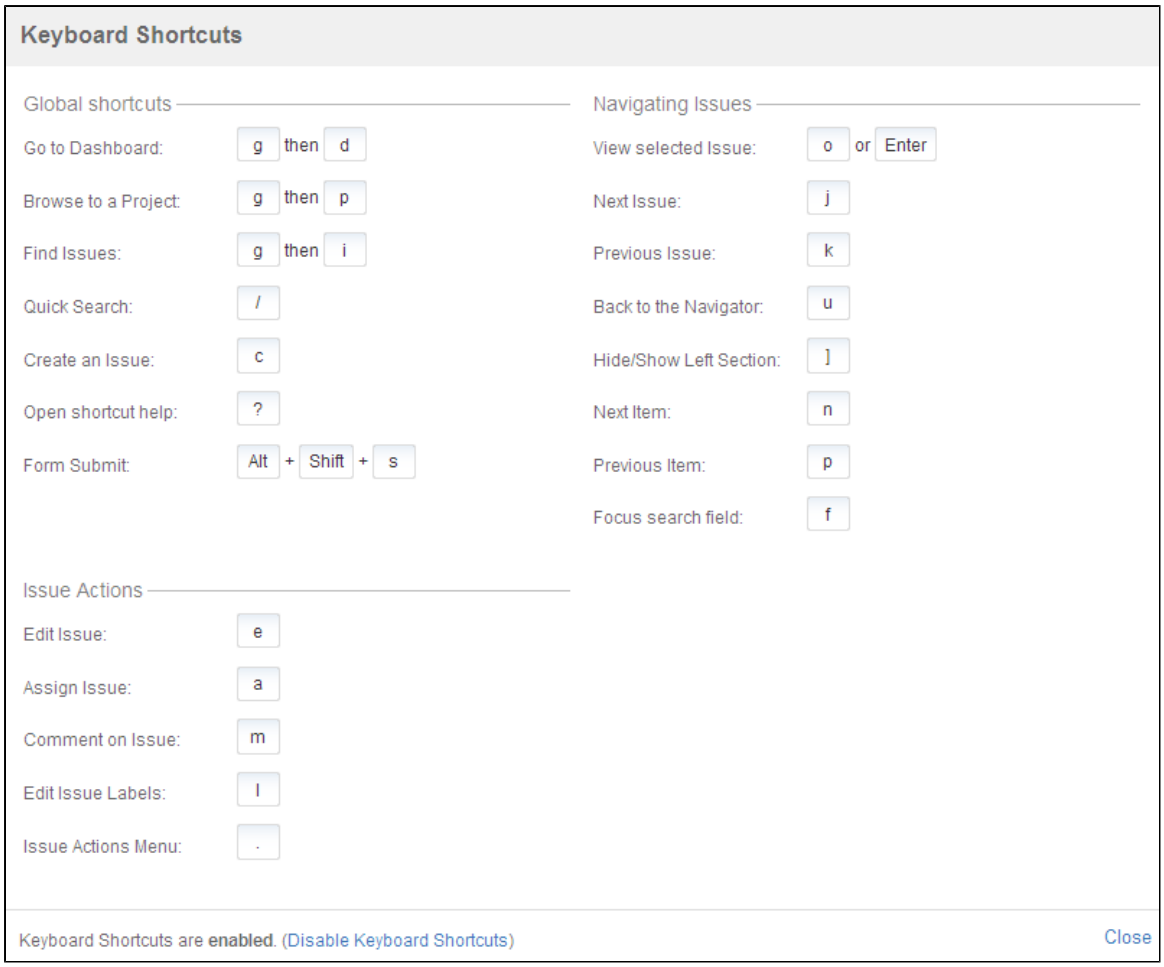
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



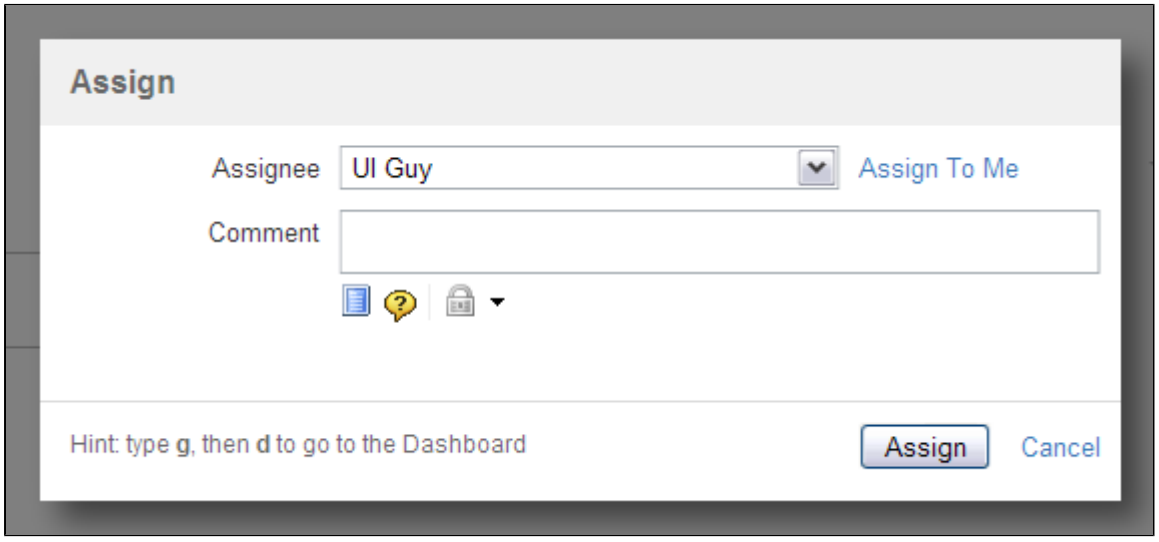
^Top

Keyboard Shortcuts and 'Operations Dialog'

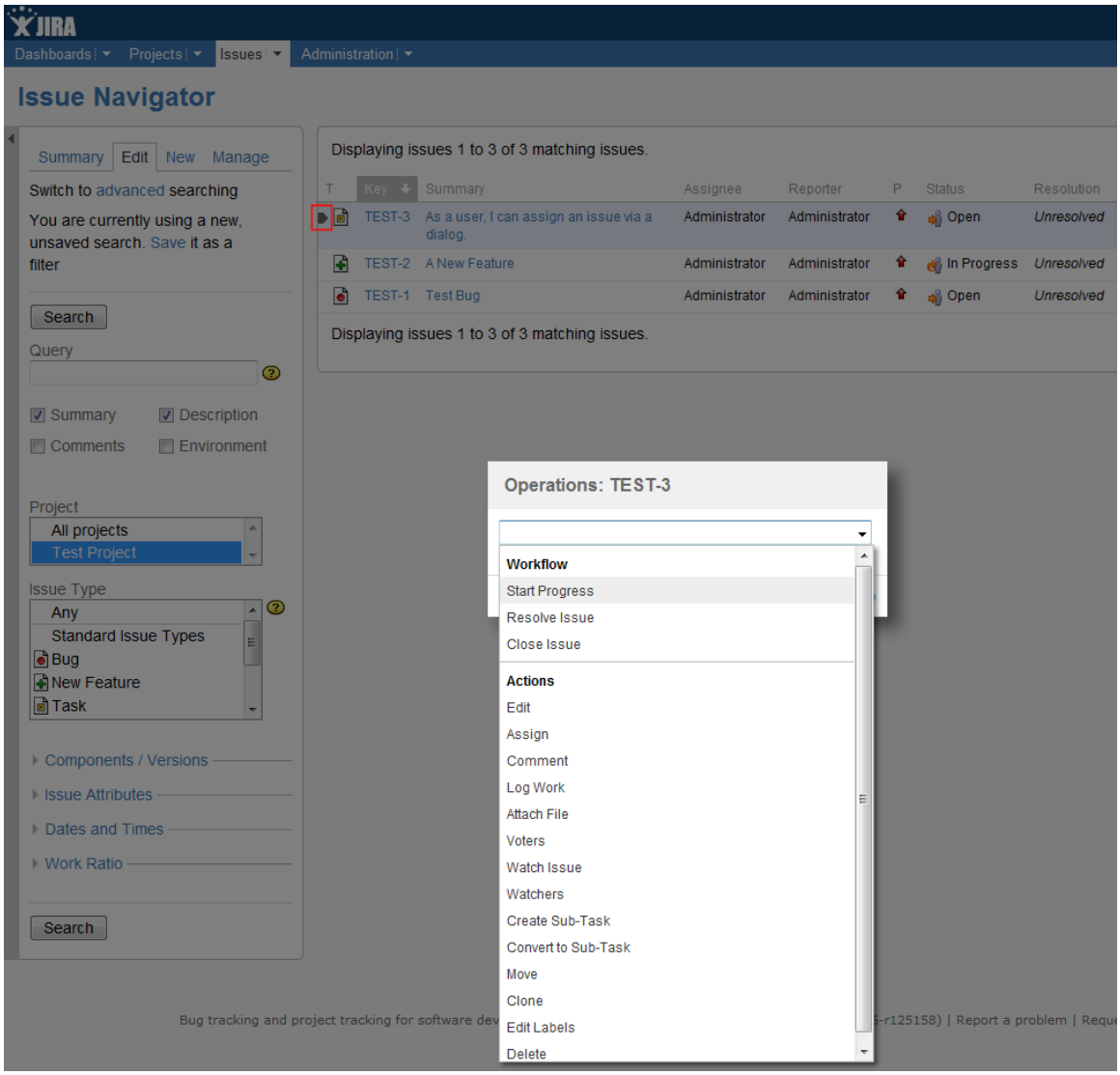
JIRA users can now do more without a mouse — perfect for power users:



Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly:



The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.

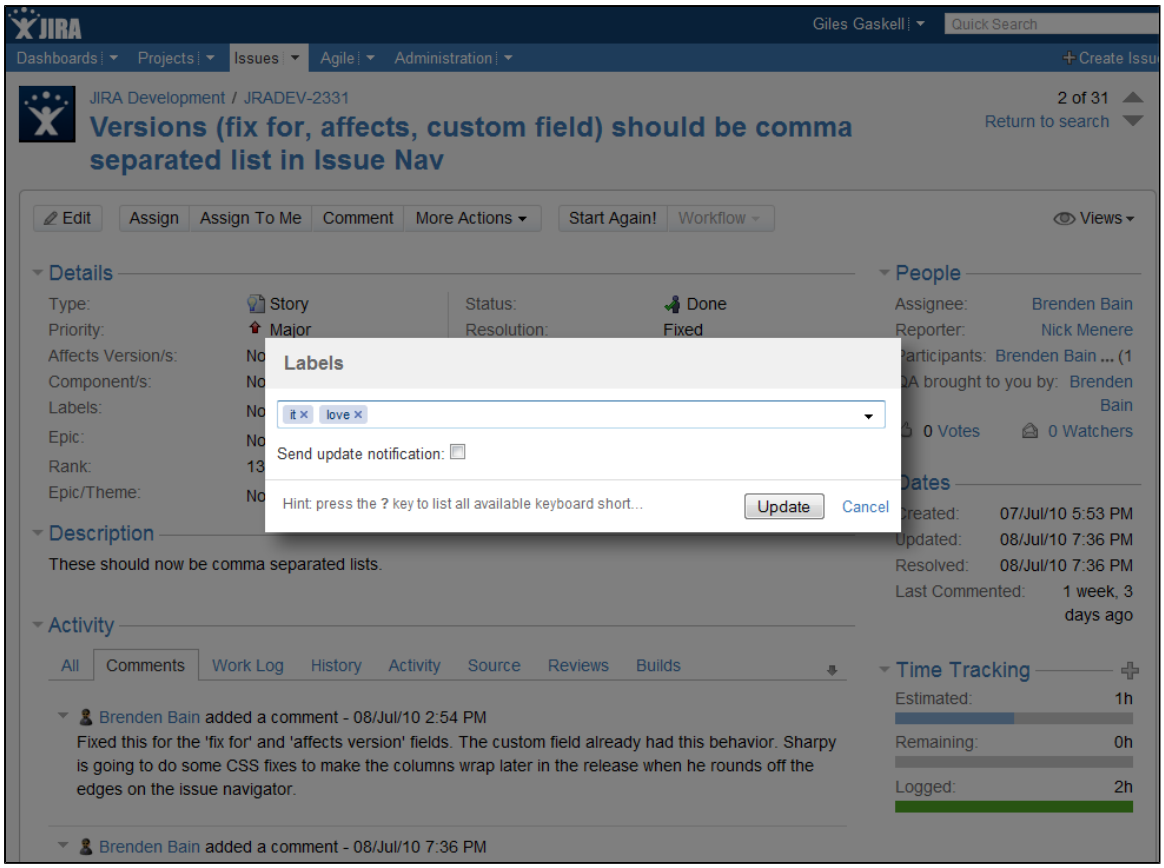


See Using Keyboard Shortcuts for more information.

^Top

Issue Labelling

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.



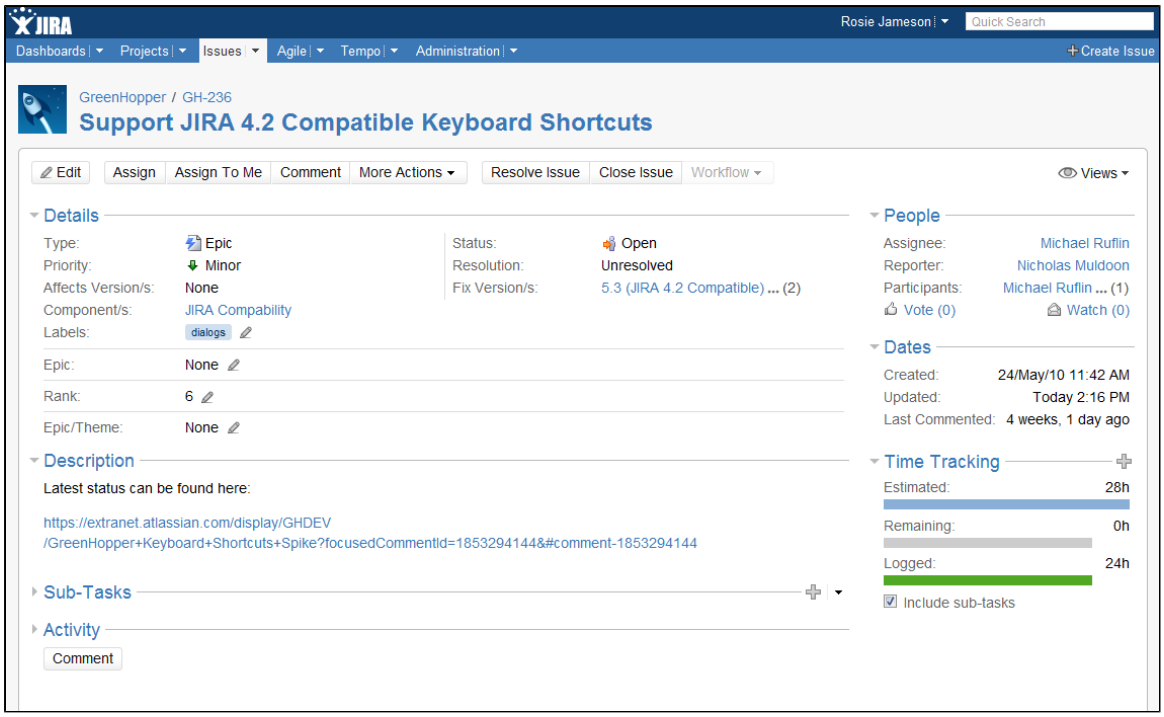
See Labelling an Issue for more information.

^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the 'View Issue' screen even more. Some improvements include:

- Tighter spacing between lines to reduce space between elements
- Stronger section headings
- The 'Description' field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



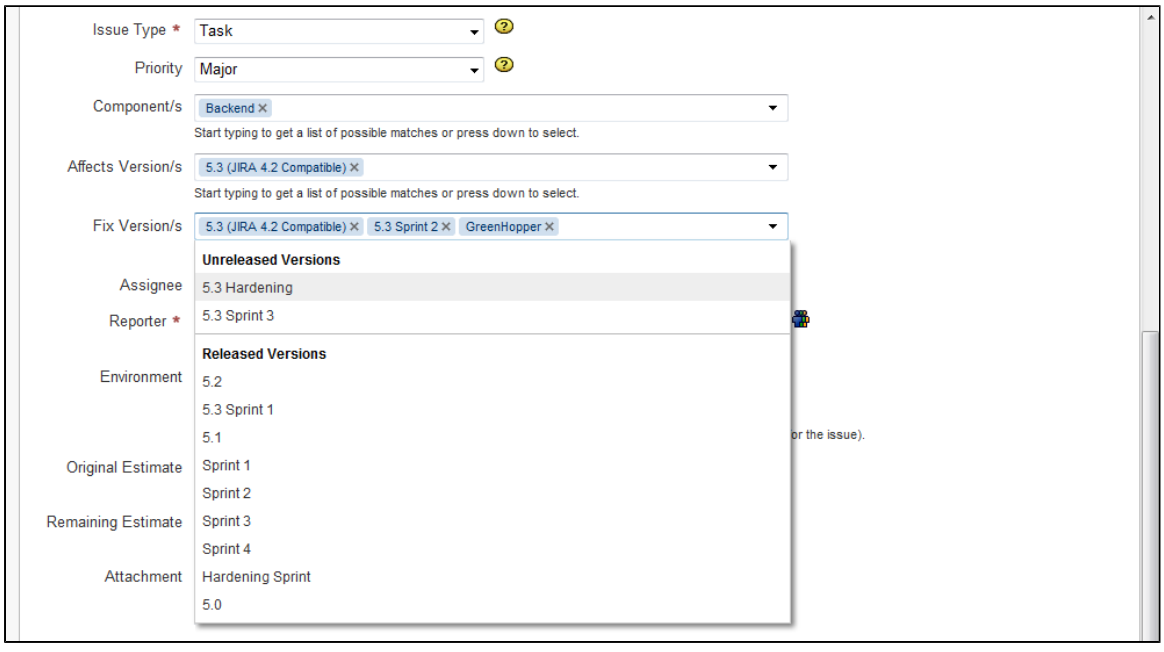
Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the '**Comments**' section.

^Top

Auto-complete for Versions and Components

Editing the **Component**, **Affects Version(s)** and **Fix Version(s)** is now quicker and easier. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

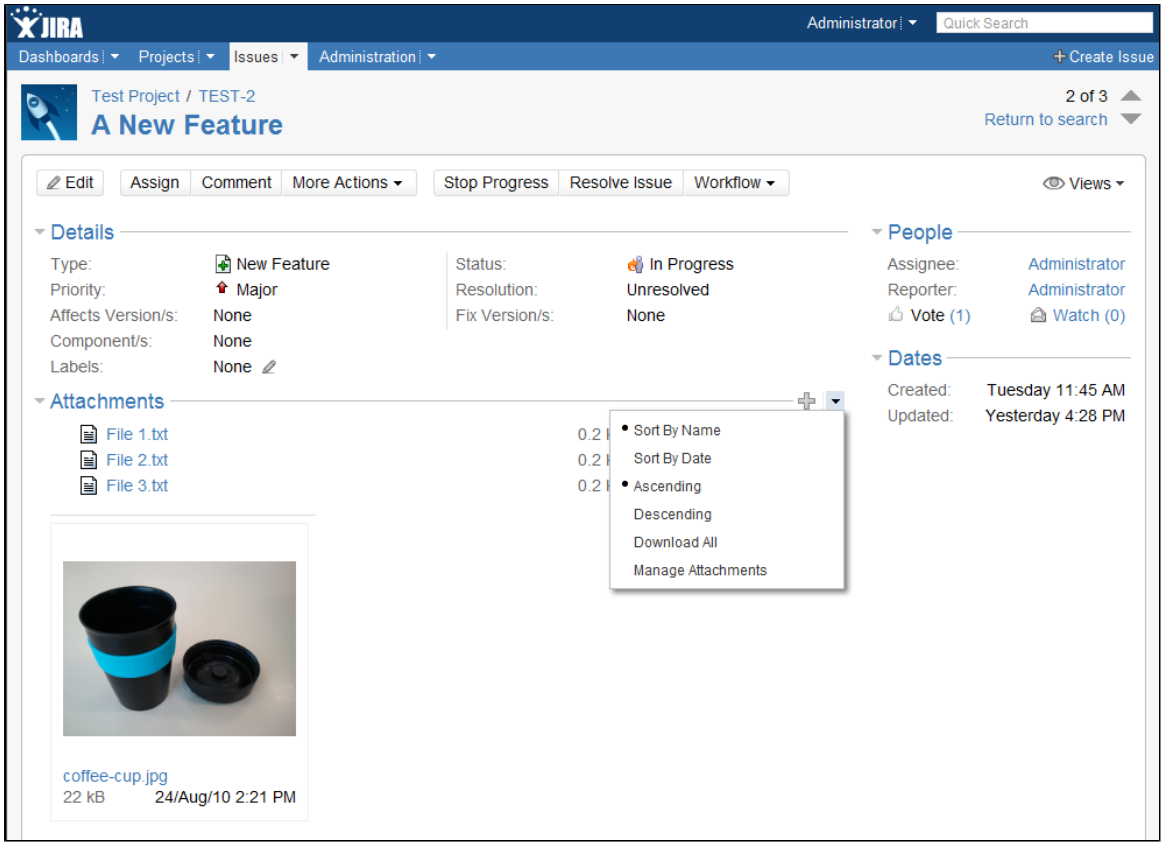
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

Attachments can now be sorted by date or by name, in ascending or descending order.

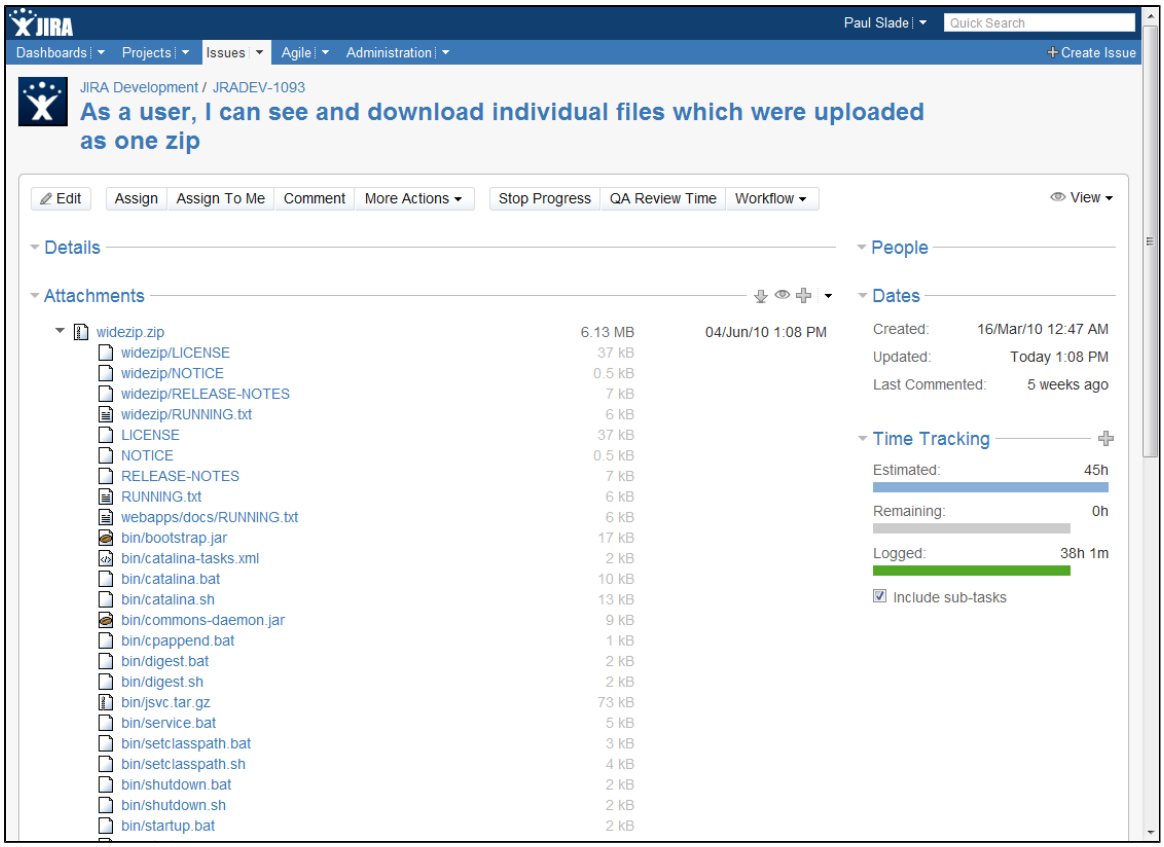


See Attaching a File for more information.

^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



See Accessing ZIP-format File Contents for more information.

^Top

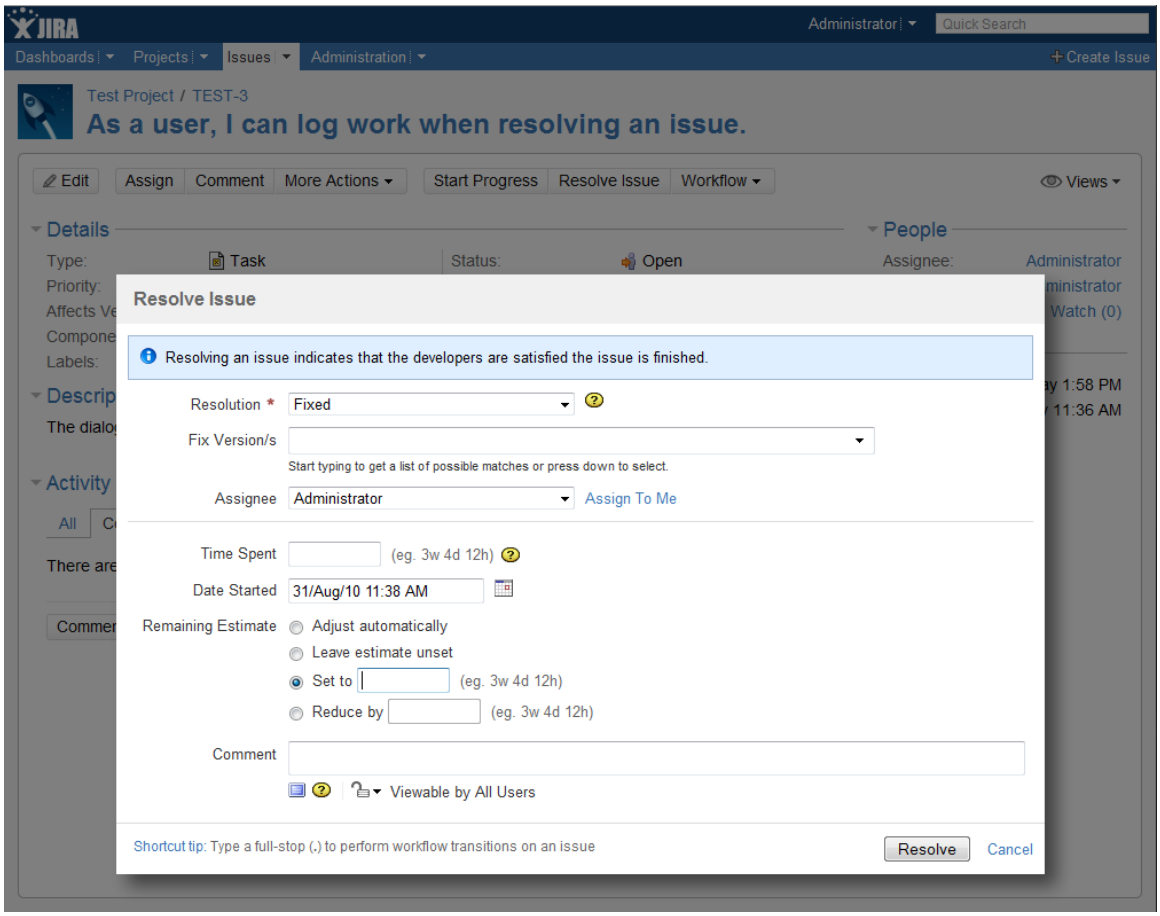
'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).

Lastly you can add the 'Log Work' field to any screen in JIRA (not shown here).



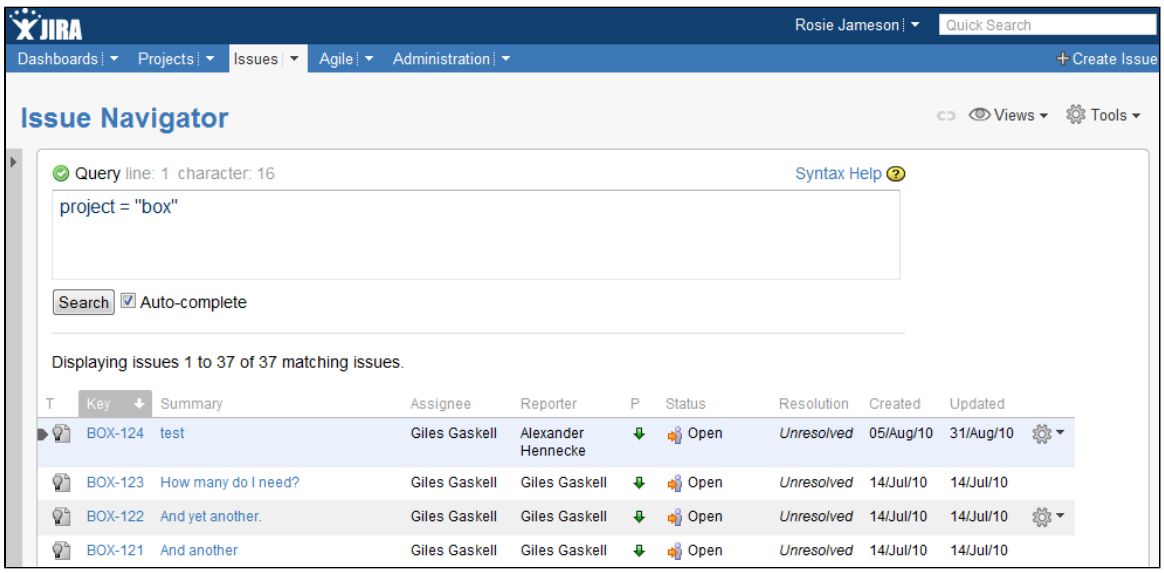
See Logging Work on an Issue for more information.

^Top

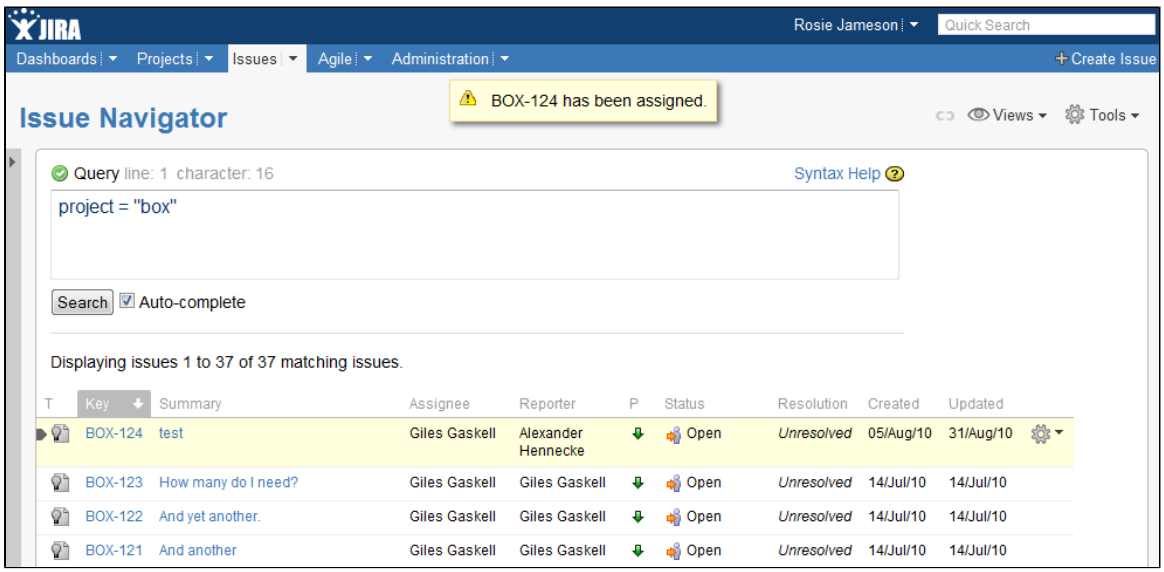
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.

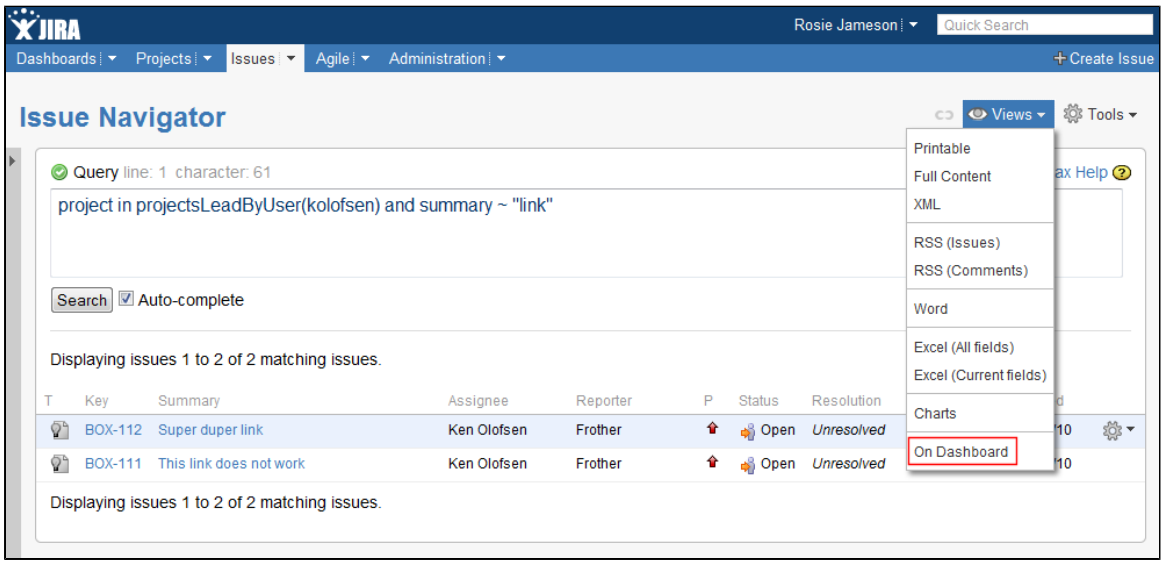


^Top

10

'Filter' Gadget creation via the Issue Navigator

When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.

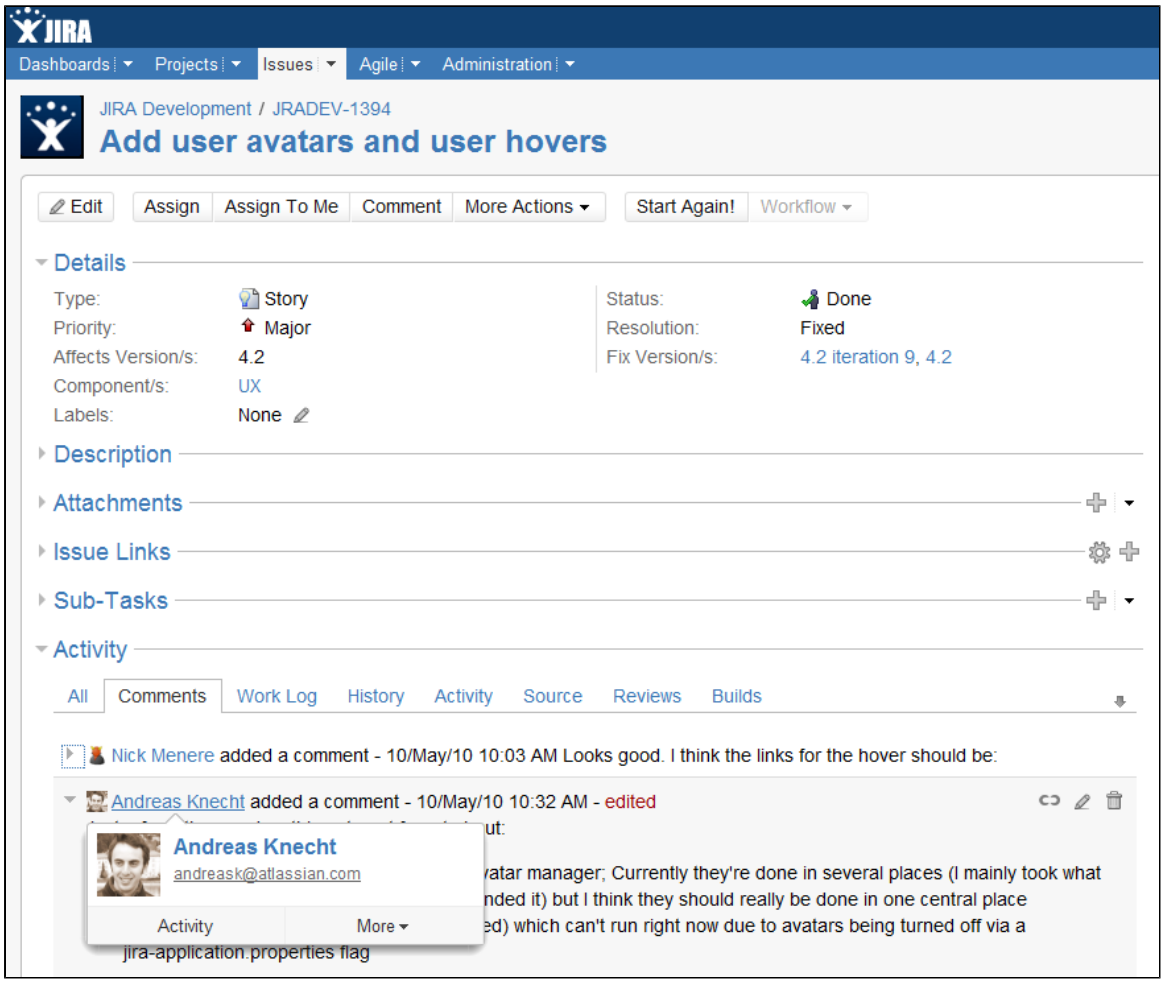


^Top

11

User Avatars and Hover Profile

JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.

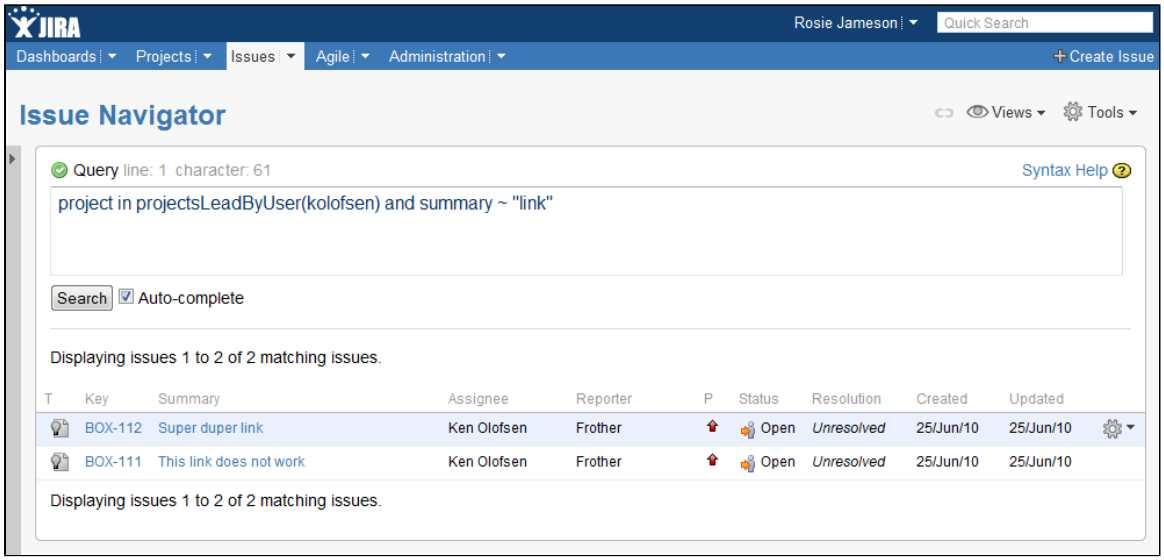


See Adding a User Avatar and Using Hover Profile for more information.

^Top

New JQL Functions

Additional JQL functions for projectsLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



See Advanced Searching for more information.

^Top

Other Enhancements and Fixes

- If you use Firefox or IE, you can now search JIRA issues from the convenience of your browser's search box. Just add your JIRA site as a search engine/provider, via the dropdown menu next to the browser's search box. This is because JIRA now supports the autodiscovery part of the OpenSearch standard.
- Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.2 Beta 1 Release Notes

1 September 2010

JIRA 4.2 Beta 1 is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

The Atlassian team is proud to bring you the JIRA 4.2 Beta 1 release. As always, we would love to hear your feedback and comments about this release. To keep the discussion centralised, please provide your feedback as a comment on JRA-22093.

JIRA 4.2 gives you a few extra minutes of precious time every day, by providing the ability to triage issues directly from the Issue Navigator without you having to open each issue. For the mouse-averse, the new 'Operations Dialog' box provides access to all menu options via the keyboard.

To help speed up your ability to work on issues, we've also made common actions and workflow operations accessible from convenient dialog boxes.

Time-tracking has become much more flexible: you can now edit the Original Estimate and set the Remaining Estimate to zero when resolving an issue. You can also log work via workflow 'transition' (or any other) screens — now accessible via speedy dialog boxes.

The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users and administrators — and everyone else who likes to label their JIRA issues.

The FishEye plugin has been updated to version 3.0.

GreenHopper 5.3 Beta 1 is also available for download, and is compatible with JIRA 4.2 Beta 1.

Note to developers: JIRA 4.2 includes the Atlassian Plugin Framework version 2.5, and an alpha release of the JIRA REST API. We are very keen to hear your feedback on the REST API — please try it out and add your comments to JRA-22139.

Highlights of JIRA 4.2 :

- Dialogs for Common Actions and Workflow Operations

- Keyboard Shortcuts and 'Operations Dialog'
- Issue Labelling
- Improvements to the 'View Issue' Screen
- Auto-complete for Versions and Components
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- 'Filter' Gadget creation via the Issue Navigator
- User Avatars and User Hover
- New JQL Functions
- Other Enhancements and Fixes

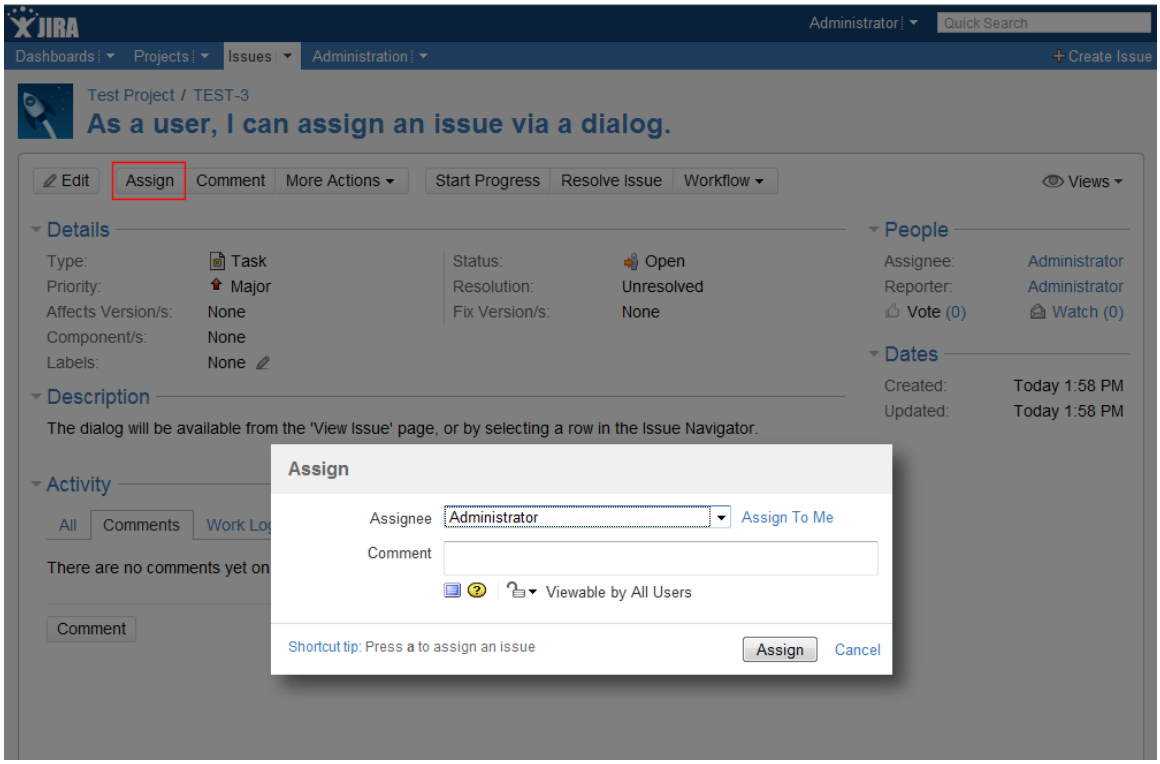
Thank you for your interest in JIRA 4.2 Beta 1
Download Beta

> **Upgrading to JIRA 4.2 Beta 1**
>
> JIRA Beta releases are available here. Before upgrading, please refer to the JIRA 4.2 Upgrade Guide.

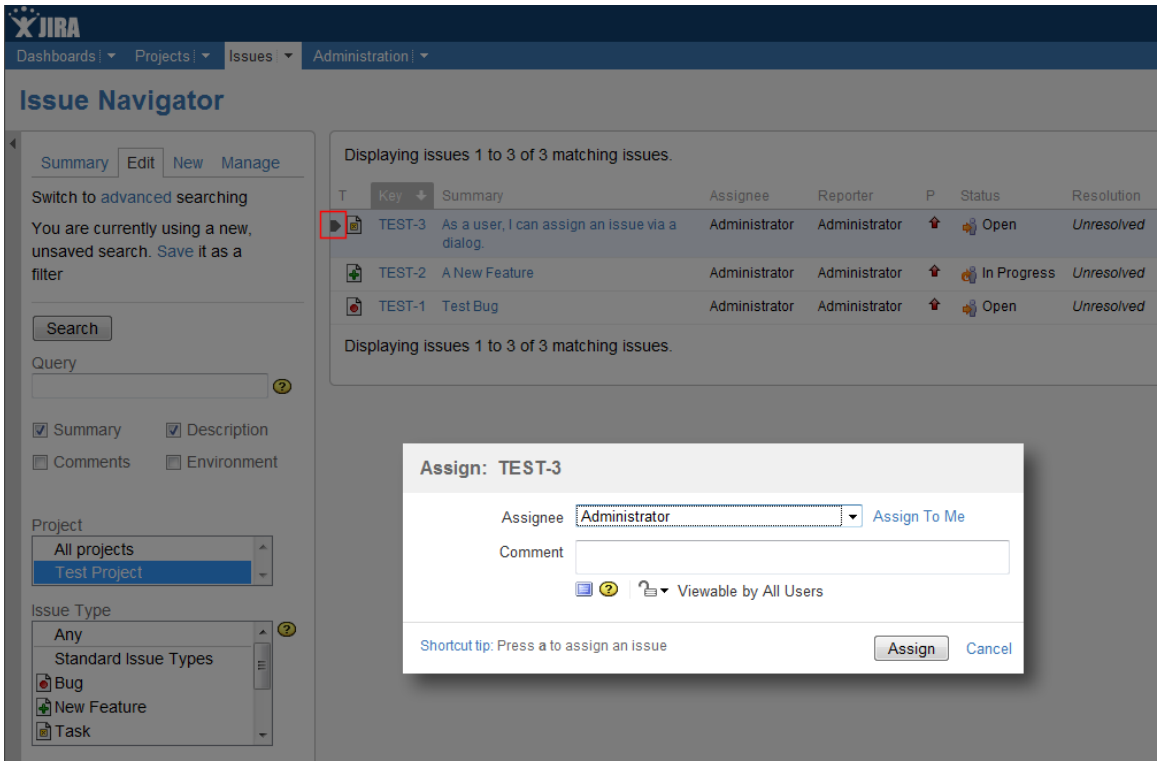
Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

For faster edits to an issue, pop-up dialogs have replaced screen changes for common actions and workflow operations.



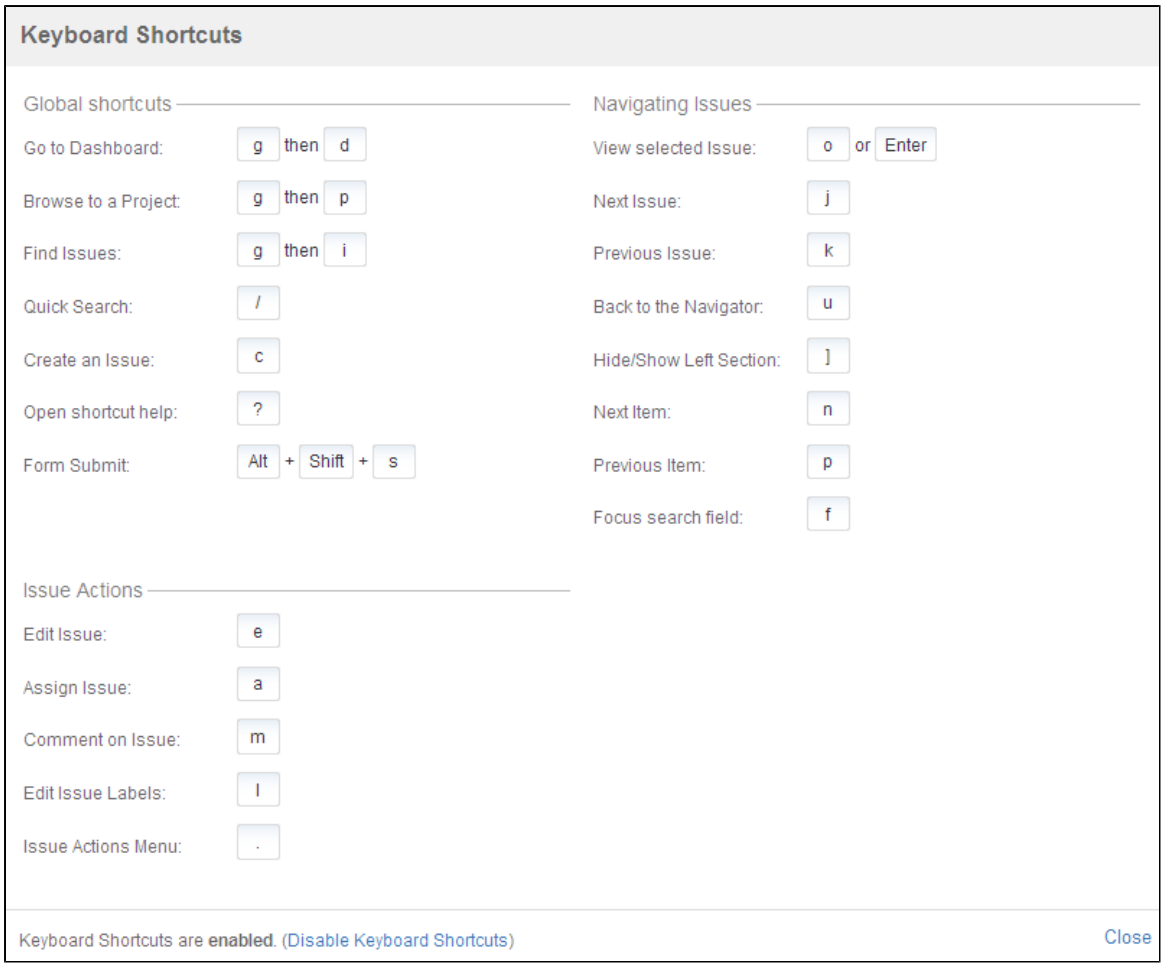
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



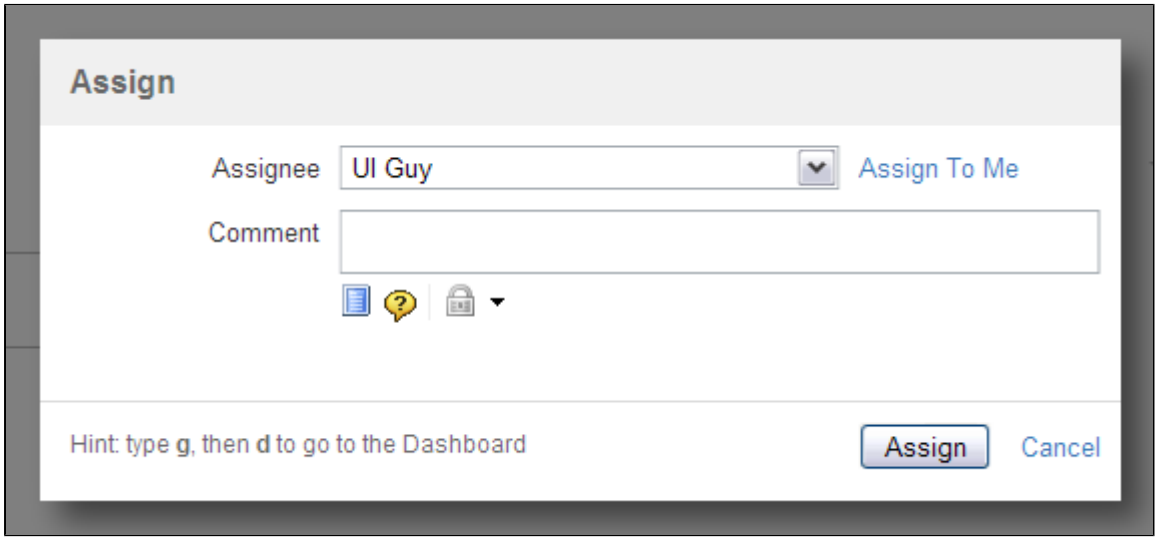
^Top

Keyboard Shortcuts and 'Operations Dialog'

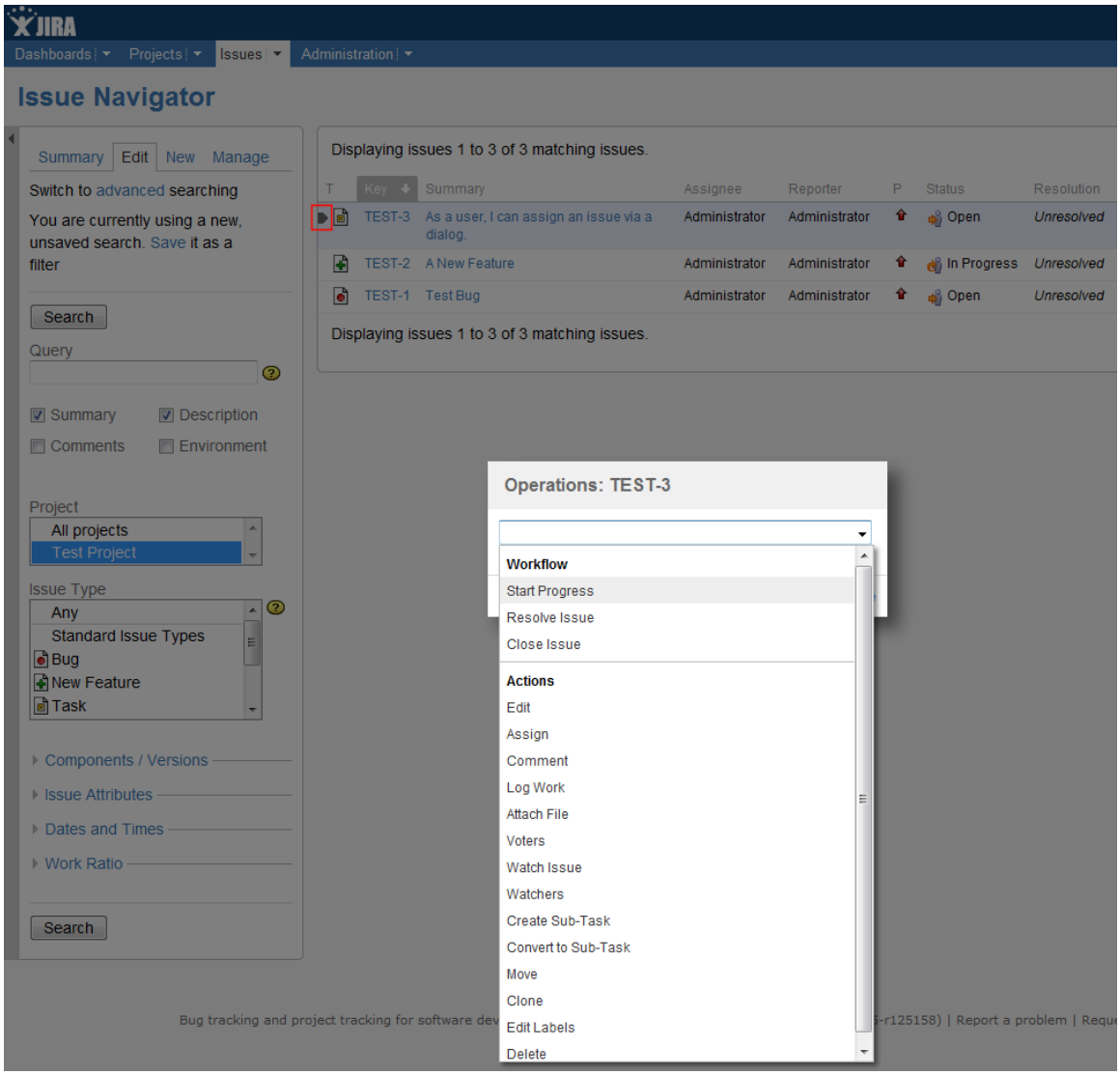
JIRA users can now do more without a mouse — perfect for power users:



Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly:



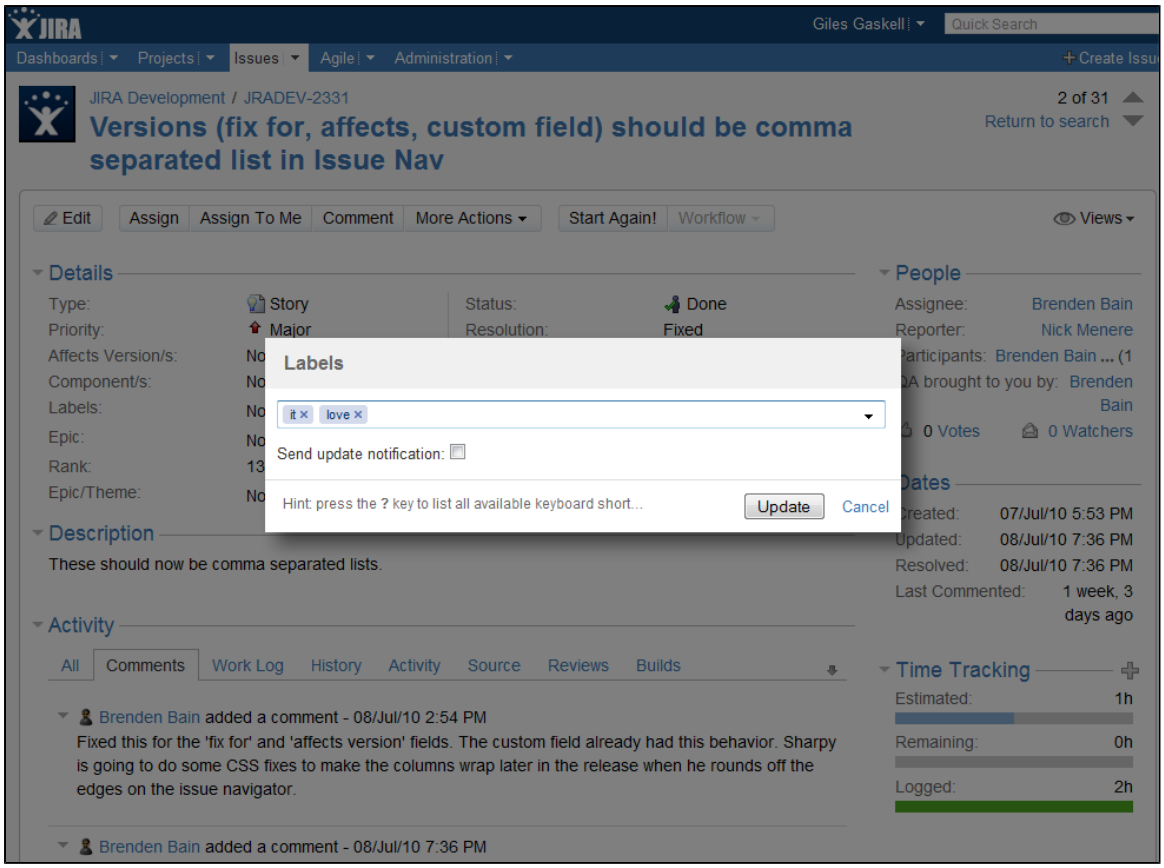
The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.



^Top

Issue Labelling

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.

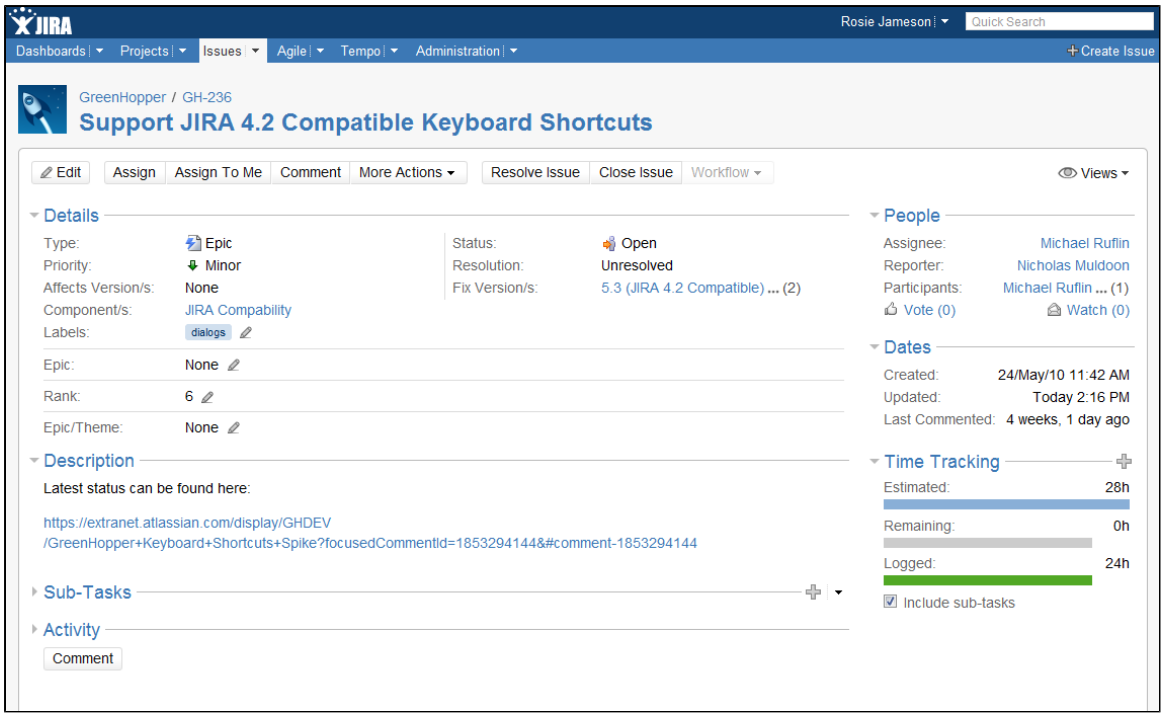


^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the 'View Issue' screen even more. Some improvements include:

- Tighter spacing between lines to reduce space between elements
- Stronger section headings
- The 'Description' field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



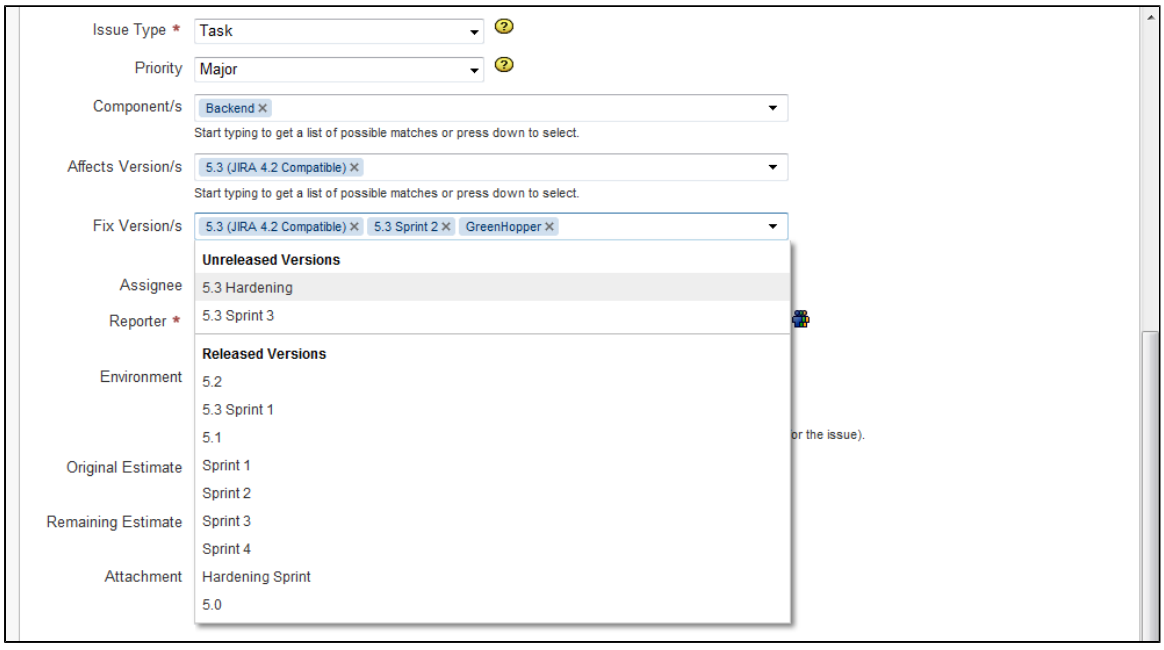
Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the '**Comments**' section.

^Top

Auto-complete for Versions and Components

Editing the **Component**, **Affects Version(s)** and **Fix Version(s)** is now quicker and easier. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

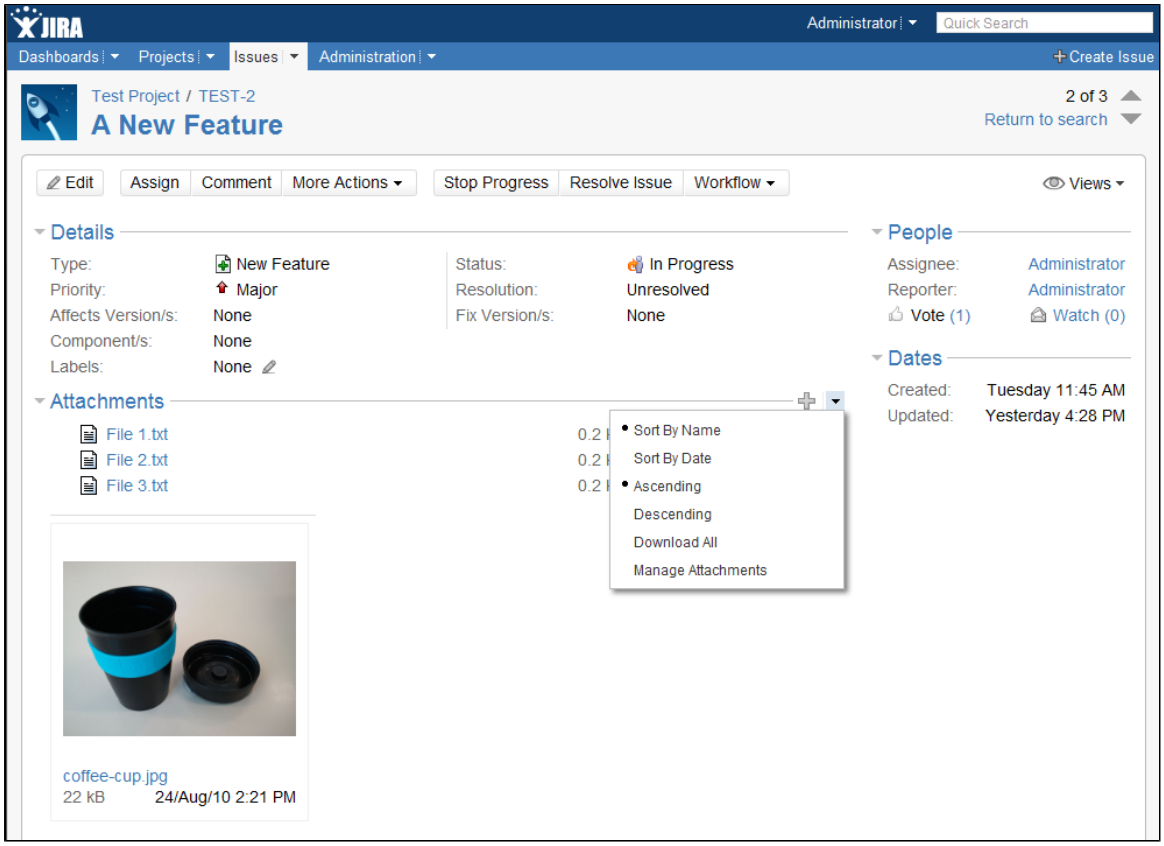
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

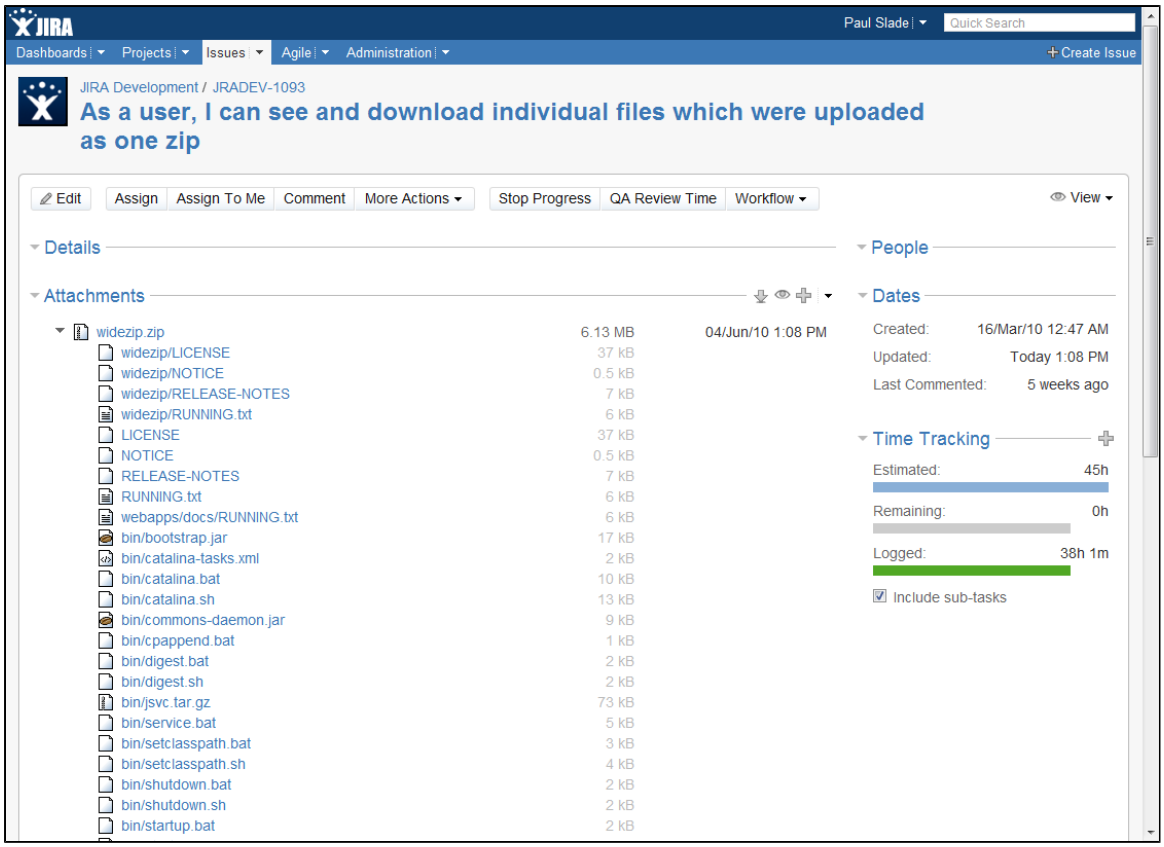
Attachments can now be sorted by date or by name, in ascending or descending order.



^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



^Top

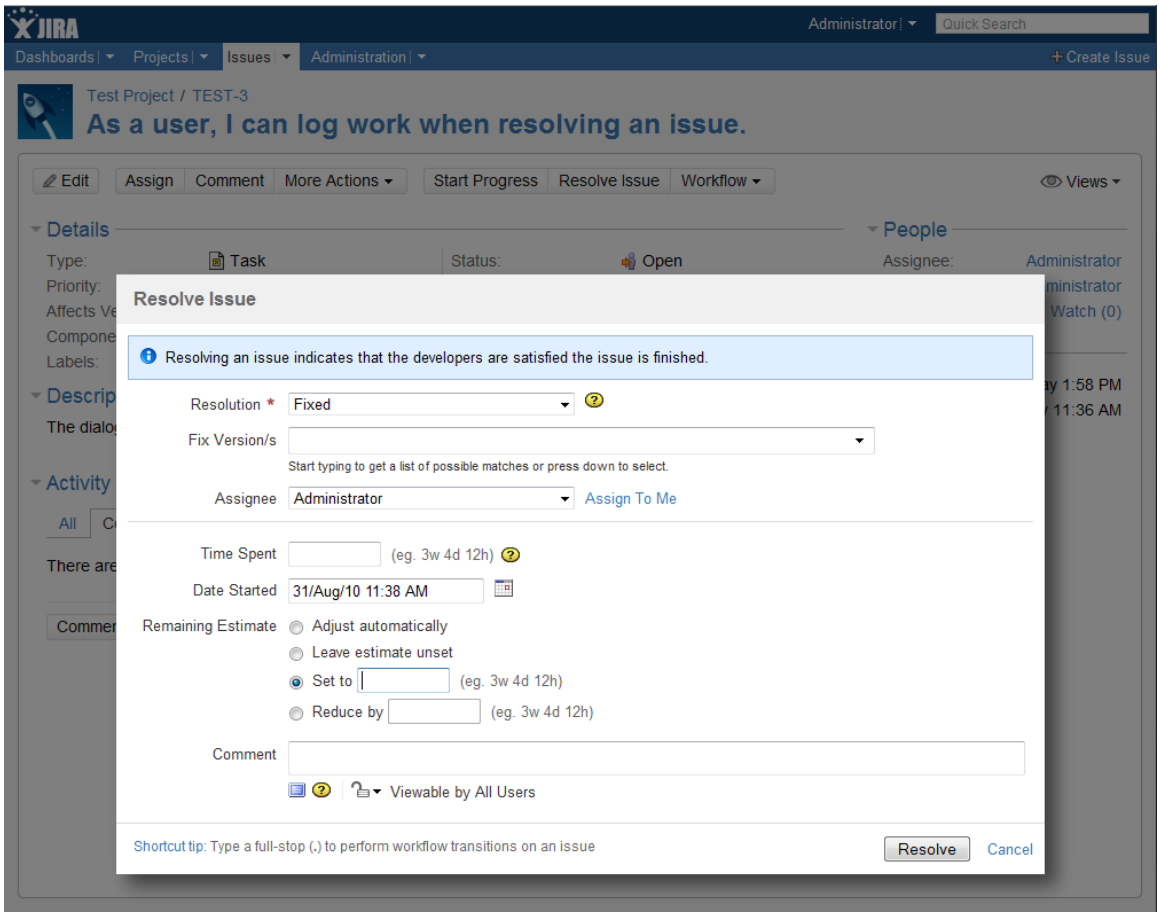
'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).

Lastly you can add the 'Log Work' field to any screen in JIRA (not shown here).



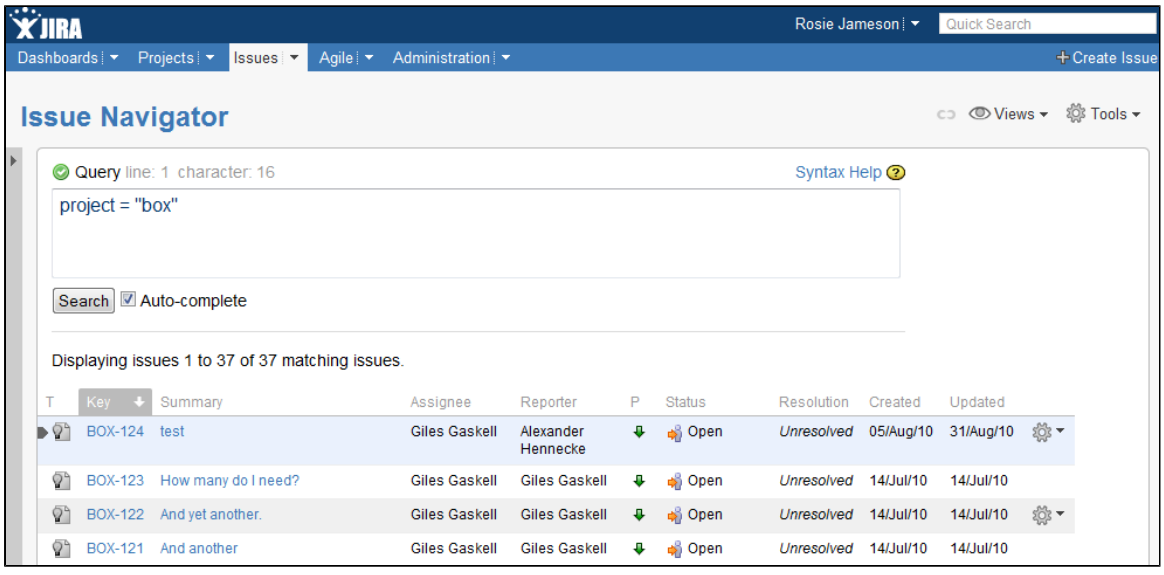
^Top

9

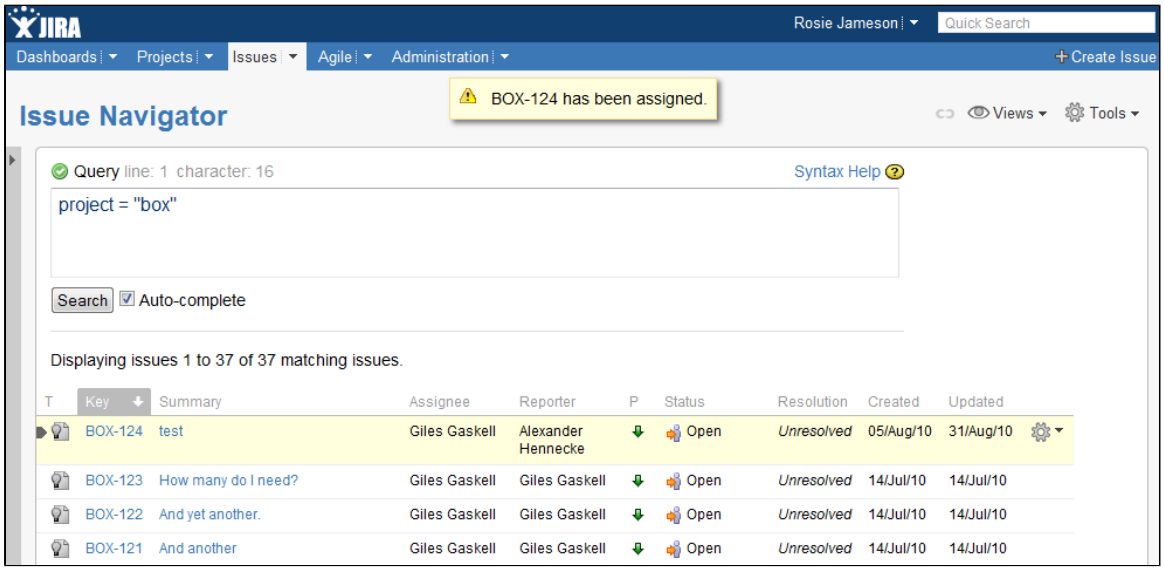
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



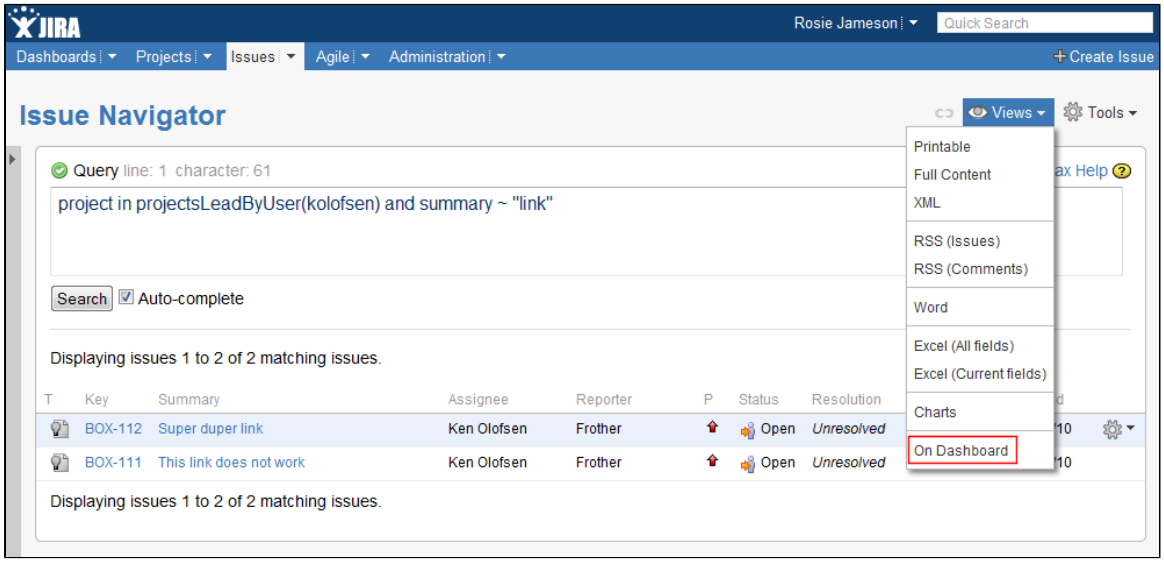
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

'Filter' Gadget creation via the Issue Navigator

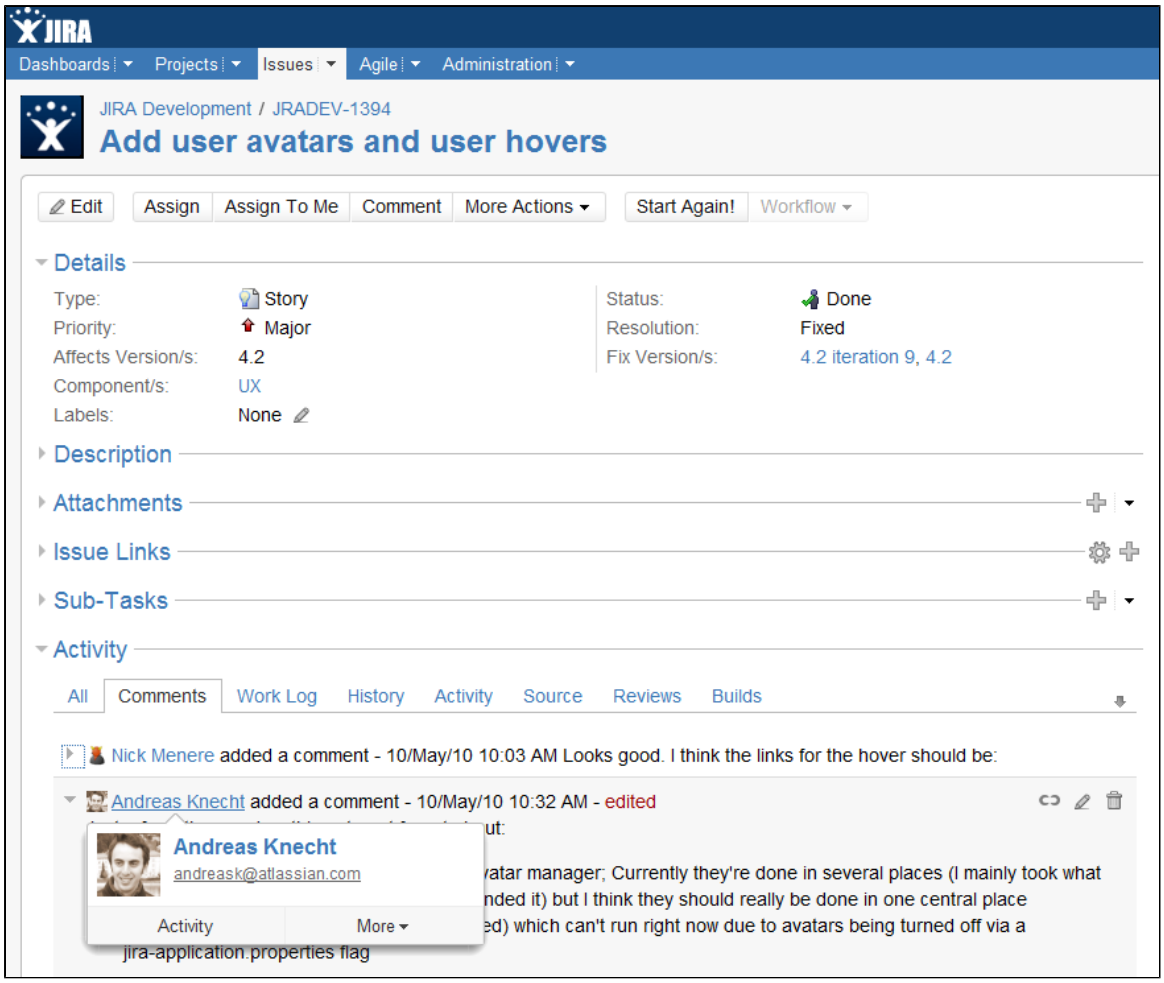
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



^Top

User Avatars and User Hover

JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.

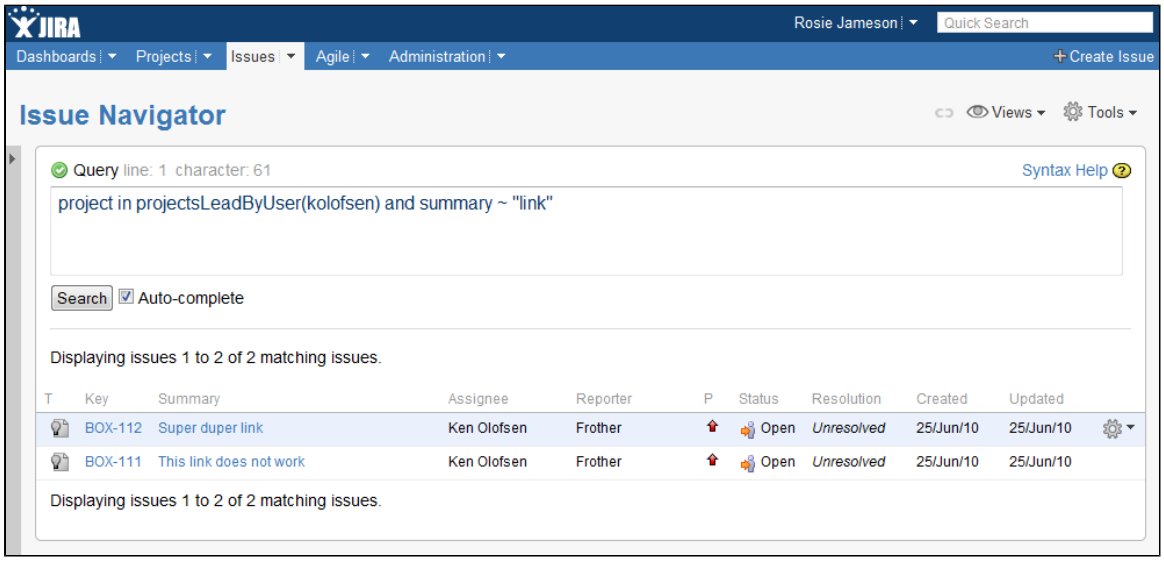


^Top

12

New JQL Functions

Additional JQL functions for projectsLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



^Top

Other Enhancements and Fixes

- If you use Firefox or IE, you can now search JIRA issues from the convenience of your browser's search box. Just add your JIRA site as a search engine/provider, via the dropdown menu next to the browser's search box. This is because JIRA now supports the autodiscovery part of the OpenSearch standard.
- Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.0 RC1 Release Notes

September 22, 2009

JIRA 4.0 Release Candidate 1 (RC1) is a public development release leading up to **JIRA 4.0**. For all production use and testing of JIRA, please use the latest official release.

A **release candidate** is a preliminary release leading up to the official release of a JIRA version. Release candidates are a snapshot of our work in progress and provide an advance preview of new features to the general public. JIRA plugin developers can also use release candidates to test and fix their plugins in advance of an official release.

It is not possible to upgrade JIRA 4.0 Beta 1, 2, 3, 4 or 5 data to RC1. The data needs to come from an already released JIRA version (for example, JIRA 3.13.5).

The **only** plugins that are compatible with JIRA 4.0 RC1 are the latest JIRA Toolkit and the GreenHopper Beta 5 plugin. Do not install any other plugins.

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

Highlights of JIRA 4.0 RC1:

- Advanced Searching
- Dashboard Gadgets
- Activity Streams
- New-look "Browse Project"
- Charting Now Comes Standard
- New-look Header
- Issue Actions in the Issue Navigator
- Project Icons
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- Plus over 900 other fixes and improvements

Thank you for your interest in JIRA 4.0 RC1
Download JIRA 4.0 RC1

Installing/Upgrading to JIRA 4.0
JIRA 4.0 RC1 can be downloaded here. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide. You will also need to go to my.atlassian.com and generate a "**JIRA 4 Evaluation**" license, if you haven't done so previously, as any existing 3.x license files will not work with 4.0 RC1.

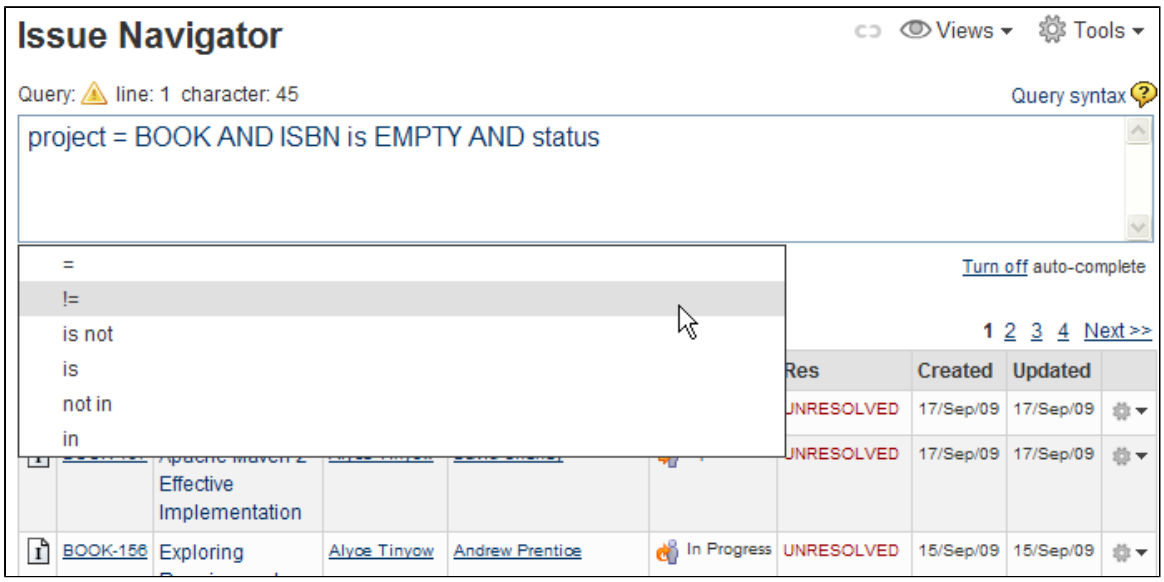
Highlights of JIRA 4.0 RC1

Advanced Searching

The power of search can never be understated, especially in a system like JIRA that sits at the centre of your development team.

JIRA Query Language (or JQL) brings search to whole new level!

JQL is a structured query language that provides support for logical operations, including AND, OR, NOT, NULL, EMPTY — even on custom fields:



Using JQL is simple even for those who don't know what "DBA" means. Just start typing and the auto-complete feature starts to suggest fields, operators and values for you to define your query.

You can now create more advanced filters such that you can stay up to date using RSS feeds & e-mail subscriptions, as well as see detailed statistics and charts, on issues that you are actually interested in.

^Top

Dashboard Gadgets

Whether you are tracking bugs or managing your entire development process, JIRA dashboards let you stay up to date on what matters most.

The new-look JIRA dashboard not only looks awesome, it now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

You can easily customise your dashboard by choosing a different layout, adding more gadgets, dragging the gadgets into different positions and changing the look of individual gadgets.

What's happened to your favourite JIRA portlets? Don't worry, every portlet that previously shipped with JIRA has been converted to a gadget.

If you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.

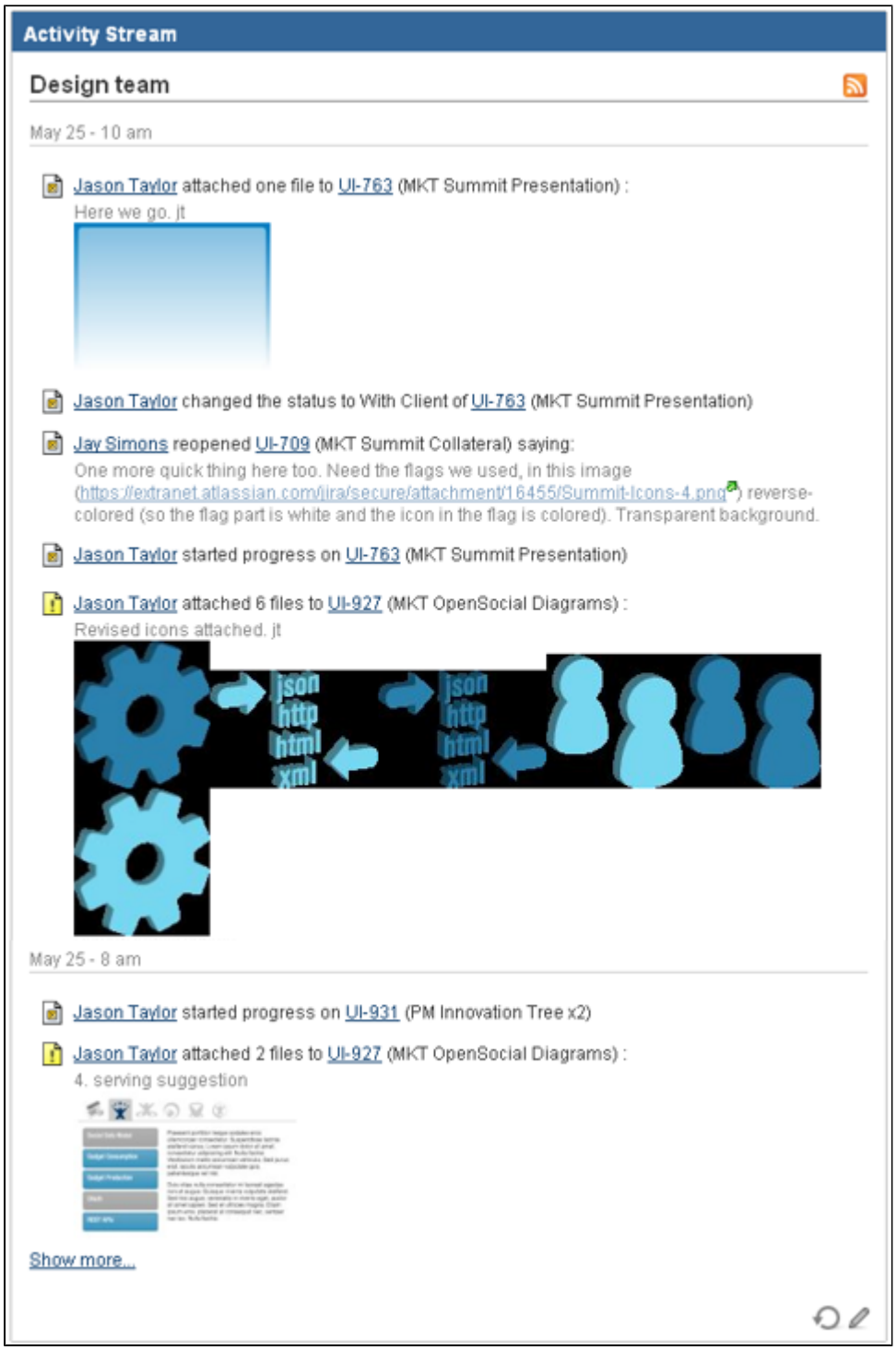
^Top

Activity Streams

The new activity stream allows you to stay up to date with exactly what is going on right this moment, what happened in that last hour or last few days.

Activity streams appear where you need them most — your user profile, project summary and view issue screens. You can even add an activity stream as a gadget on your dashboard.

The activity stream also provides an RSS feed, allowing you to subscribe to very specific RSS feeds of only the information that is most relevant to you.



See the documentation for more details.

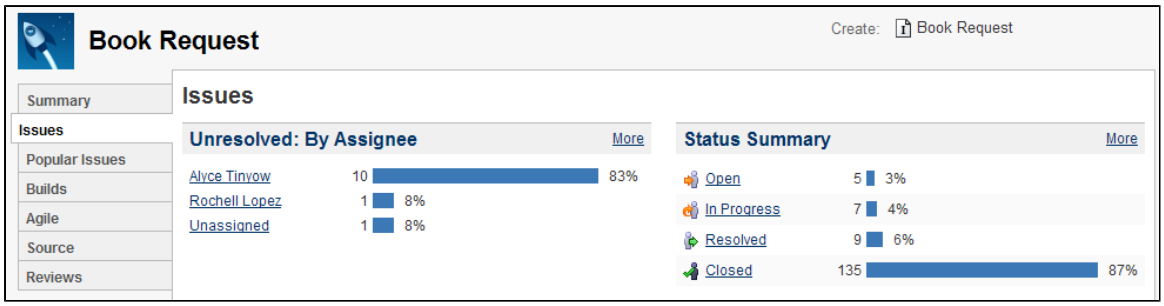
^Top

New-look "Browse Project"

Understanding the status of your projects just got a lot easier with the new browse project UI.

Quickly see what work is complete as well as outstanding. You can then drill down to specific issues you want to see.

Your Bamboo builds, FishEye source information and Crucible code reviews are only a click away, as well.

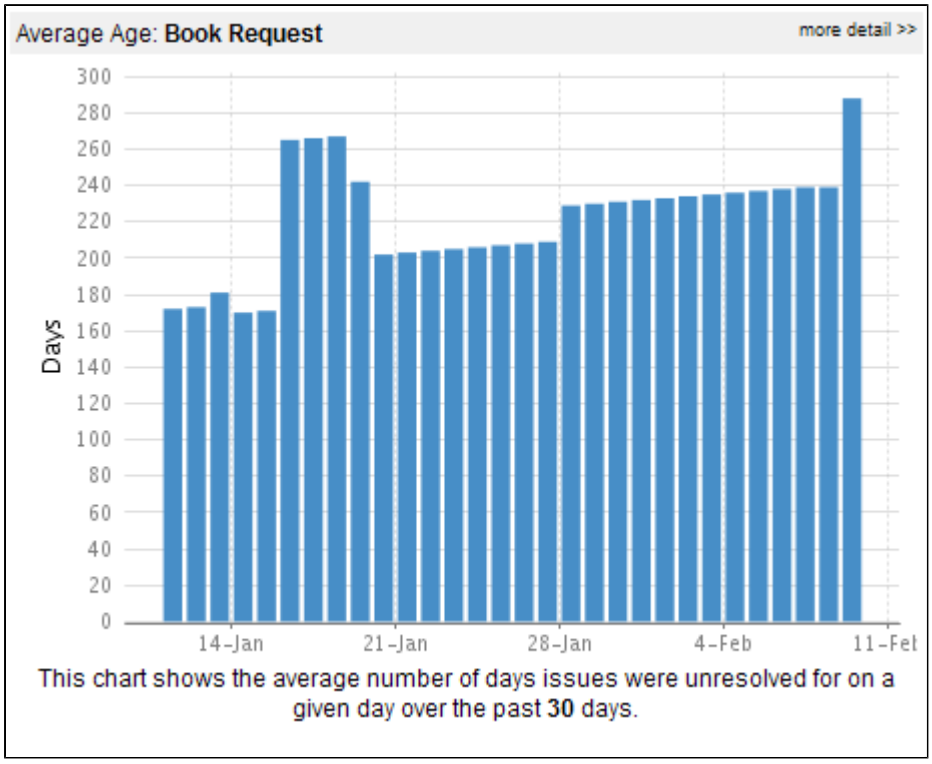


See the documentation for more about browsing projects, versions and components.
^Top

Charting Now Comes Standard

We've built charts into JIRA and given them a visual redesign as well.

- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over a given period of time.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.
- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



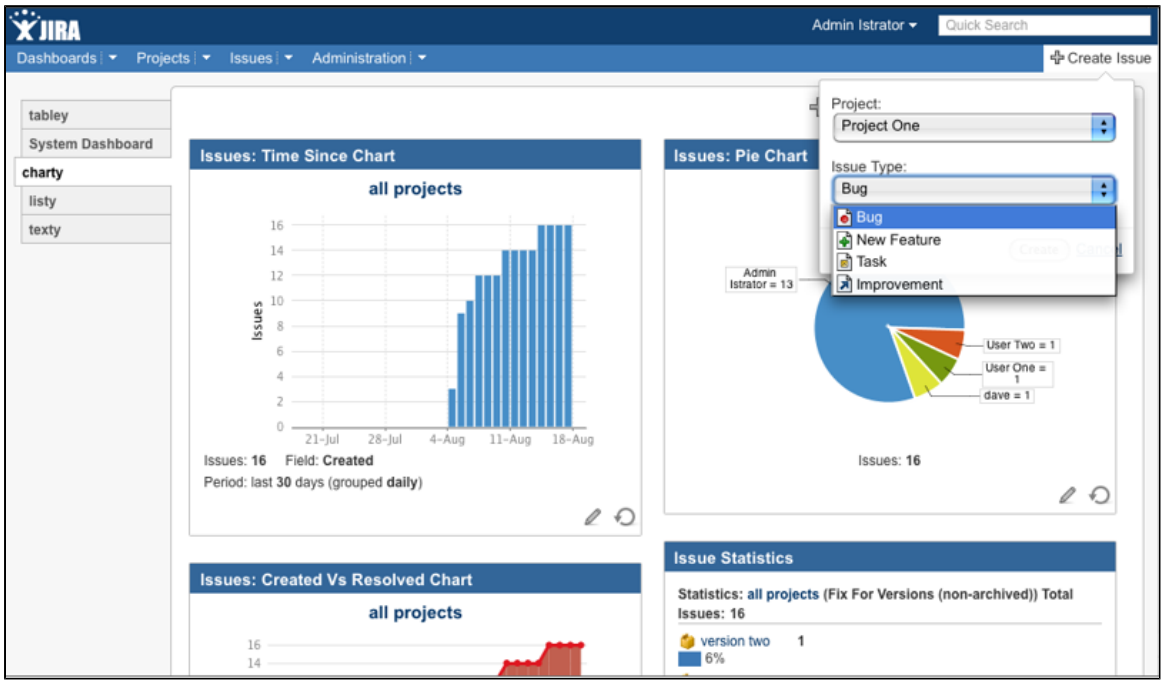
Also, the "Resolution Time" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution time recorded.

^Top

6

New-look Header

The new-look JIRA header gives you quick access to all of the most commonly-used functions. Creating an issue just got even faster!



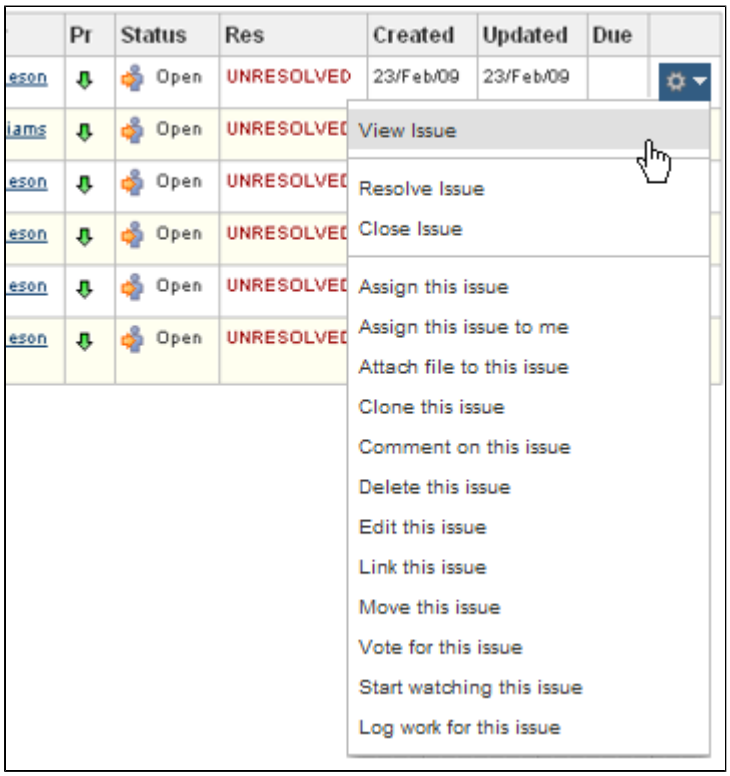
Click to zoom in:

If you prefer keystrokes rather than mouse-clicks, you'll be pleased to know that you can use your keyboard to navigate the new header menus.
^Top

7

Issue Actions in the Issue Navigator

By popular request, issues are now actionable directly from the Issue Navigator:

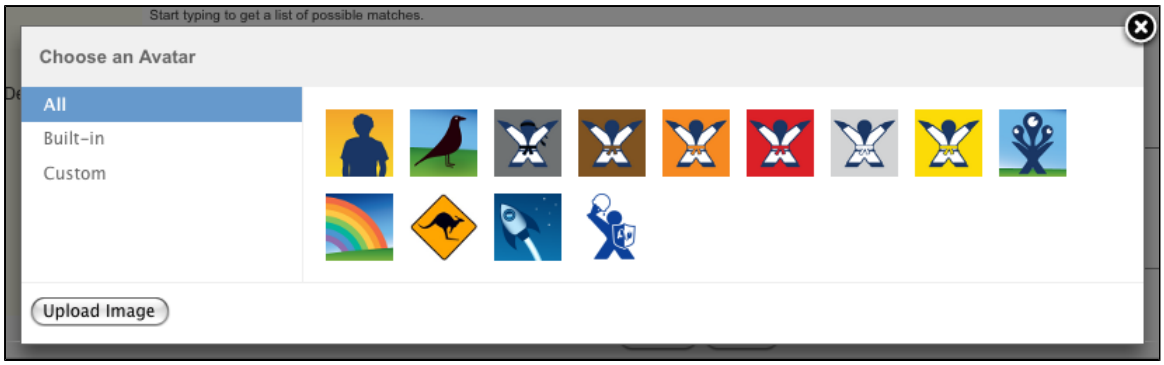


The "Actions" menu is also available for the list of sub-tasks within an issue.
^Top

8

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):



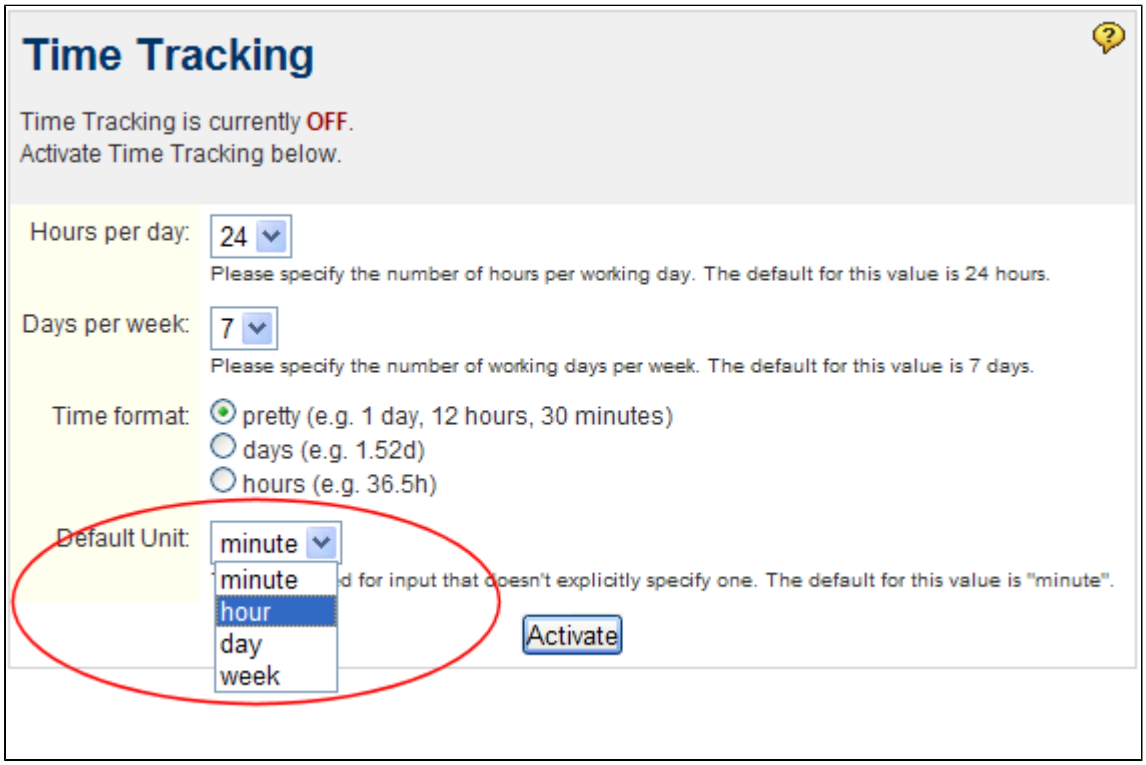
^Top

9

Default Unit for Time Tracking

You can now specify your preferred **Default Unit** (minutes/hours/days/weeks) for your JIRA system. This will be applied whenever users log

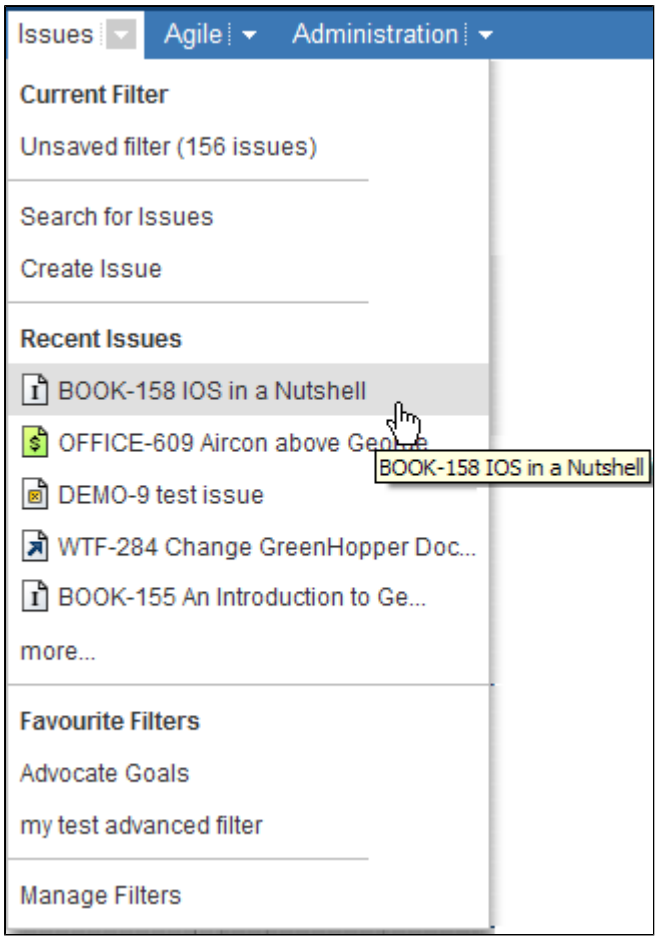
work on an issue without specifying a unit.



^Top

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.



^Top

Engine Room

Beyond the 'Back' Button

When navigating away from a page where you have modified data, you will be prompted to see if you would like to save the data or discard your changes (see JRA-14911).

Index Queue

Index updates are now put in a queue. So even if the update takes longer than 30 seconds, the operation remains on the queue and is not lost. (See JRA-14220.)

^Top

Plus over 900 other fixes and improvements

Click here for full list.

JIRA Issues (200 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-1538		Filter on Versions and Components across Projects		Resolved

JRA-1579	Create a portlet for the recent history	Resolved
JRA-2033	Add an RSS feed query for comments to individual issues	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-2810	Recently viewed issues	Resolved
JRA-2916	Allow Previous version searching	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-3206	View issues without an estimate	Resolved
JRA-3624	released/unreleased version filter	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-5383	My Votes and My Watches as filters	Resolved
JRA-7551	Provide capability to find issues by resolution date	Resolved
JRA-7626	Build search queries remotely	Resolved
JRA-7772	Ability to create advanced queries to search across all data	Resolved
JRA-8159	Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type	Resolved
JRA-8527	Put task actions directly in filter output	Resolved
JRA-8606	Need a way to find watched issues	Resolved
JRA-8852	Sort filter results by non-visible field	Resolved
JRA-8973	RSS of Project Changes	Resolved
JRA-9551	Search for all Sub-Tasks of one given issue	Resolved
JRA-9651	User Activity Log	Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"	Resolved
JRA-10443	"Not Assigned to User" criteria in filters	Resolved
JRA-10603	MultipleSelect searcher for cascading selection field	Resolved
JRA-12921	Ability to export Watched Issues to excel	Resolved
JRA-14613	Each project can have its own logo	Resolved
JRA-14616	Ability to query for issues that you are not watching	Resolved
JRA-14983	Fetch only updated or changed issues	Resolved

JRA-15575	Test and confim JIRA is compatible with Microsoft SQL Server 2008	Closed
JRA-16067	Provide field definition in XML issue view URL to customize XML view	Resolved
JRA-16120	Dashboard rewrite	Resolved
JRA-16509	Check for javascript enabled in browser	Resolved
JRA-16805	Convert legacy portlets to Gadgets	Resolved
JRA-16807	Convert Intro Portlet	Resolved
JRA-16808	Convert Quicklinks portlet	Resolved
JRA-16809	Convert Favourite Fitlers	Resolved
JRA-16811	Convert Pie Chart Portlet	Resolved
JRA-16903	Convert CreatedVsResolved Chart Portlet	Resolved
JRA-16905	Convert Average Age Chart	Resolved
JRA-16914	Convert Recently Created Portlet	Resolved
JRA-16916	Convert Time Since Chart	Resolved
JRA-16920	Add filter option for subtasks of a particular issue	Resolved
JRA-16926	Implement OAuth SPI in JIRA	Resolved
JRA-16931	Convert Saved Filter Portlet	Resolved
JRA-16978	Option "Number to Show" in Filter Statistics Portlet to limit numer of rows displayed	Resolved
JRA-17090	Convert Admin Portlet to gadget	Resolved
JRA-17094	Convert Bugzilla portlet	Resolved
JRA-17095	Convert Project/Projects/Project Table portlets into a gadget	Resolved
JRA-17096	Convert Project / Filter stats Portlet to gadget	Resolved
JRA-17101	Convert TwoDimensionalStatsPortlet to gadgets	Resolved
JRA-17112	Convert Roadmap Portlet to gadgets	Resolved
JRA-17133	Convert Resolution Time Chart to gadgets	Resolved
JRA-17140	Convert Assign To Me portlet to gadget	Resolved
JRA-17141	Convert Voted For Portlet to gadget	Resolved
JRA-17142	Convert Watching Portlet to gadget	Resolved

JRA-17143	Convert In-progress portlet to gadget	Resolved
JRA-17182	Allow REST plugins to be decorated	Resolved
JRA-17337	New Header for JIRA 4.0	Resolved
JRA-17355	Need provision to search multiple group	Resolved
JRA-17763	JQL: It will great if user can use E-mail ID's for searching in Assignee and reporter fields	Resolved
JRA-923	Allow filter by "No Fix For" across projects	Resolved
JRA-1560	Better support for logical operation (and/or/not) type of filters.	Resolved
JRA-1635	"not" qualifier on fields for searching	Resolved
JRA-1642	Create home directory instead of index & attachment directory	Resolved
JRA-1800	Improve the UI for browse project	Resolved
JRA-1844	Display attachment comments associated with their attachments	Resolved
JRA-1983	Enable filtering on "older than 1 month"	Resolved
JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-2469	It would be really nice to specficy several Asignee options in filters	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-2852	search for issues on version lower or equal to a given version	Resolved
JRA-3000	Add key NUMBER (only number) searching to default search filter.	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-3114	Request: add optional icon for each project	Resolved
JRA-3451	Enable filtering by Date Resolved	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-4227	Recent History Popup - persistance across sessions & more data	Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria	Resolved
JRA-4688	Browse Project: Within the tab panel, if components are hidden - the version info appears to be right aligned.	Resolved
JRA-5121	Filter Portlet with configurable columns	Resolved
JRA-5152	Show issue linked to another issue.	Resolved
JRA-5310	Watchlist should be exportable	Resolved

JRA-5435	Issue actions and operations on Issue Navigator	Resolved
JRA-5560	Improved query functionality	Resolved
JRA-5798	Project Portlet: needs multi project selection	Resolved
JRA-5844	allow multiple users when creating filters	Resolved
JRA-5965	Allow configure units of time tracking	Resolved
JRA-6010	Thought processing	Resolved
JRA-6170	Radio Buttons should support Select List Searcher template	Resolved
JRA-6180	Search for a custom field that is empty	Resolved
JRA-6527	Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-7068	Allow for list of issues to be saved as a filter	Resolved
JRA-7909	Search/ filter for "empty" fields	Resolved
JRA-8487	Bad logging from uk.ltd.getahead.dwr.util.CommonsLoggingOutput on startup	Resolved
JRA-8686	Allow searching of issues by Full Name for all user fields	Resolved
JRA-8758	Cannot create filter for multiple projects all issues in version "Released Versions"	Resolved
JRA-8806	Allow "Released" & "Unreleased" Version search accross multiple projects	Resolved
JRA-9115	Ability to search for issues with no due date associated	Resolved
JRA-9278	New Field "Resolution Date" automatically filled with date of setting resolution	Resolved
JRA-9658	Minor css bug (cursor)	Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.	Resolved
JRA-10405	Attachment ordering	Resolved
JRA-10492	Search for several users as Assignee or Reporter	Resolved
JRA-10644	Make filters more accessible	Resolved
JRA-10658	More columns on Dashboards	Resolved
JRA-11134	Allow setting of column order/sort with no issues in result set	Resolved
JRA-12177	Time tracking by using setting "hours" - edit issue shows "Original/Remaining Estimate" -field value in "pretty" mode	Resolved
JRA-12596	Enable cross-project filtering on special versions	Resolved
JRA-12656	Add paging/optimization for Change Log scope	Resolved

JRA-13426	Next/previous version links for 'Browse Version' screen	Resolved
JRA-13745	Clean up top toolbar by moving Profile link to username and removing Filters link	Resolved
JRA-13850	Servlet Content Listeners should implement the catch / log / rethrow pattern	Resolved
JRA-14220	Ensure the index optimize operation does not cause index lock timeouts	Resolved
JRA-14419	Warning for Websphere installation on validating entity-engine.xml	Closed
JRA-14513	JIRA Soap Service log and Access filter log footprint needs improving - Invoked Method would be handy	Resolved
JRA-14516	JIRA upgrade page should warn about possible character encoding issue if JIRA is moved between two servers.	Closed
JRA-14598	Add access key for administer project from browse project	Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype	Resolved
JRA-14826	Dashboards with a large number of portal pages cause the page to become too wide	Resolved
JRA-15018	Improved SOAP and HTTP access logging	Resolved
JRA-15266	Would prefer if the "Worklog" heading under "Operations" when viewing an issue were a link, like the rest of the operations.	Closed
JRA-15445	RPC plugin needs to be cleaned up	Resolved
JRA-15517	Upgrade JIRA to use the latest version of Lucene indexing framework - v2.3.2	Resolved
JRA-15543	Show release date next to version name in the list of versions on Browse Project screen	Resolved
JRA-15646	Convert JIRA to jQuery	Resolved
JRA-15649	Sort list of plugins in Admin section alphabetically	Resolved
JRA-15665	Address issue of plugins pushing filter/report panel off screen	Resolved
JRA-15666	Add project information to the issue XML view	Resolved
JRA-15700	Created VS Resolved cumulative + individual graphs' Y axis should should be independent	Resolved
JRA-15702	Migrate to licensing 2.0	Resolved
JRA-15732	Update email documentation to highlight that OutOfMemoryError can stop email processing	Closed
JRA-15846	Allow changing license on Data Import in new JIRA instance.	Resolved
JRA-15872	"Browse Project" URL doesn't include current project	Resolved
JRA-15886	Add logging notification for index optimization events	Resolved
JRA-15920	Include warning in EAR/WAR documentation not to edit anything directly on the application server	Closed
JRA-15962	Upgrade JIRA to Plugins 2.x	Resolved

JRA-15991	Merge translations files into one to make translating JIRA easier!	Resolved
JRA-16058	Aggressive locking in JiraCachingPropertySet causes high contention	Resolved
JRA-16113	Do not show negative values on Y axis in Created vs. Resolved chart	Resolved
JRA-16122	HTTP Basic auth should be enabled by default	Resolved
JRA-16138	Anonymous users should not be considered to "own" all anonymous comments.	Resolved
JRA-16210	Display issue count on JQL execution	Resolved
JRA-16211	Enter / return should execute JQL	Resolved
JRA-16253	Source Build documentation is out of date and incomplete	Closed
JRA-16276	Adjust colours of Resoltion date chart to be more distinshable for the colour blind	Resolved
JRA-16278	Add ability to search for versions using regex or similar	Resolved
JRA-16379	Weblogic Deployment descriptor (weblogic.xml) has changed for Weblogic 9.x	Closed
JRA-16424	log4j output should contain more information	Resolved
JRA-16443	Create a jira.field.resolution.include transition attribute	Resolved
JRA-16510	Update to jQuery 1.3.2	Resolved
JRA-16522	Searching according to multiple assignees should be provided.	Resolved
JRA-16661	No way to cancel the "Project avatar" dialog	Resolved
JRA-16698	Have the ability to log all SQL statements issued by JIRA and also have a callback for timing purposes	Resolved
JRA-16744	Improve the performance of checking if a user belongs to a particular group.	Resolved
JRA-16793	A new section "JIRA Configuration" is needed in System Info page	Resolved
JRA-16838	Ensure Save button is disabled after submitting on the chart popup	Resolved
JRA-16839	Add nicer dashboard tabs	Resolved
JRA-16846	Allow for Pluggable Decorators in JIRA	Resolved
JRA-16870	Improve multi-threaded liveness of FieldLayoutManager under load	Resolved
JRA-17025	Link to the JIRA Knowledge Base in the Apache docs	Closed
JRA-17128	Profiling document broken link	Closed
JRA-17215	Environment field is missing from the bulk operations screen	Resolved
JRA-17217	French translation for "Road Map"	Resolved

JRA-17296	Add attribute for hidding a gadget when not logged in	Resolved
JRA-17314	Need a way to use the WebResourceManager to include JS/CSS for project tab panels and portlets	Resolved
JRA-17324	Make browse project/component/version use cached url for project avatar	Resolved
JRA-17361	Style gadget mini-messages to be more atlassian like	Resolved
JRA-17391	JIRA_HOME path in windows needs to be specified using backslash(or as Mac) and not as default windows way. Needs to document it to avoid confusion.	Closed
JRA-17404	Improve logging/UI when plugins can't be loaded.	Resolved
JRA-17412	Ability for support to easily tell if a patch has been applied	Resolved
JRA-17429	JIRA should only persist a plugin as disabled if it was explicitly disabled by the user. If it got disabled because it was invalid, then JIRA should try to enable it on restart.	Resolved
JRA-17474	Choosing an non-existing drive as destination directory for windows installer give wrong error message	Resolved
JRA-17483	Need to add info about jira.home to README files in top directory	Closed
JRA-17496	re-write dashboard client-side	Closed
JRA-17515	Put the JQL slow log into its own file.	Resolved
JRA-17562	Performance Improvements for JIRA v4.0	Resolved
JRA-17674	Issue summary page: Please duplicate link "Manage Attachments" under section "Image Attachments"	Resolved
JRA-17682	Add a warning to the reports documentation.	Closed
JRA-17698	Remove dependency on backport.util.concurrent	Resolved
JRA-17756	Remove DWR from JIRA	Resolved
JRA-17779	JQL reserve words should be mention in documentation	Closed
JRA-17797	JQL: Some points to add in documentation of JQL	Closed
JRA-17916	Update JAC hardware spec on Requirements page	Closed
JRA-17950	Instructions for deleting an issue type are too vague, and docs do not further explain them	Closed
JRA-17978	Search sort order doesn't persist when searching on free text	Resolved
JRA-18017	Footer improvements - like Bamboo	Resolved
JRA-18073	Create issue button should be aware of the current project	Resolved
JRA-18101	LDAP Integration document is confusing about what order passwords will be checked.	Closed
JRA-18102	JQL: Error message can be improved for date field when user doesn't put quotes for value	Resolved
JRA-18105	add documentation on JQL "autocomplete"	Closed

JRA-18110	Tool tip text for "Manage dashboard" has unecessary spaces between words	Resolved
JRA-18133	Upgrade to the latest version of Seraph and Trusted Applications library	Resolved
JRA-18145	In order for charts to appear in the charting popup, they should be in both categories JIRA and Charts	Resolved
JRA-18147	on "General Config" screen, change "JQL Auto-complete Disabled" to just "JQL Auto-complete" (as per "Issue Picker Auto-complete")	Resolved
JRA-18165	JIRA 4.0 GUI comments	Resolved
JRA-18225	It would be cool to have the operations (save, save as, etc) in the advanced navigator view so you do not have to go to the view tab	Resolved
JRA-18228	Per-user JQL autocomplete setting	Resolved
JRA-18232	It will be great if we can limit the number of favourite filters in issue drop down to 5	Resolved
JRA-18236	JQL History links should be middle clickable	Resolved
JRA-18237	The error message 'Query is too complex to display in the Issue Navigator edit controls' should be changed to "Query is too complex to display in Simple mode"	Resolved

^Top

JIRA 4.0 Beta 5 Release Notes

September 15, 2009

JIRA 4.0 Beta 5 is a public development release leading up to **JIRA 4.0**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

It is not possible to upgrade JIRA 4.0 Beta 1, 2, 3 or 4 data to Beta 5. The data needs to come from an already released JIRA version (for example, JIRA 3.13.5).

The **only** plugin that is compatible with JIRA 4.0 Beta is the latest JIRA Toolkit. Do not install any other plugins.

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

Highlights of JIRA 4.0 Beta 5:

- Advanced Searching
- Dashboard Gadgets
- Activity Stream
- Issue Actions in the Issue Navigator
- Charting Now Comes Standard

- New-look "Browse Project"
- Project Icons
- New-look Header
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- Plus over 800 other fixes and improvements

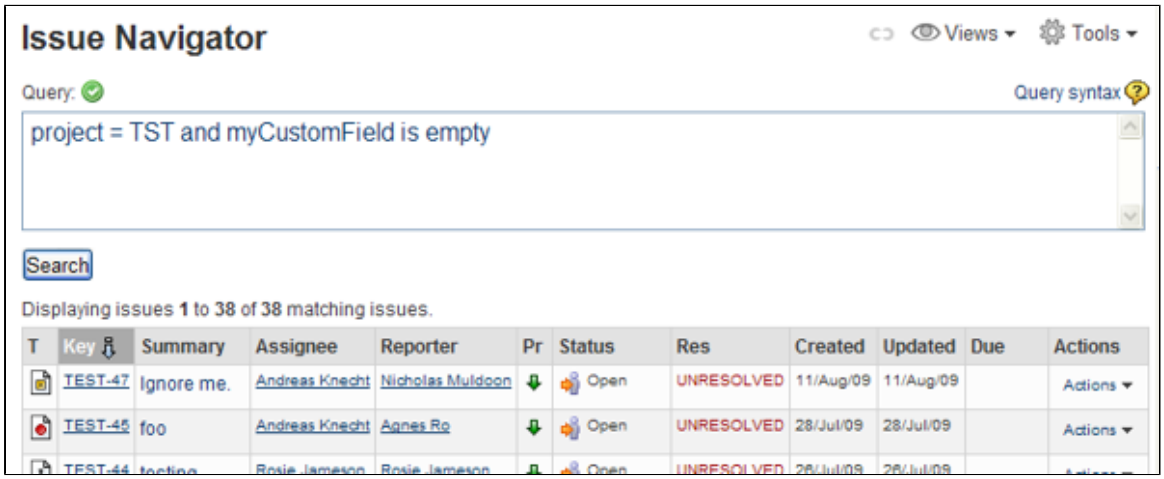
Thank you for your interest in JIRA 4.0 Beta 5
Download Beta

Installing/Upgrading to JIRA 4.0
JIRA 4.0 Beta 5 can be downloaded here. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide. You will also need to go to my.atlassian.com and generate a "**JIRA 4 Evaluation**" license, if you haven't done so previously, as any existing 3.x license files will not work with 4.0 Beta 5.

Highlights of JIRA 4.0 Beta 5

Advanced Searching

The new advanced search (JQL) provides support for logical operations, including AND, OR, NOT, NULL, EMPTY --- even on custom fields:



For more on the new JQL search syntax, please see the documentation.

^Top

Dashboard Gadgets

The new-look JIRA dashboard not only looks awesome: it now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

You can easily customise your dashboard by choosing a different layout, adding more gadgets, dragging the gadgets into different positions, and changing the look of individual gadgets.

What's happened to your favourite JIRA portlets? Don't worry: every portlet that previously shipped with JIRA has been converted to a 'legacy gadget'. And if you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.

The following gadgets are available in Beta 4:

- 'Activity Stream' gadget (see below)

- 'Admin' gadget
- 'Bamboo Plan Summary' gadget*
- 'Bamboo Status' gadget*
- 'Crucible Charting' gadget
- 'Create Issue' gadget
- 'Favourite Filters' gadget
- 'Filter Results' gadget
- 'FishEye Charting' gadget
- 'Introduction' gadget
- 'Issue Completed This Iteration' gadget
- 'Login' gadget
- 'Quicklinks' gadget
- 'Voted Issues' gadget

* Requires Bamboo 2.3.2 Beta 1 or later.

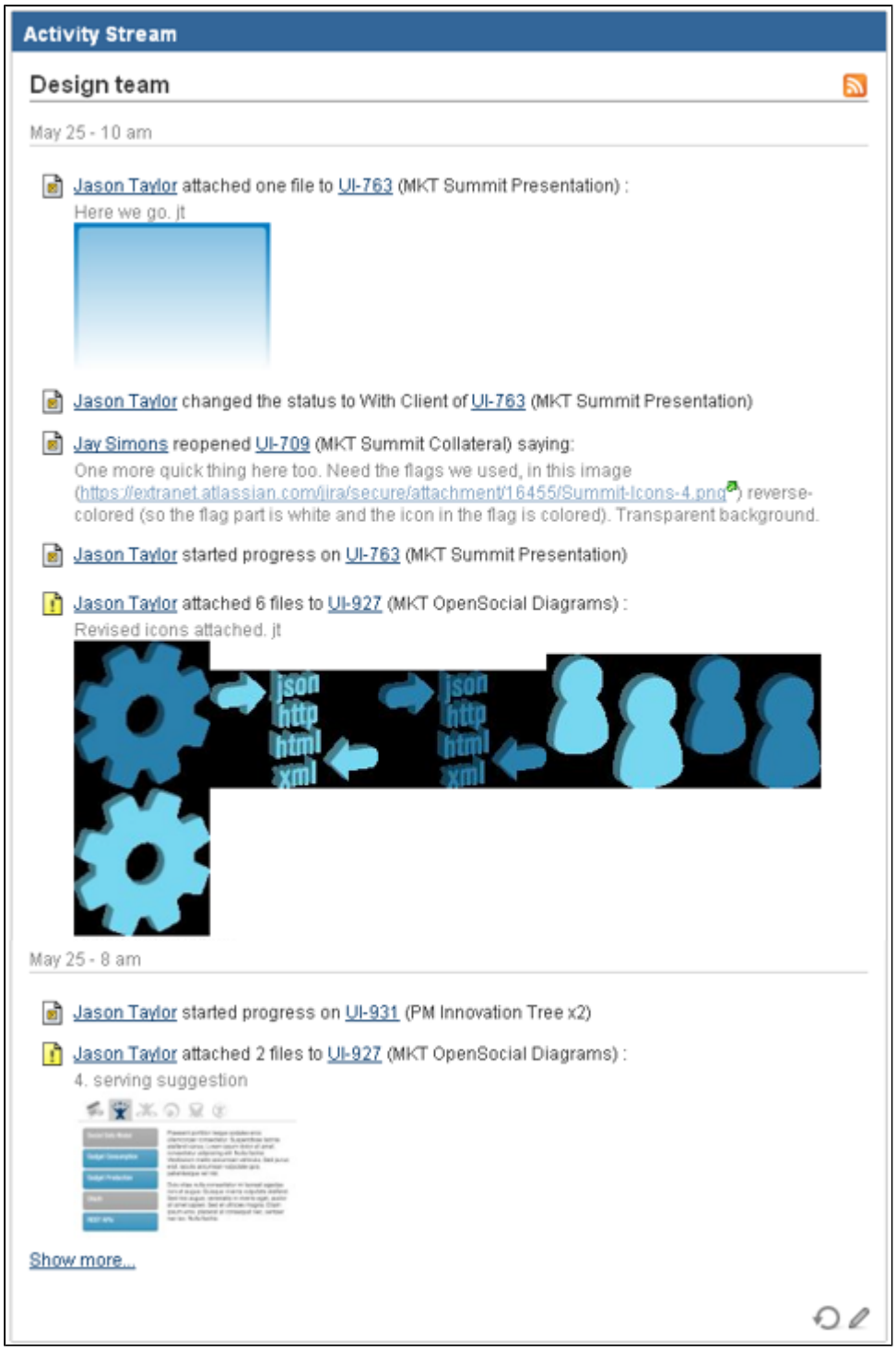
^Top

Activity Stream

The new '**Activity Stream**' gadget displays a summary of the latest activity in JIRA projects (and/or by particular people) in which you are interested.

The '**Activity Stream**' gadget also provides an RSS feed, allowing you to create very specific RSS feeds of only the information that is most relevant to you.

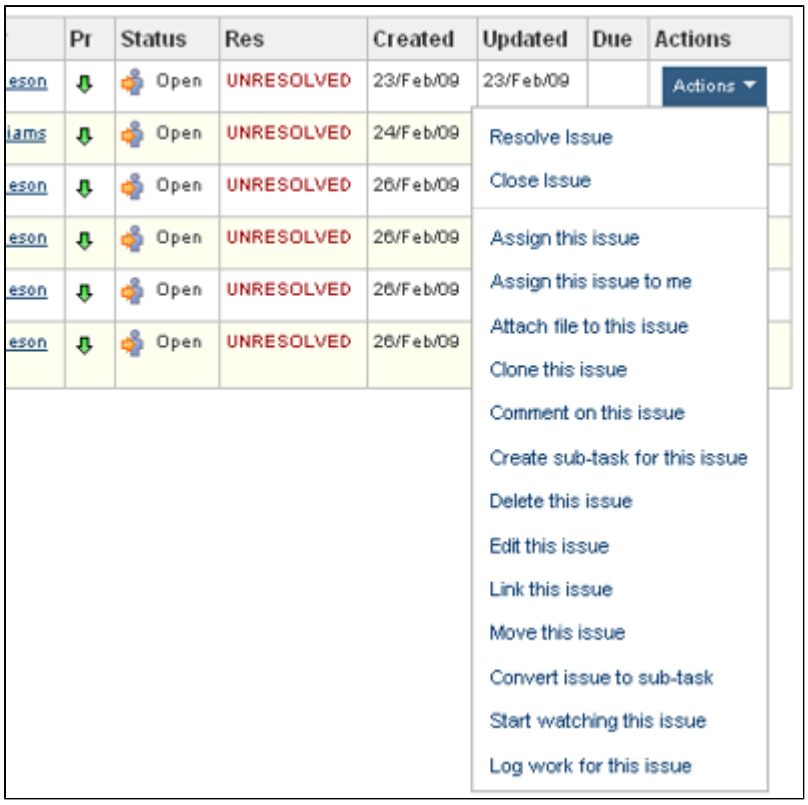
See the documentation for more details.



^Top

Issue Actions in the Issue Navigator

By popular request, issues can now be actioned directly from the Issue Navigator:

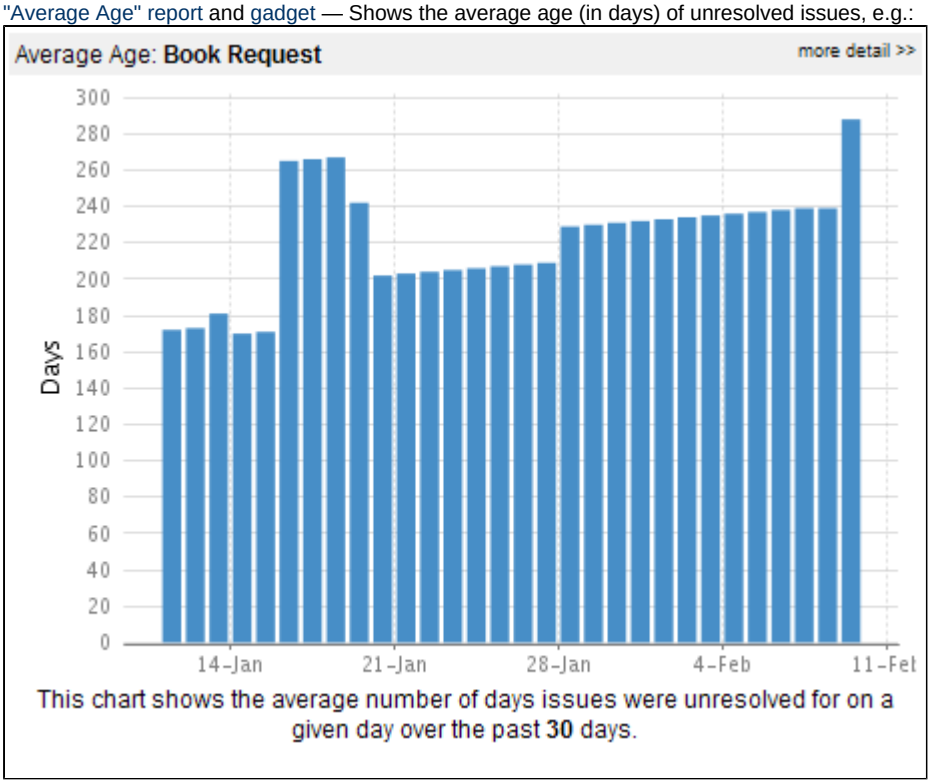


The "Actions" menu is also available for the list of sub-tasks within an issue.
^Top

Charting Now Comes Standard

The following reports and gadgets from the Charting plugin have now been integrated into JIRA:

- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over a given

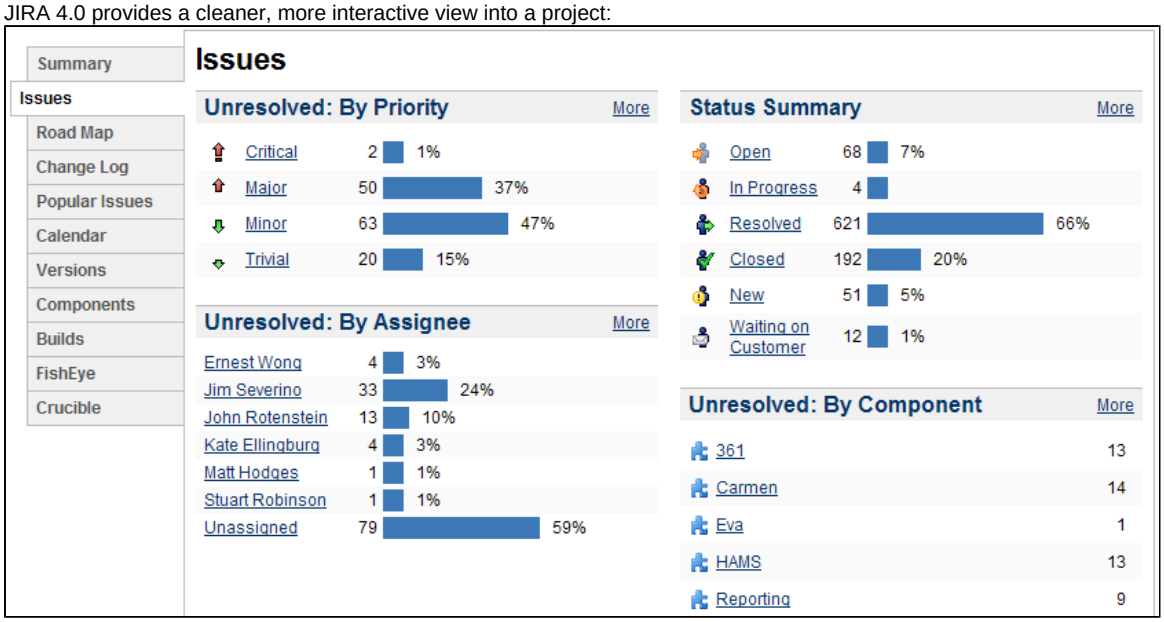
period of time.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.

Also, the "Resolved" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution date recorded.

^Top

New-look "Browse Project"

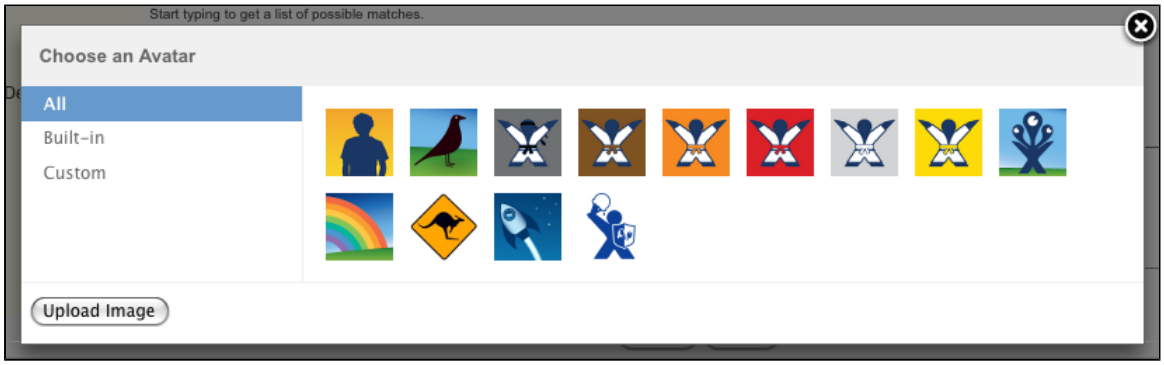
JIRA 4.0 provides a cleaner, more interactive view into a project:



See the documentation for more about browsing projects, versions and components.
^Top

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):

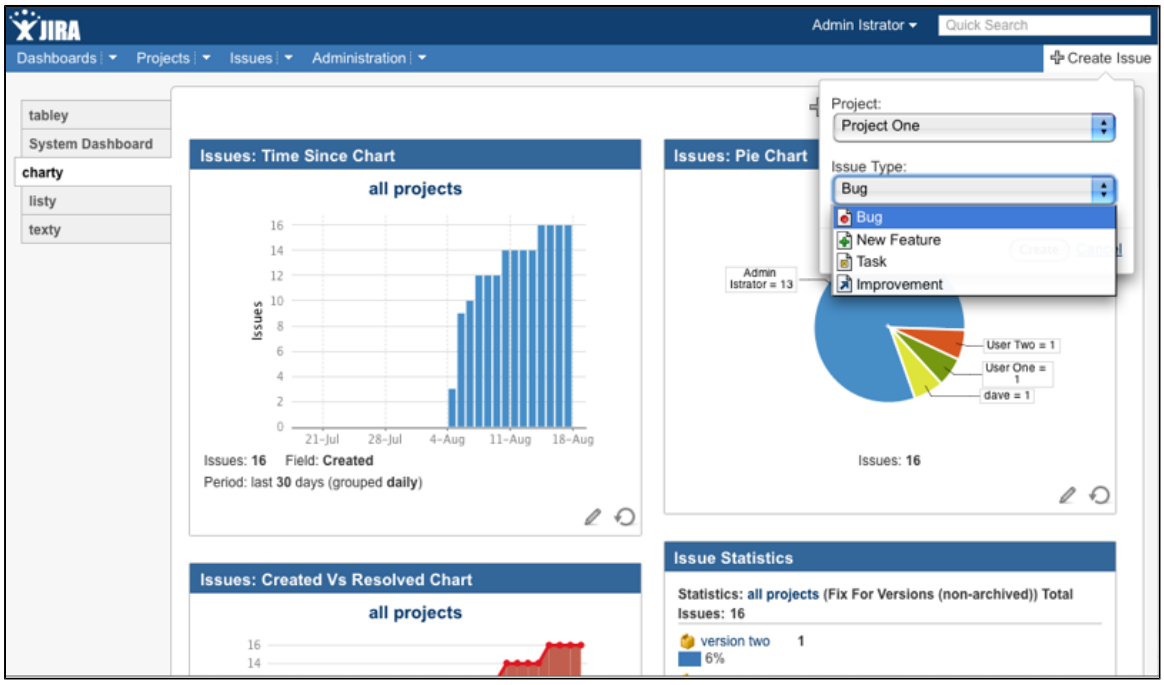


^Top

8

New-look Header

The new-look JIRA header gives you quick access to all the most commonly-used functions. Creating an issue just got super-fast!



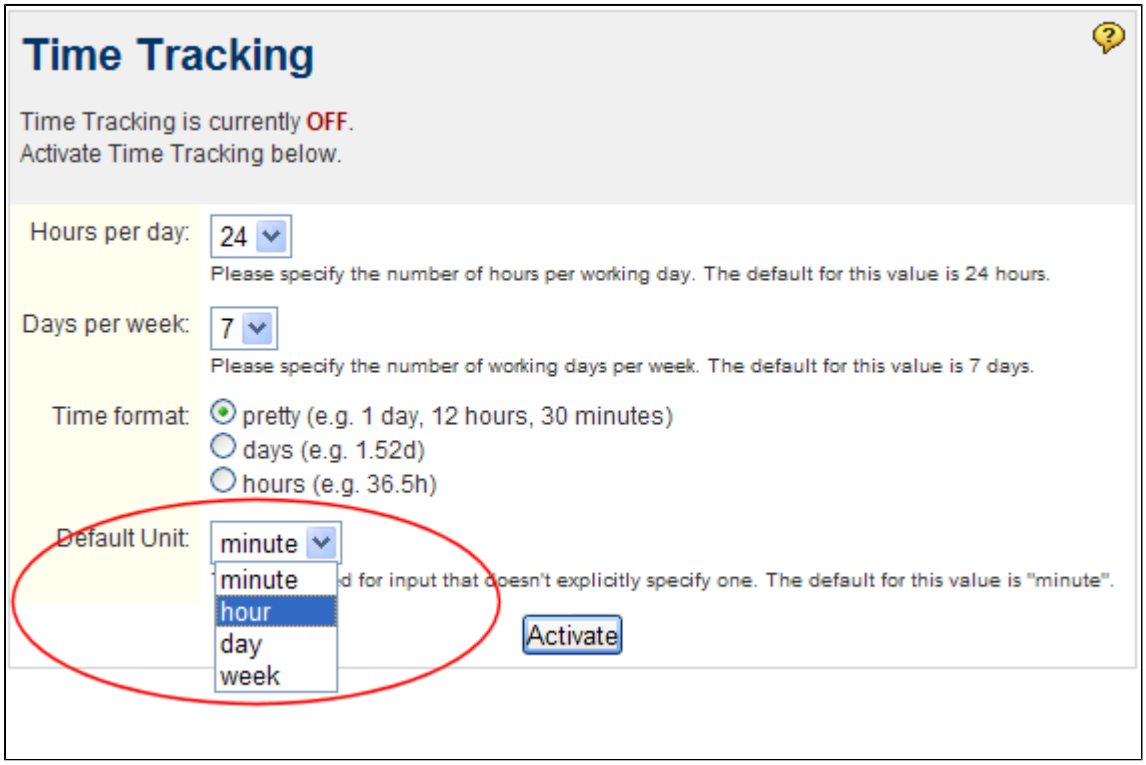
Click to zoom in:

If you prefer keystrokes rather than mouse-clicks, you'll be pleased to know that you can use your keyboard to navigate the new header menus.
^Top

9

Default Unit for Time Tracking

You can now specify your preferred **Default Unit** (minutes/hours/days/weeks) for your JIRA system. This will be applied whenever users log work on an issue without specifying a unit.



^Top

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.

^Top

Engine Room

Beyond the 'Back' Button

When navigating away from a page where you have modified data, you will be prompted to see if you would like to save the data or discard your changes (see JRA-14911).

Index Queue

Index updates are now put in a queue. So even if the update takes longer than 30 seconds, the operation remains on the queue and is not lost. (See JRA-14220.)

^Top

Plus over 800 other fixes and improvements

Click here for full list.

JIRA Issues (200 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-1538		Filter on Versions and Components across Projects		Resolved

JRA-1579	Create a portlet for the recent history	Resolved
JRA-2033	Add an RSS feed query for comments to individual issues	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-2810	Recently viewed issues	Resolved
JRA-2916	Allow Previous version searching	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-3206	View issues without an estimate	Resolved
JRA-3624	released/unreleased version filter	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-5383	My Votes and My Watches as filters	Resolved
JRA-7551	Provide capability to find issues by resolution date	Resolved
JRA-7626	Build search queries remotely	Resolved
JRA-7772	Ability to create advanced queries to search across all data	Resolved
JRA-8159	Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type	Resolved
JRA-8527	Put task actions directly in filter output	Resolved
JRA-8606	Need a way to find watched issues	Resolved
JRA-8852	Sort filter results by non-visible field	Resolved
JRA-8973	RSS of Project Changes	Resolved
JRA-9551	Search for all Sub-Tasks of one given issue	Resolved
JRA-9651	User Activity Log	Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"	Resolved
JRA-10443	"Not Assigned to User" criteria in filters	Resolved
JRA-10603	MultipleSelect searcher for cascading selection field	Resolved
JRA-12921	Ability to export Watched Issues to excel	Resolved
JRA-14613	Each project can have its own logo	Resolved
JRA-14616	Ability to query for issues that you are not watching	Resolved
JRA-14983	Fetch only updated or changed issues	Resolved

JRA-15575	Test and confim JIRA is compatible with Microsoft SQL Server 2008	Closed
JRA-16067	Provide field definition in XML issue view URL to customize XML view	Resolved
JRA-16120	Dashboard rewrite	Resolved
JRA-16509	Check for javascript enabled in browser	Resolved
JRA-16805	Convert legacy portlets to Gadgets	Resolved
JRA-16807	Convert Intro Portlet	Resolved
JRA-16808	Convert Quicklinks portlet	Resolved
JRA-16809	Convert Favourite Fitlers	Resolved
JRA-16811	Convert Pie Chart Portlet	Resolved
JRA-16903	Convert CreatedVsResolved Chart Portlet	Resolved
JRA-16905	Convert Average Age Chart	Resolved
JRA-16914	Convert Recently Created Portlet	Resolved
JRA-16916	Convert Time Since Chart	Resolved
JRA-16920	Add filter option for subtasks of a particular issue	Resolved
JRA-16926	Implement OAuth SPI in JIRA	Resolved
JRA-16931	Convert Saved Filter Portlet	Resolved
JRA-16978	Option "Number to Show" in Filter Statistics Portlet to limit numer of rows displayed	Resolved
JRA-17090	Convert Admin Portlet to gadget	Resolved
JRA-17094	Convert Bugzilla portlet	Resolved
JRA-17095	Convert Project/Projects/Project Table portlets into a gadget	Resolved
JRA-17096	Convert Project / Filter stats Portlet to gadget	Resolved
JRA-17101	Convert TwoDimensionalStatsPortlet to gadgets	Resolved
JRA-17112	Convert Roadmap Portlet to gadgets	Resolved
JRA-17133	Convert Resolution Time Chart to gadgets	Resolved
JRA-17140	Convert Assign To Me portlet to gadget	Resolved
JRA-17141	Convert Voted For Portlet to gadget	Resolved
JRA-17142	Convert Watching Portlet to gadget	Resolved

JRA-17143	Convert In-progress portlet to gadget	Resolved
JRA-17182	Allow REST plugins to be decorated	Resolved
JRA-17337	New Header for JIRA 4.0	Resolved
JRA-17355	Need provision to search multiple group	Resolved
JRA-17763	JQL: It will great if user can use E-mail ID's for searching in Assignee and reporter fields	Resolved
JRA-923	Allow filter by "No Fix For" across projects	Resolved
JRA-1560	Better support for logical operation (and/or/not) type of filters.	Resolved
JRA-1635	"not" qualifier on fields for searching	Resolved
JRA-1642	Create home directory instead of index & attachment directory	Resolved
JRA-1800	Improve the UI for browse project	Resolved
JRA-1844	Display attachment comments associated with their attachments	Resolved
JRA-1983	Enable filtering on "older than 1 month"	Resolved
JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-2469	It would be really nice to specficy several Asignee options in filters	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-2852	search for issues on version lower or equal to a given version	Resolved
JRA-3000	Add key NUMBER (only number) searching to default search filter.	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-3114	Request: add optional icon for each project	Resolved
JRA-3451	Enable filtering by Date Resolved	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-4227	Recent History Popup - persistance across sessions & more data	Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria	Resolved
JRA-4688	Browse Project: Within the tab panel, if components are hidden - the version info appears to be right aligned.	Resolved
JRA-5121	Filter Portlet with configurable columns	Resolved
JRA-5152	Show issue linked to another issue.	Resolved
JRA-5310	Watchlist should be exportable	Resolved

JRA-5435	Issue actions and operations on Issue Navigator	Resolved
JRA-5560	Improved query functionality	Resolved
JRA-5798	Project Portlet: needs multi project selection	Resolved
JRA-5844	allow multiple users when creating filters	Resolved
JRA-5965	Allow configure units of time tracking	Resolved
JRA-6010	Thought processing	Resolved
JRA-6170	Radio Buttons should support Select List Searcher template	Resolved
JRA-6180	Search for a custom field that is empty	Resolved
JRA-6527	Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-7068	Allow for list of issues to be saved as a filter	Resolved
JRA-7909	Search/ filter for "empty" fields	Resolved
JRA-8487	Bad logging from uk.ltd.getahead.dwr.util.CommonsLoggingOutput on startup	Resolved
JRA-8686	Allow searching of issues by Full Name for all user fields	Resolved
JRA-8758	Cannot create filter for multiple projects all issues in version "Released Versions"	Resolved
JRA-8806	Allow "Released" & "Unreleased" Version search accross multiple projects	Resolved
JRA-9115	Ability to search for issues with no due date associated	Resolved
JRA-9278	New Field "Resolution Date" automatically filled with date of setting resolution	Resolved
JRA-9658	Minor css bug (cursor)	Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.	Resolved
JRA-10405	Attachment ordering	Resolved
JRA-10492	Search for several users as Assignee or Reporter	Resolved
JRA-10644	Make filters more accessible	Resolved
JRA-10658	More columns on Dashboards	Resolved
JRA-11134	Allow setting of column order/sort with no issues in result set	Resolved
JRA-12177	Time tracking by using setting "hours" - edit issue shows "Original/Remaining Estimate" -field value in "pretty" mode	Resolved
JRA-12596	Enable cross-project filtering on special versions	Resolved
JRA-12656	Add paging/optimization for Change Log scope	Resolved

JRA-13426	Next/previous version links for 'Browse Version' screen	Resolved
JRA-13745	Clean up top toolbar by moving Profile link to username and removing Filters link	Resolved
JRA-13850	Servlet Content Listeners should implement the catch / log / rethrow pattern	Resolved
JRA-14220	Ensure the index optimize operation does not cause index lock timeouts	Resolved
JRA-14419	Warning for Websphere installation on validating entity-engine.xml	Closed
JRA-14513	JIRA Soap Service log and Access filter log footprint needs improving - Invoked Method would be handy	Resolved
JRA-14516	JIRA upgrade page should warn about possible character encoding issue if JIRA is moved between two servers.	Closed
JRA-14598	Add access key for administer project from browse project	Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype	Resolved
JRA-14826	Dashboards with a large number of portal pages cause the page to become too wide	Resolved
JRA-15018	Improved SOAP and HTTP access logging	Resolved
JRA-15266	Would prefer if the "Worklog" heading under "Operations" when viewing an issue were a link, like the rest of the operations.	Closed
JRA-15445	RPC plugin needs to be cleaned up	Resolved
JRA-15517	Upgrade JIRA to use the latest version of Lucene indexing framework - v2.3.2	Resolved
JRA-15543	Show release date next to version name in the list of versions on Browse Project screen	Resolved
JRA-15646	Convert JIRA to jQuery	Resolved
JRA-15649	Sort list of plugins in Admin section alphabetically	Resolved
JRA-15665	Address issue of plugins pushing filter/report panel off screen	Resolved
JRA-15666	Add project information to the issue XML view	Resolved
JRA-15700	Created VS Resolved cumulative + individual graphs' Y axis should should be independent	Resolved
JRA-15702	Migrate to licensing 2.0	Resolved
JRA-15732	Update email documentation to highlight that OutOfMemoryError can stop email processing	Closed
JRA-15846	Allow changing license on Data Import in new JIRA instance.	Resolved
JRA-15872	"Browse Project" URL doesn't include current project	Resolved
JRA-15886	Add logging notification for index optimization events	Resolved
JRA-15920	Include warning in EAR/WAR documentation not to edit anything directly on the application server	Closed
JRA-15962	Upgrade JIRA to Plugins 2.x	Resolved

JRA-15991	Merge translations files into one to make translating JIRA easier!	Resolved
JRA-16058	Aggressive locking in JiraCachingPropertySet causes high contention	Resolved
JRA-16113	Do not show negative values on Y axis in Created vs. Resolved chart	Resolved
JRA-16122	HTTP Basic auth should be enabled by default	Resolved
JRA-16138	Anonymous users should not be considered to "own" all anonymous comments.	Resolved
JRA-16210	Display issue count on JQL execution	Resolved
JRA-16211	Enter / return should execute JQL	Resolved
JRA-16253	Source Build documentation is out of date and incomplete	Closed
JRA-16276	Adjust colours of Resoltion date chart to be more distinshable for the colour blind	Resolved
JRA-16278	Add ability to search for versions using regex or similar	Resolved
JRA-16379	Weblogic Deployment descriptor (weblogic.xml) has changed for Weblogic 9.x	Closed
JRA-16424	log4j output should contain more information	Resolved
JRA-16443	Create a jira.field.resolution.include transition attribute	Resolved
JRA-16510	Update to jQuery 1.3.2	Resolved
JRA-16522	Searching according to multiple assignees should be provided.	Resolved
JRA-16661	No way to cancel the "Project avatar" dialog	Resolved
JRA-16698	Have the ability to log all SQL statements issued by JIRA and also have a callback for timing purposes	Resolved
JRA-16744	Improve the performance of checking if a user belongs to a particular group.	Resolved
JRA-16793	A new section "JIRA Configuration" is needed in System Info page	Resolved
JRA-16838	Ensure Save button is disabled after submitting on the chart popup	Resolved
JRA-16839	Add nicer dashboard tabs	Resolved
JRA-16846	Allow for Pluggable Decorators in JIRA	Resolved
JRA-16870	Improve multi-threaded liveness of FieldLayoutManager under load	Resolved
JRA-17025	Link to the JIRA Knowledge Base in the Apache docs	Closed
JRA-17128	Profiling document broken link	Closed
JRA-17215	Environment field is missing from the bulk operations screen	Resolved
JRA-17217	French translation for "Road Map"	Resolved

JRA-17296	Add attribute for hidding a gadget when not logged in	Resolved
JRA-17314	Need a way to use the WebResourceManager to include JS/CSS for project tab panels and portlets	Resolved
JRA-17324	Make browse project/component/version use cached url for project avatar	Resolved
JRA-17361	Style gadget mini-messages to be more atlassian like	Resolved
JRA-17391	JIRA_HOME path in windows needs to be specified using backslash(or as Mac) and not as default windows way. Needs to document it to avoid confusion.	Closed
JRA-17404	Improve logging/UI when plugins can't be loaded.	Resolved
JRA-17412	Ability for support to easily tell if a patch has been applied	Resolved
JRA-17429	JIRA should only persist a plugin as disabled if it was explicitly disabled by the user. If it got disabled because it was invalid, then JIRA should try to enable it on restart.	Resolved
JRA-17474	Choosing an non-existing drive as destination directory for windows installer give wrong error message	Resolved
JRA-17483	Need to add info about jira.home to README files in top directory	Closed
JRA-17496	re-write dashboard client-side	Closed
JRA-17515	Put the JQL slow log into its own file.	Resolved
JRA-17562	Performance Improvements for JIRA v4.0	Resolved
JRA-17674	Issue summary page: Please duplicate link "Manage Attachments" under section "Image Attachments"	Resolved
JRA-17682	Add a warning to the reports documentation.	Closed
JRA-17698	Remove dependency on backport.util.concurrent	Resolved
JRA-17756	Remove DWR from JIRA	Resolved
JRA-17779	JQL reserve words should be mention in documentation	Closed
JRA-17797	JQL: Some points to add in documentation of JQL	Closed
JRA-17916	Update JAC hardware spec on Requirements page	Closed
JRA-17950	Instructions for deleting an issue type are too vague, and docs do not further explain them	Closed
JRA-17978	Search sort order doesn't persist when searching on free text	Resolved
JRA-18017	Footer improvements - like Bamboo	Resolved
JRA-18073	Create issue button should be aware of the current project	Resolved
JRA-18101	LDAP Integration document is confusing about what order passwords will be checked.	Closed
JRA-18102	JQL: Error message can be improved for date field when user doesn't put quotes for value	Resolved
JRA-18105	add documentation on JQL "autocomplete"	Closed

JRA-18110	Tool tip text for "Manage dashboard" has unecessary spaces between words	Resolved
JRA-18133	Upgrade to the latest version of Seraph and Trusted Applications library	Resolved
JRA-18145	In order for charts to appear in the charting popup, they should be in both categories JIRA and Charts	Resolved
JRA-18147	on "General Config" screen, change "JQL Auto-complete Disabled" to just "JQL Auto-complete" (as per "Issue Picker Auto-complete")	Resolved
JRA-18165	JIRA 4.0 GUI comments	Resolved
JRA-18225	It would be cool to have the operations (save, save as, etc) in the advanced navigator view so you do not have to go to the view tab	Resolved
JRA-18228	Per-user JQL autocomplete setting	Resolved
JRA-18232	It will be great if we can limit the number of favourite filters in issue drop down to 5	Resolved
JRA-18236	JQL History links should be middle clickable	Resolved
JRA-18237	The error message 'Query is too complex to display in the Issue Navigator edit controls' should be changed to "Query is too complex to display in Simple mode"	Resolved

^Top

JIRA 4.0 Beta 4 Release Notes

September 6, 2009

JIRA 4.0 Beta 4 is a public development release leading up to **JIRA 4.0**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

It is not possible to upgrade JIRA 4.0 Beta 1, 2 or 3 data to Beta 4. The data needs to come from an already released JIRA version (for example, JIRA 3.13.5).

The **only** plugin that is compatible with JIRA 4.0 Beta is the latest JIRA Toolkit. Do not install any other plugins.

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

Highlights of JIRA 4.0 Beta 4:

- Advanced Searching
- Dashboard Gadgets
- Activity Stream
- Issue Actions in the Issue Navigator
- Charting Now Comes Standard

- New-look "Browse Project"
- Project Icons
- New-look Header
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- Plus over 800 other fixes and improvements

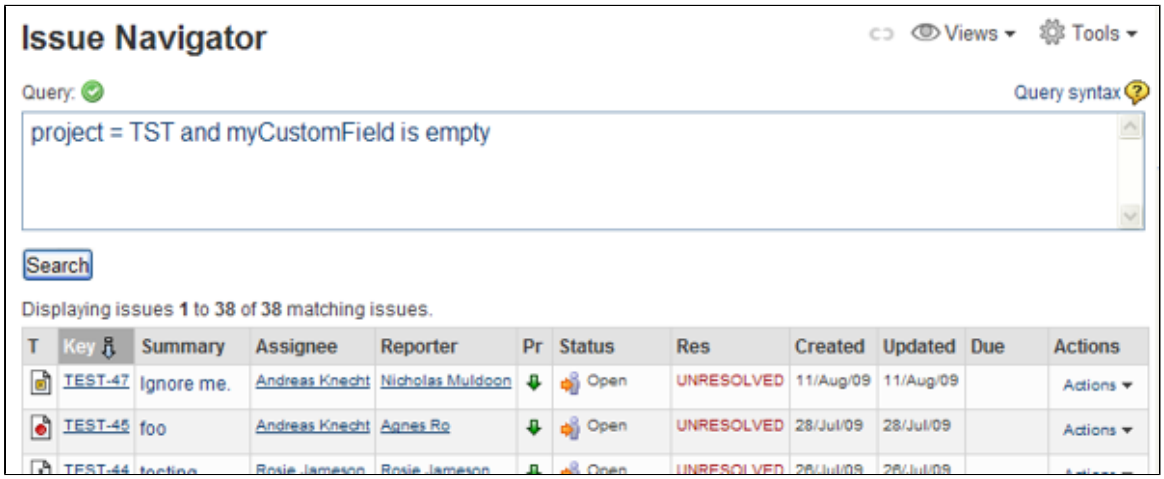
Thank you for your interest in JIRA 4.0 Beta 4
Download Beta

Installing/Upgrading to JIRA 4.0
JIRA 4.0 Beta 4 can be downloaded here. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide. You will also need to go to my.atlassian.com and generate a "**JIRA 4 Evaluation**" license, as any existing 3.x license files will not work with 4.0 Beta 4.

Highlights of JIRA 4.0 Beta 4

Advanced Searching

The new advanced search (JQL) provides support for logical operations, including AND, OR, NOT, NULL, EMPTY --- even on custom fields:



For more on the new JQL search syntax, please see the documentation.

^Top

Dashboard Gadgets

The new-look JIRA dashboard not only looks awesome: it now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

You can easily customise your dashboard by choosing a different layout, adding more gadgets, dragging the gadgets into different positions, and changing the look of individual gadgets.

What's happened to your favourite JIRA portlets? Don't worry: every portlet that previously shipped with JIRA has been converted to a 'legacy gadget'. And if you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.

The following gadgets are available in Beta 4:

- 'Activity Stream' gadget (see below)

- 'Admin' gadget
- 'Bamboo Plan Summary' gadget*
- 'Bamboo Status' gadget*
- 'Crucible Charting' gadget
- 'Create Issue' gadget
- 'Favourite Filters' gadget
- 'Filter Results' gadget
- 'FishEye Charting' gadget
- 'Introduction' gadget
- 'Issue Completed This Iteration' gadget
- 'Login' gadget
- 'Quicklinks' gadget
- 'Voted Issues' gadget

* Requires Bamboo 2.3.2 Beta 1 or later.

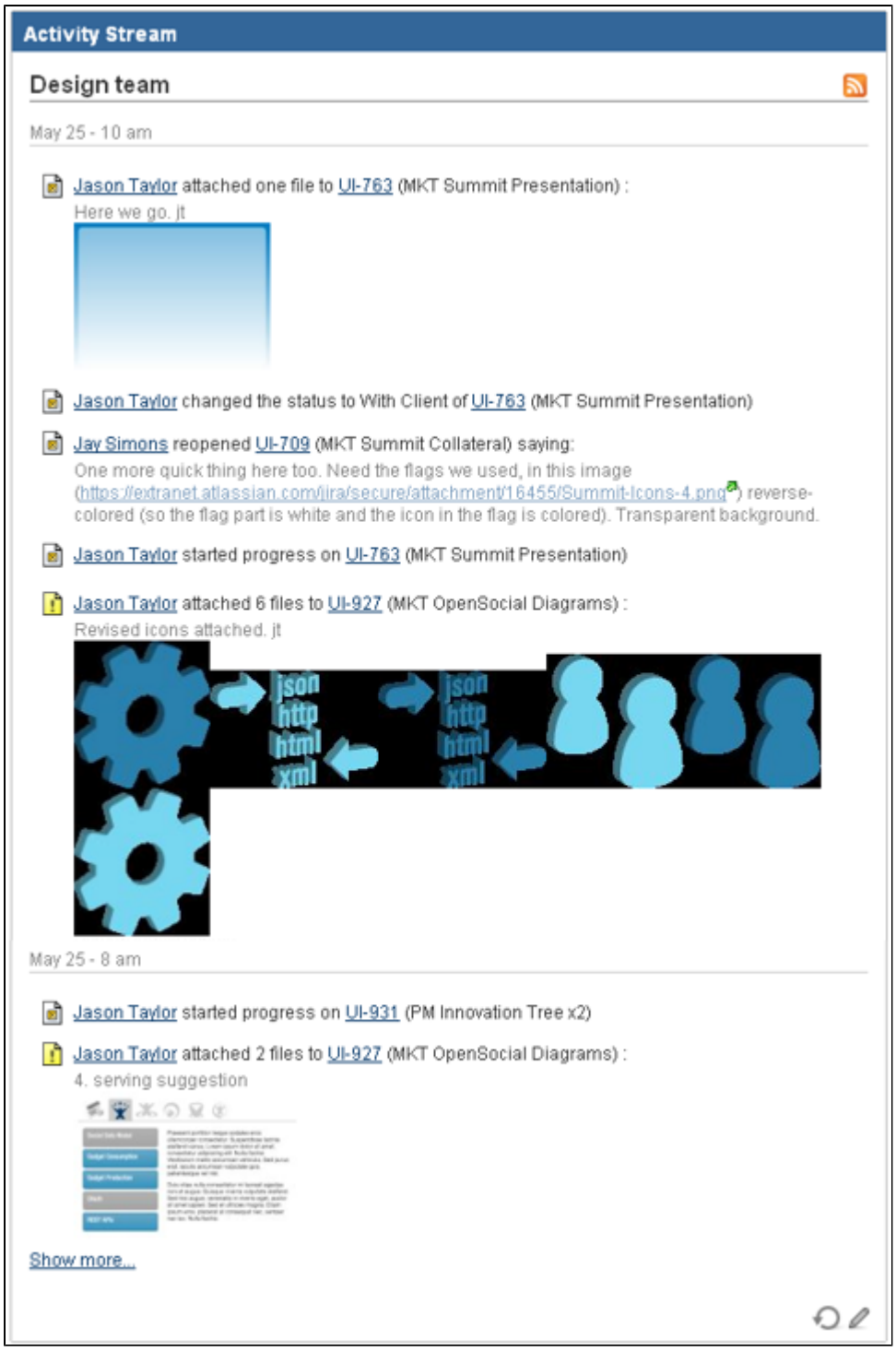
^Top

Activity Stream

The new '**Activity Stream**' gadget displays a summary of the latest activity in JIRA projects (and/or by particular people) in which you are interested.

The '**Activity Stream**' gadget also provides an RSS feed, allowing you to create very specific RSS feeds of only the information that is most relevant to you.

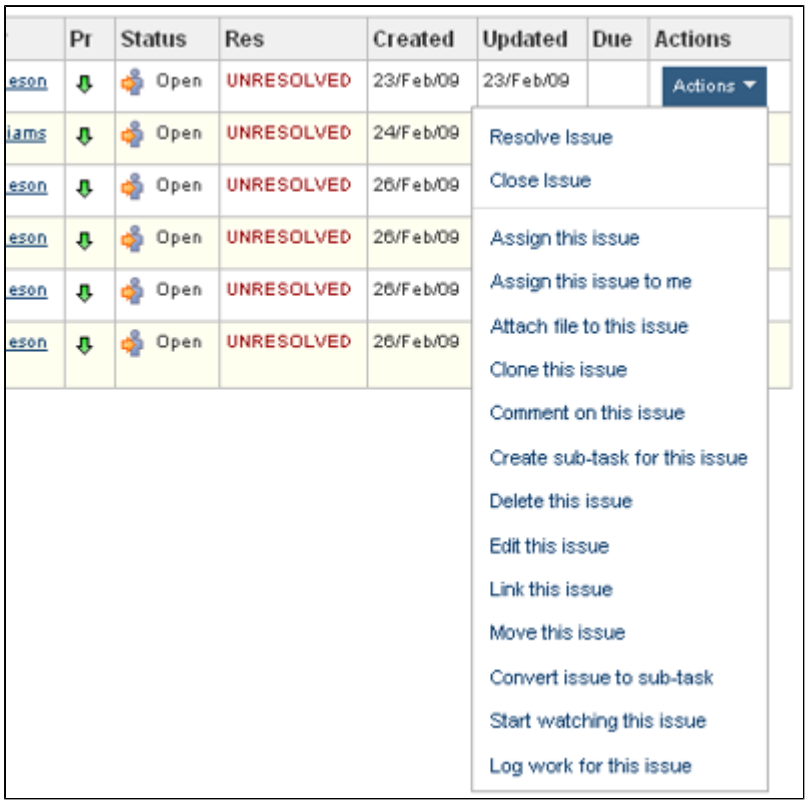
See the documentation for more details.



^Top

Issue Actions in the Issue Navigator

By popular request, issues can now be actioned directly from the Issue Navigator:

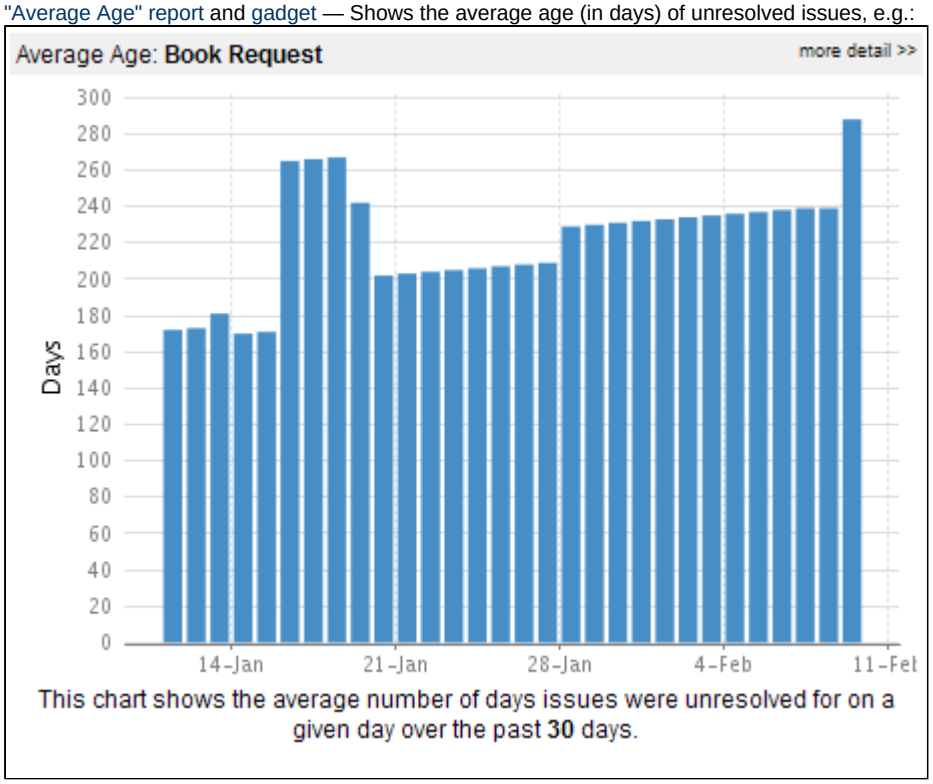


The "Actions" menu is also available for the list of sub-tasks within an issue.
^Top

Charting Now Comes Standard

The following reports and gadgets from the Charting plugin have now been integrated into JIRA:

- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over a given

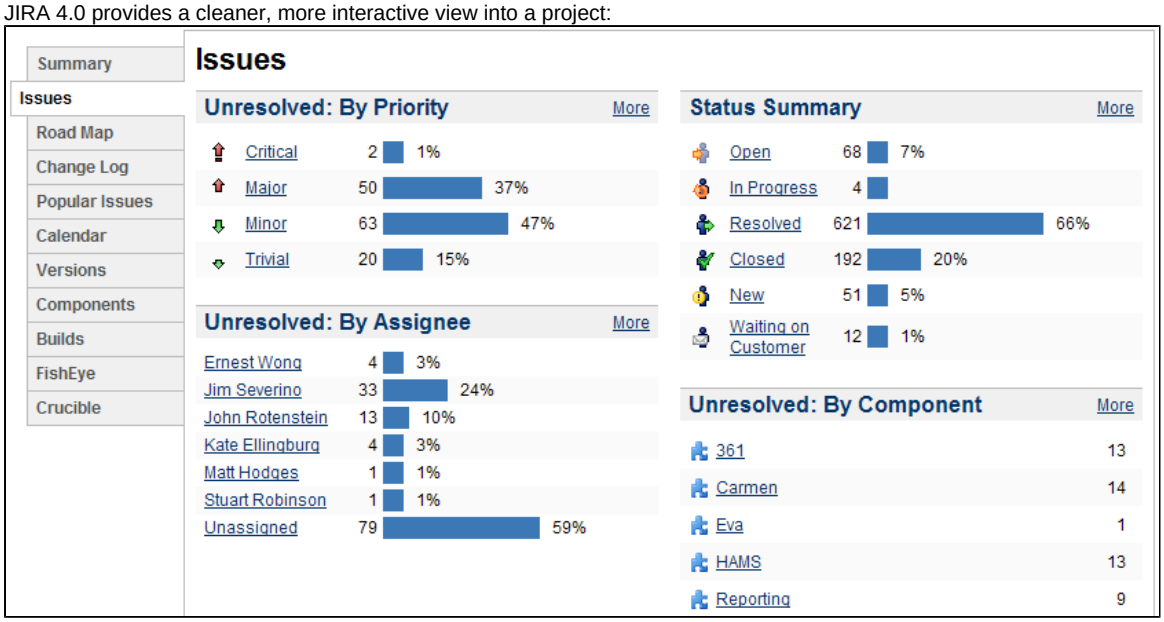
period of time.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.

Also, the "Resolved" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution date recorded.

^Top

New-look "Browse Project"

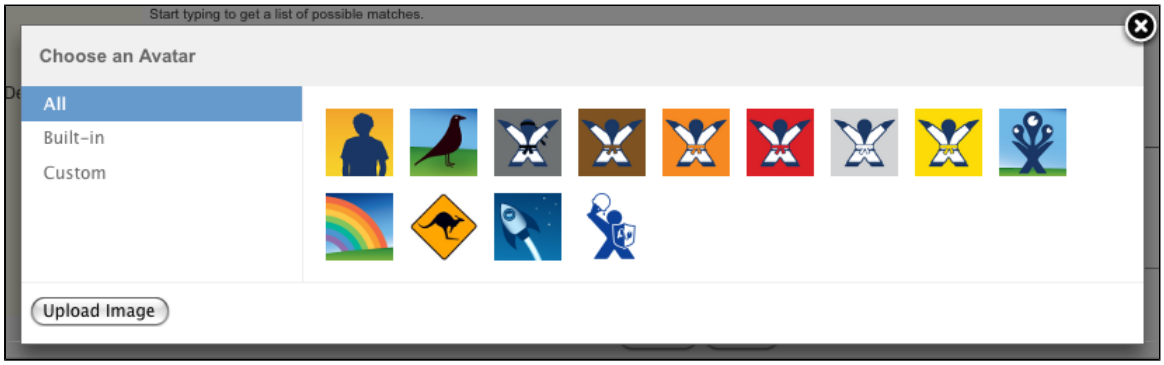
JIRA 4.0 provides a cleaner, more interactive view into a project:



See the documentation for more about browsing projects, versions and components.
^Top

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):

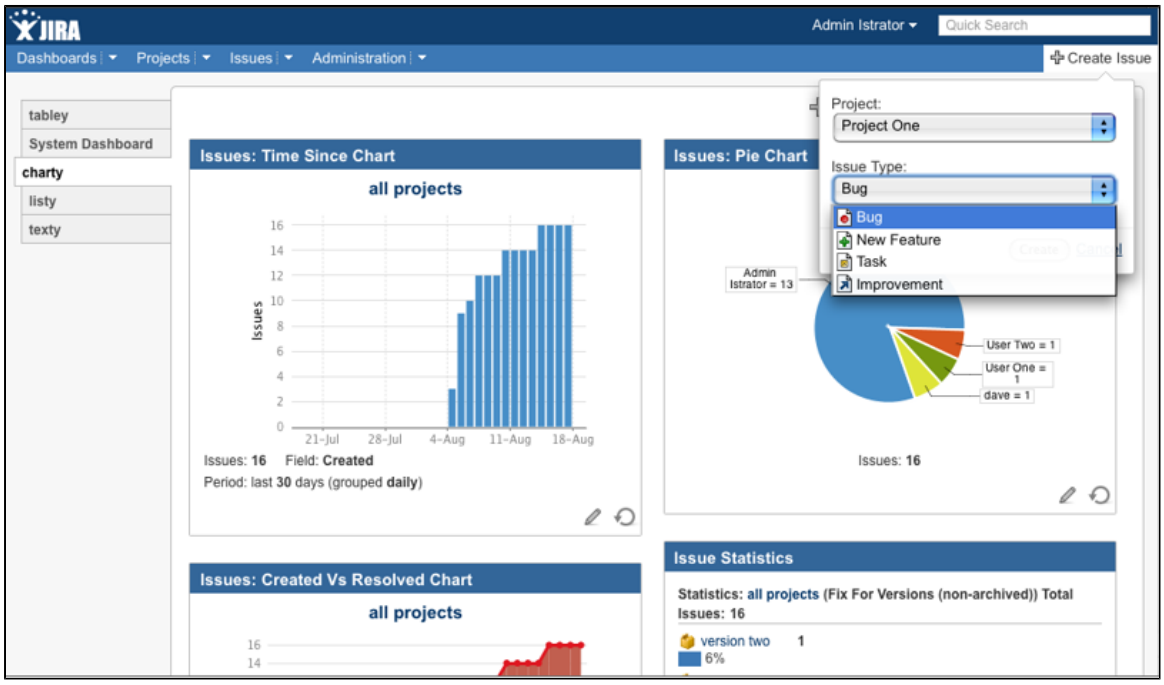


^Top

8

New-look Header

The new-look JIRA header gives you quick access to all the most commonly-used functions. Creating an issue just got super-fast!



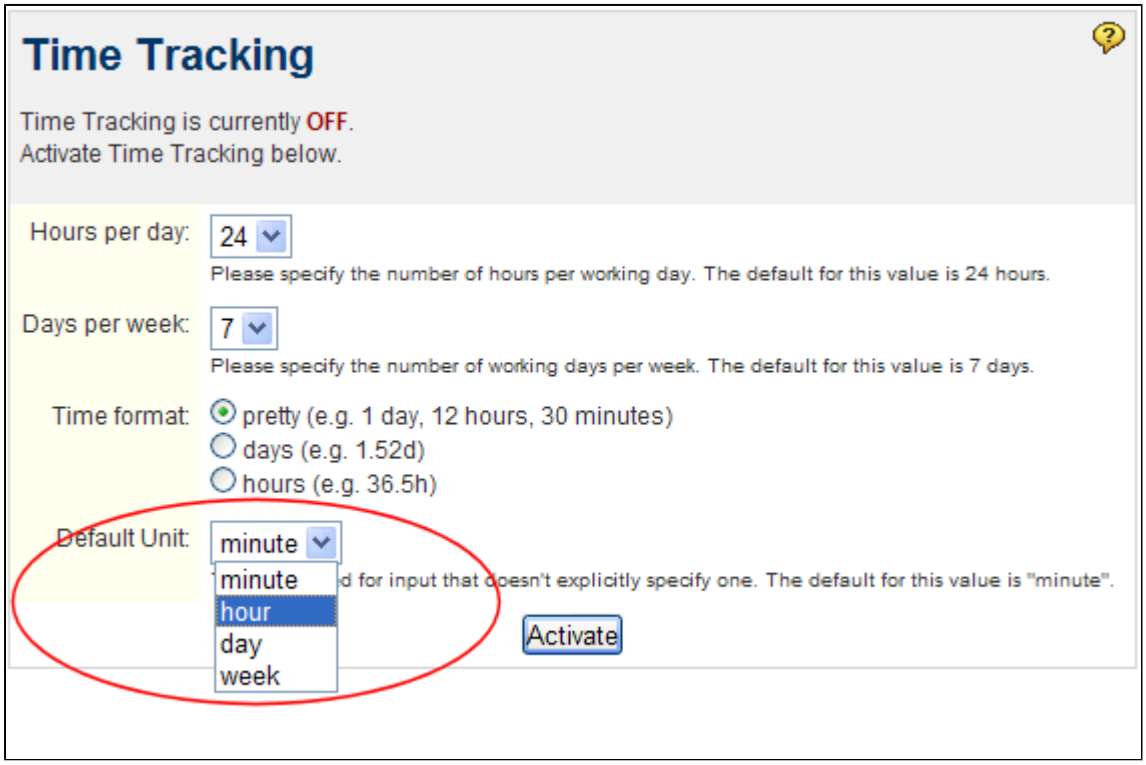
Click to zoom in:

If you prefer keystrokes rather than mouse-clicks, you'll be pleased to know that you can use your keyboard to navigate the new header menus.
^Top

9

Default Unit for Time Tracking

You can now specify your preferred **Default Unit** (minutes/hours/days/weeks) for your JIRA system. This will be applied whenever users log work on an issue without specifying a unit.



^Top

10

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.

^Top

Engine Room

Beyond the 'Back' Button

When navigating away from a page where you have modified data, you will be prompted to see if you would like to save the data or discard your changes (see JRA-14911).

Index Queue

Index updates are now put in a queue. So even if the update takes longer than 30 seconds, the operation remains on the queue and is not lost. (See JRA-14220.)

^Top

Plus over 800 other fixes and improvements

Click here for full list.

JIRA Issues (200 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-1538		Filter on Versions and Components across Projects		Resolved

JRA-1579	Create a portlet for the recent history	Resolved
JRA-2033	Add an RSS feed query for comments to individual issues	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-2810	Recently viewed issues	Resolved
JRA-2916	Allow Previous version searching	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-3206	View issues without an estimate	Resolved
JRA-3624	released/unreleased version filter	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-5383	My Votes and My Watches as filters	Resolved
JRA-7551	Provide capability to find issues by resolution date	Resolved
JRA-7626	Build search queries remotely	Resolved
JRA-7772	Ability to create advanced queries to search across all data	Resolved
JRA-8159	Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type	Resolved
JRA-8527	Put task actions directly in filter output	Resolved
JRA-8606	Need a way to find watched issues	Resolved
JRA-8852	Sort filter results by non-visible field	Resolved
JRA-8973	RSS of Project Changes	Resolved
JRA-9551	Search for all Sub-Tasks of one given issue	Resolved
JRA-9651	User Activity Log	Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"	Resolved
JRA-10443	"Not Assigned to User" criteria in filters	Resolved
JRA-10603	MultipleSelect searcher for cascading selection field	Resolved
JRA-12921	Ability to export Watched Issues to excel	Resolved
JRA-14613	Each project can have its own logo	Resolved
JRA-14616	Ability to query for issues that you are not watching	Resolved
JRA-14983	Fetch only updated or changed issues	Resolved

JRA-15575	Test and confim JIRA is compatible with Microsoft SQL Server 2008	Closed
JRA-16067	Provide field definition in XML issue view URL to customize XML view	Resolved
JRA-16120	Dashboard rewrite	Resolved
JRA-16509	Check for javascript enabled in browser	Resolved
JRA-16805	Convert legacy portlets to Gadgets	Resolved
JRA-16807	Convert Intro Portlet	Resolved
JRA-16808	Convert Quicklinks portlet	Resolved
JRA-16809	Convert Favourite Fitlers	Resolved
JRA-16811	Convert Pie Chart Portlet	Resolved
JRA-16903	Convert CreatedVsResolved Chart Portlet	Resolved
JRA-16905	Convert Average Age Chart	Resolved
JRA-16914	Convert Recently Created Portlet	Resolved
JRA-16916	Convert Time Since Chart	Resolved
JRA-16920	Add filter option for subtasks of a particular issue	Resolved
JRA-16926	Implement OAuth SPI in JIRA	Resolved
JRA-16931	Convert Saved Filter Portlet	Resolved
JRA-16978	Option "Number to Show" in Filter Statistics Portlet to limit numer of rows displayed	Resolved
JRA-17090	Convert Admin Portlet to gadget	Resolved
JRA-17094	Convert Bugzilla portlet	Resolved
JRA-17095	Convert Project/Projects/Project Table portlets into a gadget	Resolved
JRA-17096	Convert Project / Filter stats Portlet to gadget	Resolved
JRA-17101	Convert TwoDimensionalStatsPortlet to gadgets	Resolved
JRA-17112	Convert Roadmap Portlet to gadgets	Resolved
JRA-17133	Convert Resolution Time Chart to gadgets	Resolved
JRA-17140	Convert Assign To Me portlet to gadget	Resolved
JRA-17141	Convert Voted For Portlet to gadget	Resolved
JRA-17142	Convert Watching Portlet to gadget	Resolved

JRA-17143	Convert In-progress portlet to gadget	Resolved
JRA-17182	Allow REST plugins to be decorated	Resolved
JRA-17337	New Header for JIRA 4.0	Resolved
JRA-17355	Need provision to search multiple group	Resolved
JRA-17763	JQL: It will great if user can use E-mail ID's for searching in Assignee and reporter fields	Resolved
JRA-923	Allow filter by "No Fix For" across projects	Resolved
JRA-1560	Better support for logical operation (and/or/not) type of filters.	Resolved
JRA-1635	"not" qualifier on fields for searching	Resolved
JRA-1642	Create home directory instead of index & attachment directory	Resolved
JRA-1800	Improve the UI for browse project	Resolved
JRA-1844	Display attachment comments associated with their attachments	Resolved
JRA-1983	Enable filtering on "older than 1 month"	Resolved
JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-2469	It would be really nice to specficy several Asignee options in filters	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-2852	search for issues on version lower or equal to a given version	Resolved
JRA-3000	Add key NUMBER (only number) searching to default search filter.	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-3114	Request: add optional icon for each project	Resolved
JRA-3451	Enable filtering by Date Resolved	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-4227	Recent History Popup - persistance across sessions & more data	Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria	Resolved
JRA-4688	Browse Project: Within the tab panel, if components are hidden - the version info appears to be right aligned.	Resolved
JRA-5121	Filter Portlet with configurable columns	Resolved
JRA-5152	Show issue linked to another issue.	Resolved
JRA-5310	Watchlist should be exportable	Resolved

JRA-5435	Issue actions and operations on Issue Navigator	Resolved
JRA-5560	Improved query functionality	Resolved
JRA-5798	Project Portlet: needs multi project selection	Resolved
JRA-5844	allow multiple users when creating filters	Resolved
JRA-5965	Allow configure units of time tracking	Resolved
JRA-6010	Thought processing	Resolved
JRA-6170	Radio Buttons should support Select List Searcher template	Resolved
JRA-6180	Search for a custom field that is empty	Resolved
JRA-6527	Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-7068	Allow for list of issues to be saved as a filter	Resolved
JRA-7909	Search/ filter for "empty" fields	Resolved
JRA-8487	Bad logging from uk.ltd.getahead.dwr.util.CommonsLoggingOutput on startup	Resolved
JRA-8686	Allow searching of issues by Full Name for all user fields	Resolved
JRA-8758	Cannot create filter for multiple projects all issues in version "Released Versions"	Resolved
JRA-8806	Allow "Released" & "Unreleased" Version search accross multiple projects	Resolved
JRA-9115	Ability to search for issues with no due date associated	Resolved
JRA-9278	New Field "Resolution Date" automatically filled with date of setting resolution	Resolved
JRA-9658	Minor css bug (cursor)	Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.	Resolved
JRA-10405	Attachment ordering	Resolved
JRA-10492	Search for several users as Assignee or Reporter	Resolved
JRA-10644	Make filters more accessible	Resolved
JRA-10658	More columns on Dashboards	Resolved
JRA-11134	Allow setting of column order/sort with no issues in result set	Resolved
JRA-12177	Time tracking by using setting "hours" - edit issue shows "Original/Remaining Estimate" -field value in "pretty" mode	Resolved
JRA-12596	Enable cross-project filtering on special versions	Resolved
JRA-12656	Add paging/optimization for Change Log scope	Resolved

JRA-13426	Next/previous version links for 'Browse Version' screen	Resolved
JRA-13745	Clean up top toolbar by moving Profile link to username and removing Filters link	Resolved
JRA-13850	Servlet Content Listeners should implement the catch / log / rethrow pattern	Resolved
JRA-14220	Ensure the index optimize operation does not cause index lock timeouts	Resolved
JRA-14419	Warning for Websphere installation on validating entity-engine.xml	Closed
JRA-14513	JIRA Soap Service log and Access filter log footprint needs improving - Invoked Method would be handy	Resolved
JRA-14516	JIRA upgrade page should warn about possible character encoding issue if JIRA is moved between two servers.	Closed
JRA-14598	Add access key for administer project from browse project	Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype	Resolved
JRA-14826	Dashboards with a large number of portal pages cause the page to become too wide	Resolved
JRA-15018	Improved SOAP and HTTP access logging	Resolved
JRA-15266	Would prefer if the "Worklog" heading under "Operations" when viewing an issue were a link, like the rest of the operations.	Closed
JRA-15445	RPC plugin needs to be cleaned up	Resolved
JRA-15517	Upgrade JIRA to use the latest version of Lucene indexing framework - v2.3.2	Resolved
JRA-15543	Show release date next to version name in the list of versions on Browse Project screen	Resolved
JRA-15646	Convert JIRA to jQuery	Resolved
JRA-15649	Sort list of plugins in Admin section alphabetically	Resolved
JRA-15665	Address issue of plugins pushing filter/report panel off screen	Resolved
JRA-15666	Add project information to the issue XML view	Resolved
JRA-15700	Created VS Resolved cumulative + individual graphs' Y axis should should be independent	Resolved
JRA-15702	Migrate to licensing 2.0	Resolved
JRA-15732	Update email documentation to highlight that OutOfMemoryError can stop email processing	Closed
JRA-15846	Allow changing license on Data Import in new JIRA instance.	Resolved
JRA-15872	"Browse Project" URL doesn't include current project	Resolved
JRA-15886	Add logging notification for index optimization events	Resolved
JRA-15920	Include warning in EAR/WAR documentation not to edit anything directly on the application server	Closed
JRA-15962	Upgrade JIRA to Plugins 2.x	Resolved

JRA-15991	Merge translations files into one to make translating JIRA easier!	Resolved
JRA-16058	Aggressive locking in JiraCachingPropertySet causes high contention	Resolved
JRA-16113	Do not show negative values on Y axis in Created vs. Resolved chart	Resolved
JRA-16122	HTTP Basic auth should be enabled by default	Resolved
JRA-16138	Anonymous users should not be considered to "own" all anonymous comments.	Resolved
JRA-16210	Display issue count on JQL execution	Resolved
JRA-16211	Enter / return should execute JQL	Resolved
JRA-16253	Source Build documentation is out of date and incomplete	Closed
JRA-16276	Adjust colours of Resoltion date chart to be more distinshable for the colour blind	Resolved
JRA-16278	Add ability to search for versions using regex or similar	Resolved
JRA-16379	Weblogic Deployment descriptor (weblogic.xml) has changed for Weblogic 9.x	Closed
JRA-16424	log4j output should contain more information	Resolved
JRA-16443	Create a jira.field.resolution.include transition attribute	Resolved
JRA-16510	Update to jQuery 1.3.2	Resolved
JRA-16522	Searching according to multiple assignees should be provided.	Resolved
JRA-16661	No way to cancel the "Project avatar" dialog	Resolved
JRA-16698	Have the ability to log all SQL statements issued by JIRA and also have a callback for timing purposes	Resolved
JRA-16744	Improve the performance of checking if a user belongs to a particular group.	Resolved
JRA-16793	A new section "JIRA Configuration" is needed in System Info page	Resolved
JRA-16838	Ensure Save button is disabled after submitting on the chart popup	Resolved
JRA-16839	Add nicer dashboard tabs	Resolved
JRA-16846	Allow for Pluggable Decorators in JIRA	Resolved
JRA-16870	Improve multi-threaded liveness of FieldLayoutManager under load	Resolved
JRA-17025	Link to the JIRA Knowledge Base in the Apache docs	Closed
JRA-17128	Profiling document broken link	Closed
JRA-17215	Environment field is missing from the bulk operations screen	Resolved
JRA-17217	French translation for "Road Map"	Resolved

JRA-17296	Add attribute for hidding a gadget when not logged in	Resolved
JRA-17314	Need a way to use the WebResourceManager to include JS/CSS for project tab panels and portlets	Resolved
JRA-17324	Make browse project/component/version use cached url for project avatar	Resolved
JRA-17361	Style gadget mini-messages to be more atlassian like	Resolved
JRA-17391	JIRA_HOME path in windows needs to be specified using backslash(or as Mac) and not as default windows way. Needs to document it to avoid confusion.	Closed
JRA-17404	Improve logging/UI when plugins can't be loaded.	Resolved
JRA-17412	Ability for support to easily tell if a patch has been applied	Resolved
JRA-17429	JIRA should only persist a plugin as disabled if it was explicitly disabled by the user. If it got disabled because it was invalid, then JIRA should try to enable it on restart.	Resolved
JRA-17474	Choosing an non-existing drive as destination directory for windows installer give wrong error message	Resolved
JRA-17483	Need to add info about jira.home to README files in top directory	Closed
JRA-17496	re-write dashboard client-side	Closed
JRA-17515	Put the JQL slow log into its own file.	Resolved
JRA-17562	Performance Improvements for JIRA v4.0	Resolved
JRA-17674	Issue summary page: Please duplicate link "Manage Attachments" under section "Image Attachments"	Resolved
JRA-17682	Add a warning to the reports documentation.	Closed
JRA-17698	Remove dependency on backport.util.concurrent	Resolved
JRA-17756	Remove DWR from JIRA	Resolved
JRA-17779	JQL reserve words should be mention in documentation	Closed
JRA-17797	JQL: Some points to add in documentation of JQL	Closed
JRA-17916	Update JAC hardware spec on Requirements page	Closed
JRA-17950	Instructions for deleting an issue type are too vague, and docs do not further explain them	Closed
JRA-17978	Search sort order doesn't persist when searching on free text	Resolved
JRA-18017	Footer improvements - like Bamboo	Resolved
JRA-18073	Create issue button should be aware of the current project	Resolved
JRA-18101	LDAP Integration document is confusing about what order passwords will be checked.	Closed
JRA-18102	JQL: Error message can be improved for date field when user doesn't put quotes for value	Resolved
JRA-18105	add documentation on JQL "autocomplete"	Closed

JRA-18110	Tool tip text for "Manage dashboard" has unecessary spaces between words	Resolved
JRA-18133	Upgrade to the latest version of Seraph and Trusted Applications library	Resolved
JRA-18145	In order for charts to appear in the charting popup, they should be in both categories JIRA and Charts	Resolved
JRA-18147	on "General Config" screen, change "JQL Auto-complete Disabled" to just "JQL Auto-complete" (as per "Issue Picker Auto-complete")	Resolved
JRA-18165	JIRA 4.0 GUI comments	Resolved
JRA-18225	It would be cool to have the operations (save, save as, etc) in the advanced navigator view so you do not have to go to the view tab	Resolved
JRA-18228	Per-user JQL autocomplete setting	Resolved
JRA-18232	It will be great if we can limit the number of favourite filters in issue drop down to 5	Resolved
JRA-18236	JQL History links should be middle clickable	Resolved
JRA-18237	The error message 'Query is too complex to display in the Issue Navigator edit controls' should be changed to "Query is too complex to display in Simple mode"	Resolved

^Top

JIRA 4.0 Beta 3 Release Notes

August 27, 2009

JIRA 4.0 Beta 3 is a public development release leading up to **JIRA 4.0**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

It is not possible to upgrade JIRA 4.0 Beta 1 or 2 data to Beta 3. The data needs to come from an already released JIRA version (for example, JIRA 3.13.5).

The **only** plugin that is compatible with JIRA 4.0 Beta is the latest JIRA Toolkit. Do not install any other plugins.

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

Highlights of JIRA 4.0 Beta 3:

- Advanced Searching
- Dashboard Gadgets
- Activity Stream

- Issue Actions in the Issue Navigator
- Charting Now Comes Standard
- New-look "Browse Project"
- Project Icons
- New-look Header
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- Plus over 450 other fixes and improvements

Thank you for your interest in JIRA 4.0 Beta 3

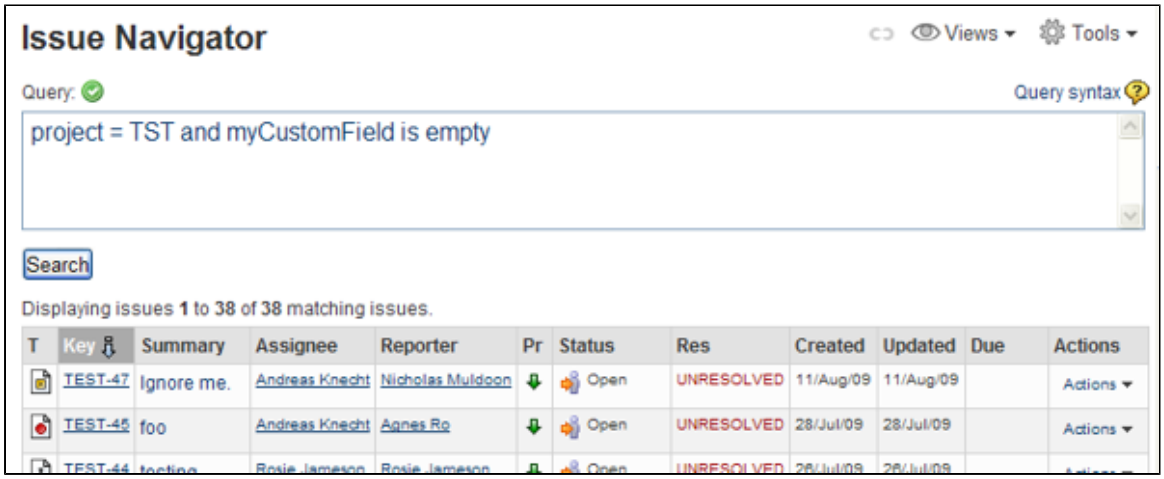
Download Beta

Installing/Upgrading to JIRA 4.0
JIRA 4.0 Beta 3 can be downloaded here. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide. You will also need to go to my.atlassian.com and generate a "**JIRA 4 Evaluation**" license, as any existing 3.x license files will not work with 4.0 Beta 3.

Highlights of JIRA 4.0 Beta 3

Advanced Searching

The new advanced search (JQL) provides support for logical operations, including AND, OR, NOT, NULL, EMPTY --- even on custom fields:



For more on the new JQL search syntax, please see the documentation.

^Top

Dashboard Gadgets

The new-look JIRA dashboard not only looks awesome: it now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

You can easily customise your dashboard by choosing a different layout, adding more gadgets, dragging the gadgets into different positions, and changing the look of individual gadgets.

What's happened to your favourite JIRA portlets? Don't worry: every portlet that previously shipped with JIRA has been converted to a 'legacy gadget'. And if you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.

The following gadgets are available in Beta 3:

- 'Activity Stream' gadget (see below)
- 'Admin' gadget
- 'Bamboo Plan Summary' gadget*
- 'Bamboo Status' gadget*
- 'Crucible Charting' gadget
- 'Create Issue' gadget
- 'Favourite Filters' gadget
- 'Filter Results' gadget
- 'FishEye Charting' gadget
- 'Introduction' gadget
- 'Issue Completed This Iteration' gadget
- 'Login' gadget
- 'Quicklinks' gadget
- 'Voted Issues' gadget

* Requires Bamboo 2.3.2 Beta 1 or later.

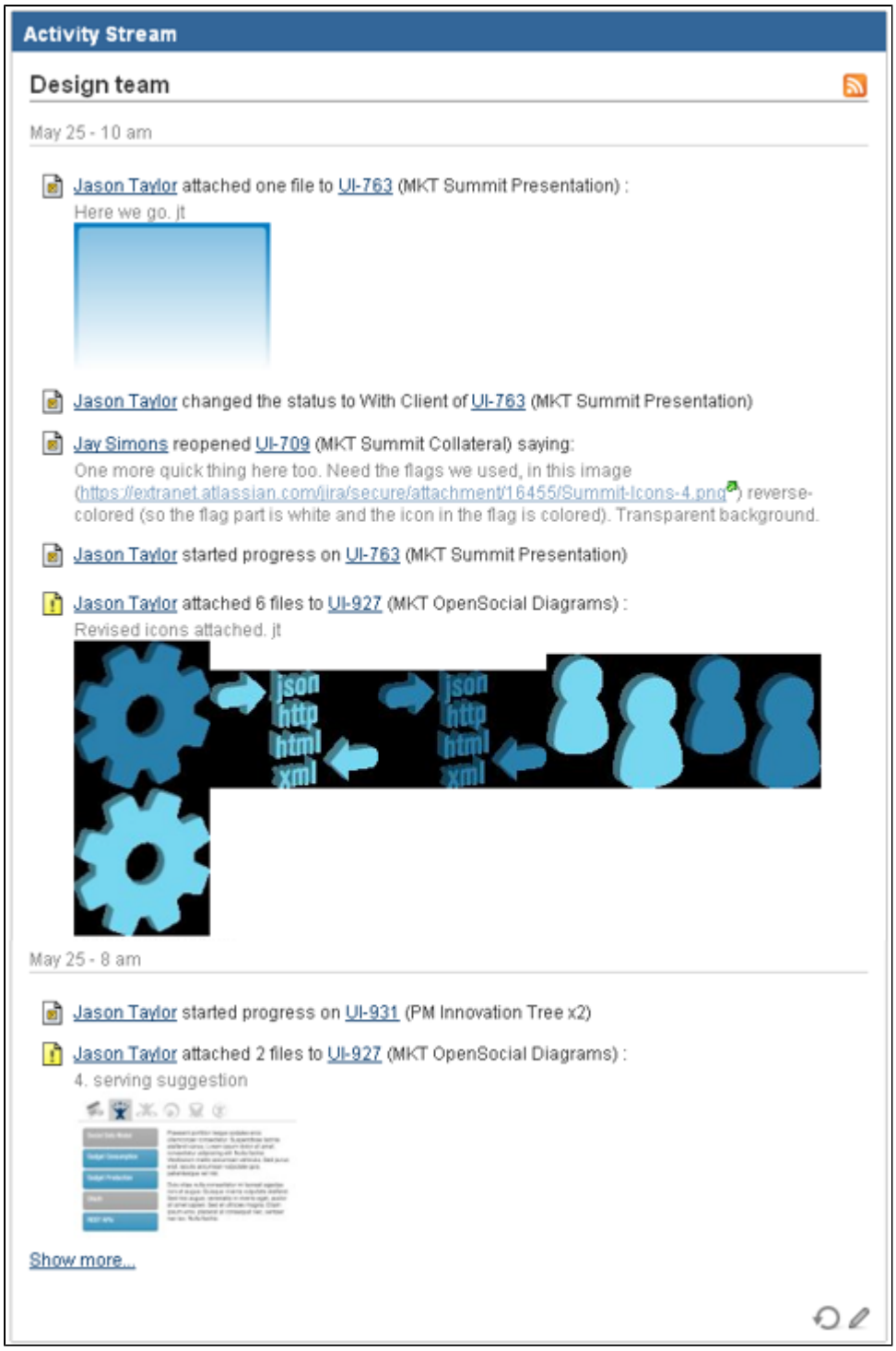
^Top

Activity Stream

The new '**Activity Stream**' gadget displays a summary of the latest activity in JIRA projects (and/or by particular people) in which you are interested.

The '**Activity Stream**' gadget also provides an RSS feed, allowing you to create very specific RSS feeds of only the information that is most relevant to you.

See the documentation for more details.

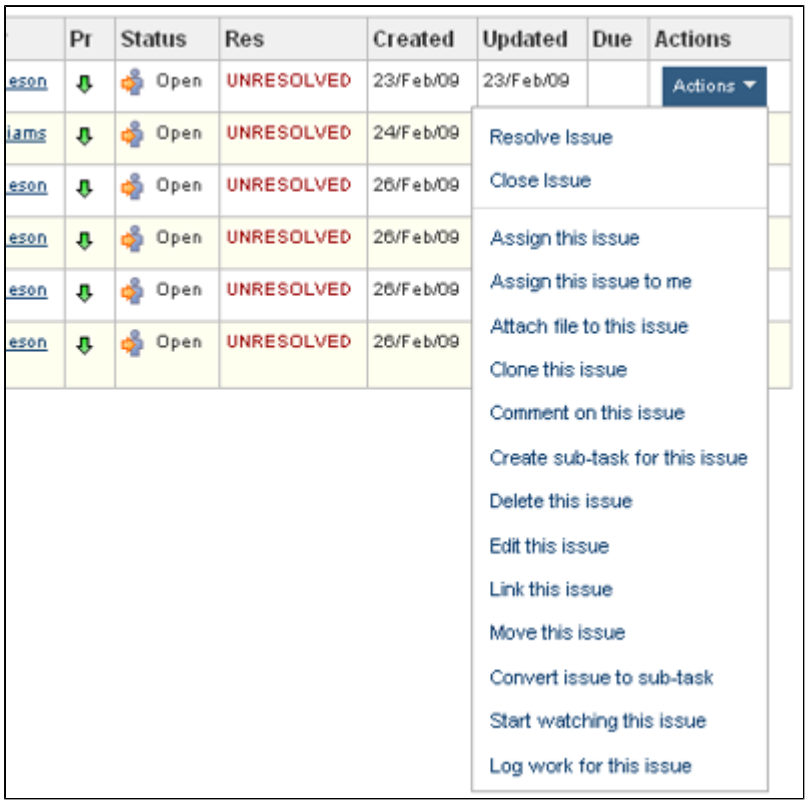


^Top

4

Issue Actions in the Issue Navigator

By popular request, issues can now be actioned directly from the Issue Navigator:

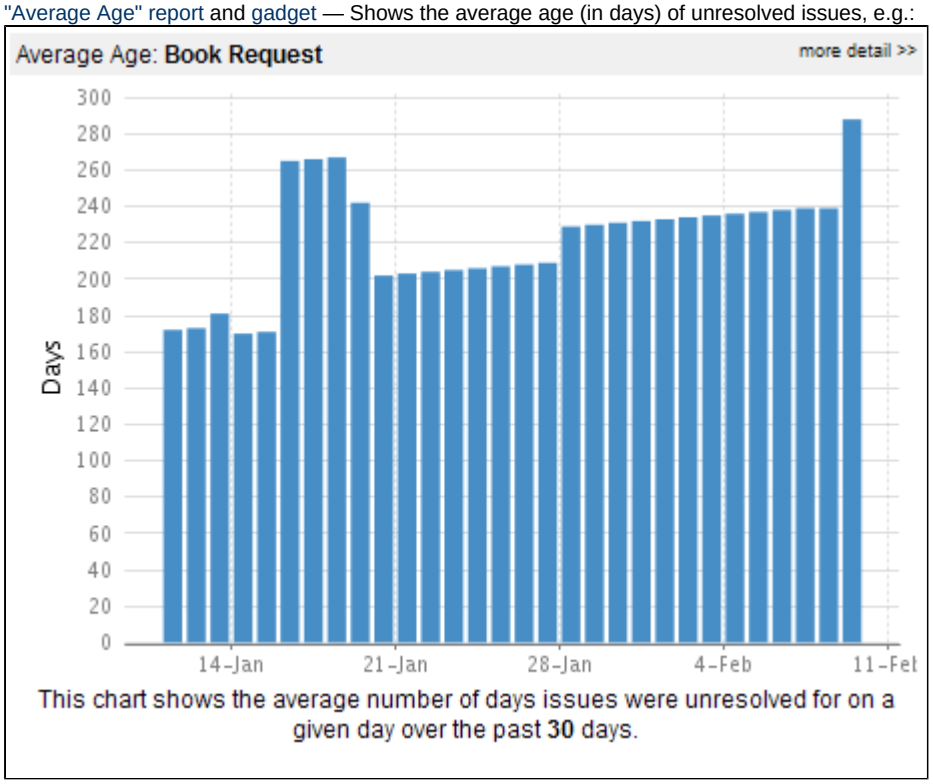


The "Actions" menu is also available for the list of sub-tasks within an issue.
^Top

Charting Now Comes Standard

The following reports and gadgets from the Charting plugin have now been integrated into JIRA:

- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over a given

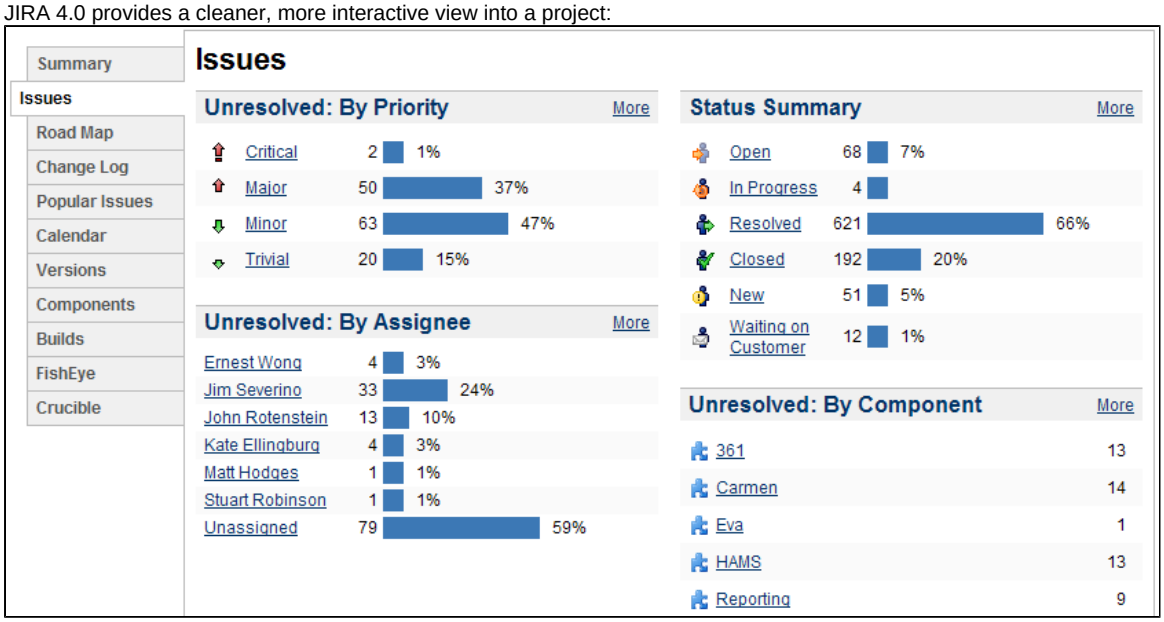
period of time.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.

Also, the "Resolved" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution date recorded.

^Top

New-look "Browse Project"

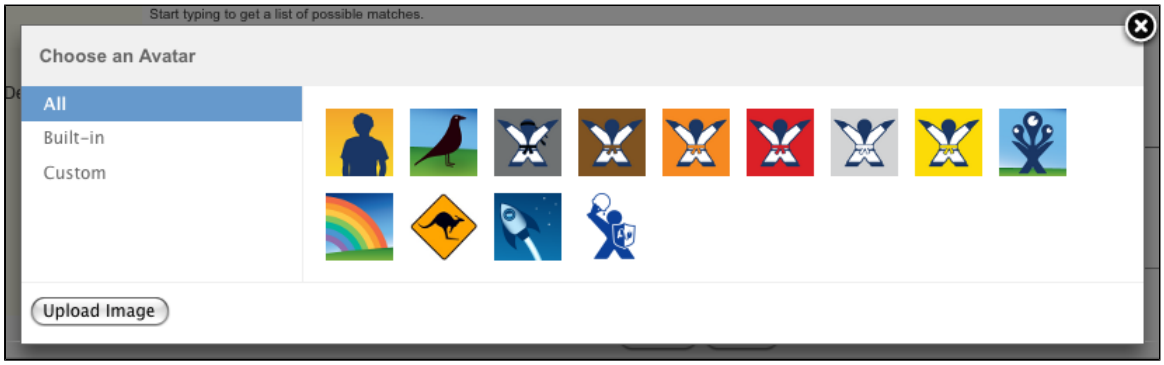
JIRA 4.0 provides a cleaner, more interactive view into a project:



See the documentation for more about browsing projects, versions and components.
^Top

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):



^Top

8

New-look Header

The new-look JIRA header gives you quick access to all the most commonly-used functions. Creating an issue just got super-fast!

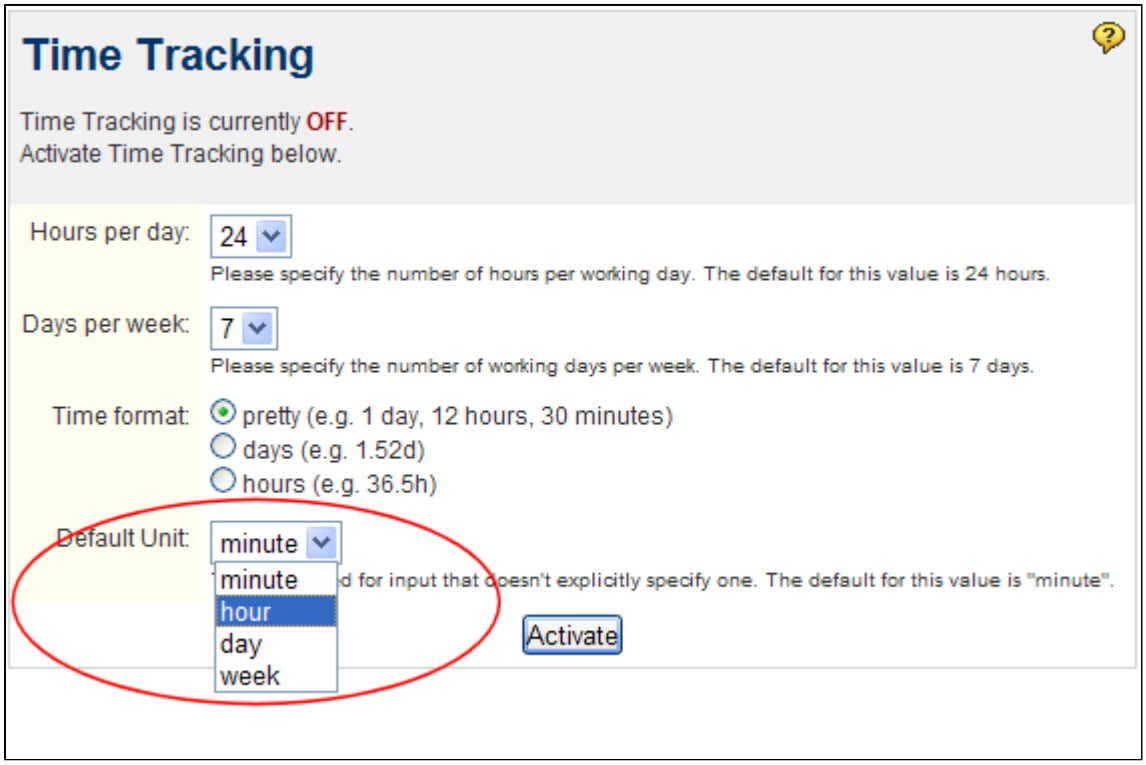
Click to zoom in:

If you prefer keystrokes rather than mouse-clicks, you'll be pleased to know that you can use your keyboard to navigate the new header menus.
^Top

9

Default Unit for Time Tracking

You can now specify your preferred **Default Unit** (minutes/hours/days/weeks) for your JIRA system. This will be applied whenever users log work on an issue without specifying a unit.



^Top

10

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.

^Top

Engine Room

Beyond the 'Back' Button

When navigating away from a page where you have modified data, you will be prompted to see if you would like to save the data or discard your changes (see JRA-14911).

Index Queue

Index updates are now put in a queue. So even if the update takes longer than 30 seconds, the operation remains on the queue and is not lost. (See JRA-14220.)

^Top

Plus over 450 other fixes and improvements

Click here for full list.

JIRA Issues (200 issues)

Key	Type	Summary	Priority	Status
JRA-1538		Filter on Versions and Components across Projects		Resolved

JRA-1579	Create a portlet for the recent history	Resolved
JRA-2033	Add an RSS feed query for comments to individual issues	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-2810	Recently viewed issues	Resolved
JRA-2916	Allow Previous version searching	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-3206	View issues without an estimate	Resolved
JRA-3624	released/unreleased version filter	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-5383	My Votes and My Watches as filters	Resolved
JRA-7551	Provide capability to find issues by resolution date	Resolved
JRA-7626	Build search queries remotely	Resolved
JRA-7772	Ability to create advanced queries to search across all data	Resolved
JRA-8159	Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type	Resolved
JRA-8527	Put task actions directly in filter output	Resolved
JRA-8606	Need a way to find watched issues	Resolved
JRA-8852	Sort filter results by non-visible field	Resolved
JRA-8973	RSS of Project Changes	Resolved
JRA-9551	Search for all Sub-Tasks of one given issue	Resolved
JRA-9651	User Activity Log	Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"	Resolved
JRA-10443	"Not Assigned to User" criteria in filters	Resolved
JRA-10603	MultipleSelect searcher for cascading selection field	Resolved
JRA-12921	Ability to export Watched Issues to excel	Resolved
JRA-14613	Each project can have its own logo	Resolved
JRA-14616	Ability to query for issues that you are not watching	Resolved
JRA-14983	Fetch only updated or changed issues	Resolved

JRA-15575	Test and confim JIRA is compatible with Microsoft SQL Server 2008	Closed
JRA-16067	Provide field definition in XML issue view URL to customize XML view	Resolved
JRA-16120	Dashboard rewrite	Resolved
JRA-16509	Check for javascript enabled in browser	Resolved
JRA-16805	Convert legacy portlets to Gadgets	Resolved
JRA-16807	Convert Intro Portlet	Resolved
JRA-16808	Convert Quicklinks portlet	Resolved
JRA-16809	Convert Favourite Fitlers	Resolved
JRA-16811	Convert Pie Chart Portlet	Resolved
JRA-16903	Convert CreatedVsResolved Chart Portlet	Resolved
JRA-16905	Convert Average Age Chart	Resolved
JRA-16914	Convert Recently Created Portlet	Resolved
JRA-16916	Convert Time Since Chart	Resolved
JRA-16920	Add filter option for subtasks of a particular issue	Resolved
JRA-16926	Implement OAuth SPI in JIRA	Resolved
JRA-16931	Convert Saved Filter Portlet	Resolved
JRA-16978	Option "Number to Show" in Filter Statistics Portlet to limit numer of rows displayed	Resolved
JRA-17090	Convert Admin Portlet to gadget	Resolved
JRA-17094	Convert Bugzilla portlet	Resolved
JRA-17095	Convert Project/Projects/Project Table portlets into a gadget	Resolved
JRA-17096	Convert Project / Filter stats Portlet to gadget	Resolved
JRA-17101	Convert TwoDimensionalStatsPortlet to gadgets	Resolved
JRA-17112	Convert Roadmap Portlet to gadgets	Resolved
JRA-17133	Convert Resolution Time Chart to gadgets	Resolved
JRA-17140	Convert Assign To Me portlet to gadget	Resolved
JRA-17141	Convert Voted For Portlet to gadget	Resolved
JRA-17142	Convert Watching Portlet to gadget	Resolved

JRA-17143	Convert In-progress portlet to gadget	Resolved
JRA-17182	Allow REST plugins to be decorated	Resolved
JRA-17337	New Header for JIRA 4.0	Resolved
JRA-17355	Need provision to search multiple group	Resolved
JRA-17763	JQL: It will great if user can use E-mail ID's for searching in Assignee and reporter fields	Resolved
JRA-923	Allow filter by "No Fix For" across projects	Resolved
JRA-1560	Better support for logical operation (and/or/not) type of filters.	Resolved
JRA-1635	"not" qualifier on fields for searching	Resolved
JRA-1642	Create home directory instead of index & attachment directory	Resolved
JRA-1800	Improve the UI for browse project	Resolved
JRA-1844	Display attachment comments associated with their attachments	Resolved
JRA-1983	Enable filtering on "older than 1 month"	Resolved
JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-2469	It would be really nice to specficy several Asignee options in filters	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-2852	search for issues on version lower or equal to a given version	Resolved
JRA-3000	Add key NUMBER (only number) searching to default search filter.	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-3114	Request: add optional icon for each project	Resolved
JRA-3451	Enable filtering by Date Resolved	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-4227	Recent History Popup - persistance across sessions & more data	Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria	Resolved
JRA-4688	Browse Project: Within the tab panel, if components are hidden - the version info appears to be right aligned.	Resolved
JRA-5121	Filter Portlet with configurable columns	Resolved
JRA-5152	Show issue linked to another issue.	Resolved
JRA-5310	Watchlist should be exportable	Resolved

JRA-5435	Issue actions and operations on Issue Navigator	Resolved
JRA-5560	Improved query functionality	Resolved
JRA-5798	Project Portlet: needs multi project selection	Resolved
JRA-5844	allow multiple users when creating filters	Resolved
JRA-5965	Allow configure units of time tracking	Resolved
JRA-6010	Thought processing	Resolved
JRA-6170	Radio Buttons should support Select List Searcher template	Resolved
JRA-6180	Search for a custom field that is empty	Resolved
JRA-6527	Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-7068	Allow for list of issues to be saved as a filter	Resolved
JRA-7909	Search/ filter for "empty" fields	Resolved
JRA-8487	Bad logging from uk.ltd.getahead.dwr.util.CommonsLoggingOutput on startup	Resolved
JRA-8686	Allow searching of issues by Full Name for all user fields	Resolved
JRA-8758	Cannot create filter for multiple projects all issues in version "Released Versions"	Resolved
JRA-8806	Allow "Released" & "Unreleased" Version search accross multiple projects	Resolved
JRA-9115	Ability to search for issues with no due date associated	Resolved
JRA-9278	New Field "Resolution Date" automatically filled with date of setting resolution	Resolved
JRA-9658	Minor css bug (cursor)	Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.	Resolved
JRA-10405	Attachment ordering	Resolved
JRA-10492	Search for several users as Assignee or Reporter	Resolved
JRA-10644	Make filters more accessible	Resolved
JRA-10658	More columns on Dashboards	Resolved
JRA-11134	Allow setting of column order/sort with no issues in result set	Resolved
JRA-12177	Time tracking by using setting "hours" - edit issue shows "Original/Remaining Estimate" -field value in "pretty" mode	Resolved
JRA-12596	Enable cross-project filtering on special versions	Resolved
JRA-12656	Add paging/optimization for Change Log scope	Resolved

JRA-13426	Next/previous version links for 'Browse Version' screen	Resolved
JRA-13745	Clean up top toolbar by moving Profile link to username and removing Filters link	Resolved
JRA-13850	Servlet Content Listeners should implement the catch / log / rethrow pattern	Resolved
JRA-14220	Ensure the index optimize operation does not cause index lock timeouts	Resolved
JRA-14419	Warning for Websphere installation on validating entity-engine.xml	Closed
JRA-14513	JIRA Soap Service log and Access filter log footprint needs improving - Invoked Method would be handy	Resolved
JRA-14516	JIRA upgrade page should warn about possible character encoding issue if JIRA is moved between two servers.	Closed
JRA-14598	Add access key for administer project from browse project	Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype	Resolved
JRA-14826	Dashboards with a large number of portal pages cause the page to become too wide	Resolved
JRA-15018	Improved SOAP and HTTP access logging	Resolved
JRA-15266	Would prefer if the "Worklog" heading under "Operations" when viewing an issue were a link, like the rest of the operations.	Closed
JRA-15445	RPC plugin needs to be cleaned up	Resolved
JRA-15517	Upgrade JIRA to use the latest version of Lucene indexing framework - v2.3.2	Resolved
JRA-15543	Show release date next to version name in the list of versions on Browse Project screen	Resolved
JRA-15646	Convert JIRA to jQuery	Resolved
JRA-15649	Sort list of plugins in Admin section alphabetically	Resolved
JRA-15665	Address issue of plugins pushing filter/report panel off screen	Resolved
JRA-15666	Add project information to the issue XML view	Resolved
JRA-15700	Created VS Resolved cumulative + individual graphs' Y axis should should be independent	Resolved
JRA-15702	Migrate to licensing 2.0	Resolved
JRA-15732	Update email documentation to highlight that OutOfMemoryError can stop email processing	Closed
JRA-15846	Allow changing license on Data Import in new JIRA instance.	Resolved
JRA-15872	"Browse Project" URL doesn't include current project	Resolved
JRA-15886	Add logging notification for index optimization events	Resolved
JRA-15920	Include warning in EAR/WAR documentation not to edit anything directly on the application server	Closed
JRA-15962	Upgrade JIRA to Plugins 2.x	Resolved

JRA-15991	Merge translations files into one to make translating JIRA easier!	Resolved
JRA-16058	Aggressive locking in JiraCachingPropertySet causes high contention	Resolved
JRA-16113	Do not show negative values on Y axis in Created vs. Resolved chart	Resolved
JRA-16122	HTTP Basic auth should be enabled by default	Resolved
JRA-16138	Anonymous users should not be considered to "own" all anonymous comments.	Resolved
JRA-16210	Display issue count on JQL execution	Resolved
JRA-16211	Enter / return should execute JQL	Resolved
JRA-16253	Source Build documentation is out of date and incomplete	Closed
JRA-16276	Adjust colours of Resoltion date chart to be more distinshable for the colour blind	Resolved
JRA-16278	Add ability to search for versions using regex or similar	Resolved
JRA-16379	Weblogic Deployment descriptor (weblogic.xml) has changed for Weblogic 9.x	Closed
JRA-16424	log4j output should contain more information	Resolved
JRA-16443	Create a jira.field.resolution.include transition attribute	Resolved
JRA-16510	Update to jQuery 1.3.2	Resolved
JRA-16522	Searching according to multiple assignees should be provided.	Resolved
JRA-16661	No way to cancel the "Project avatar" dialog	Resolved
JRA-16698	Have the ability to log all SQL statements issued by JIRA and also have a callback for timing purposes	Resolved
JRA-16744	Improve the performance of checking if a user belongs to a particular group.	Resolved
JRA-16793	A new section "JIRA Configuration" is needed in System Info page	Resolved
JRA-16838	Ensure Save button is disabled after submitting on the chart popup	Resolved
JRA-16839	Add nicer dashboard tabs	Resolved
JRA-16846	Allow for Pluggable Decorators in JIRA	Resolved
JRA-16870	Improve multi-threaded liveness of FieldLayoutManager under load	Resolved
JRA-17025	Link to the JIRA Knowledge Base in the Apache docs	Closed
JRA-17128	Profiling document broken link	Closed
JRA-17215	Environment field is missing from the bulk operations screen	Resolved
JRA-17217	French translation for "Road Map"	Resolved

JRA-17296	Add attribute for hidding a gadget when not logged in	Resolved
JRA-17314	Need a way to use the WebResourceManager to include JS/CSS for project tab panels and portlets	Resolved
JRA-17324	Make browse project/component/version use cached url for project avatar	Resolved
JRA-17361	Style gadget mini-messages to be more atlassian like	Resolved
JRA-17391	JIRA_HOME path in windows needs to be specified using backslash(or as Mac) and not as default windows way. Needs to document it to avoid confusion.	Closed
JRA-17404	Improve logging/UI when plugins can't be loaded.	Resolved
JRA-17412	Ability for support to easily tell if a patch has been applied	Resolved
JRA-17429	JIRA should only persist a plugin as disabled if it was explicitly disabled by the user. If it got disabled because it was invalid, then JIRA should try to enable it on restart.	Resolved
JRA-17474	Choosing an non-existing drive as destination directory for windows installer give wrong error message	Resolved
JRA-17483	Need to add info about jira.home to README files in top directory	Closed
JRA-17496	re-write dashboard client-side	Closed
JRA-17515	Put the JQL slow log into its own file.	Resolved
JRA-17562	Performance Improvements for JIRA v4.0	Resolved
JRA-17674	Issue summary page: Please duplicate link "Manage Attachments" under section "Image Attachments"	Resolved
JRA-17682	Add a warning to the reports documentation.	Closed
JRA-17698	Remove dependency on backport.util.concurrent	Resolved
JRA-17756	Remove DWR from JIRA	Resolved
JRA-17779	JQL reserve words should be mention in documentation	Closed
JRA-17797	JQL: Some points to add in documentation of JQL	Closed
JRA-17916	Update JAC hardware spec on Requirements page	Closed
JRA-17950	Instructions for deleting an issue type are too vague, and docs do not further explain them	Closed
JRA-17978	Search sort order doesn't persist when searching on free text	Resolved
JRA-18017	Footer improvements - like Bamboo	Resolved
JRA-18073	Create issue button should be aware of the current project	Resolved
JRA-18101	LDAP Integration document is confusing about what order passwords will be checked.	Closed
JRA-18102	JQL: Error message can be improved for date field when user doesn't put quotes for value	Resolved
JRA-18105	add documentation on JQL "autocomplete"	Closed

JRA-18110	Tool tip text for "Manage dashboard" has unecessary spaces between words	Resolved
JRA-18133	Upgrade to the latest version of Seraph and Trusted Applications library	Resolved
JRA-18145	In order for charts to appear in the charting popup, they should be in both categories JIRA and Charts	Resolved
JRA-18147	on "General Config" screen, change "JQL Auto-complete Disabled" to just "JQL Auto-complete" (as per "Issue Picker Auto-complete")	Resolved
JRA-18165	JIRA 4.0 GUI comments	Resolved
JRA-18225	It would be cool to have the operations (save, save as, etc) in the advanced navigator view so you do not have to go to the view tab	Resolved
JRA-18228	Per-user JQL autocomplete setting	Resolved
JRA-18232	It will be great if we can limit the number of favourite filters in issue drop down to 5	Resolved
JRA-18236	JQL History links should be middle clickable	Resolved
JRA-18237	The error message 'Query is too complex to display in the Issue Navigator edit controls' should be changed to "Query is too complex to display in Simple mode"	Resolved

^Top

JIRA 4.0 Beta 2 Release Notes

July 23, 2009

JIRA 4.0 Beta 2 is a public development release leading up to **JIRA 4.0**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

It is not possible to upgrade JIRA 4.0 Beta 1 data to Beta 2. The data needs to come from an already released JIRA version (for example, JIRA 3.13.5).

The **only** plugin that is compatible with JIRA 4.0 Beta is the latest JIRA Toolkit. Do not install any other plugins.

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

Highlights of JIRA 4.0 Beta 2:

- Advanced Searching
- Dashboard Gadgets
- Activity Stream

- Issue Actions in the Issue Navigator
- Charting Now Comes Standard
- New-look "Browse Project"
- Project Icons
- New-look Header
- Default Unit for Time Tracking
- "History" is now permanent
- Engine Room
- Plus over 250 other fixes and improvements

Thank you for your interest in JIRA 4.0 Beta 2

Download Beta

Known Issues

Upgrading to JIRA 4.0
JIRA 4.0 Beta can be downloaded here. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide.

Known Issues

- **Supported browsers:** For optimal experience with JIRA 4.0 Beta 2, please use Firefox 3.x. Support for other browsers will be added prior to the launch of JIRA 4.0.
- **Portlet-to-gadget upgrade task:** This has not yet been implemented. Your existing portlets will be displayed in legacy mode.

^Top

Highlights of JIRA 4.0 Beta 2

Advanced Searching

The new advanced search (JQL) provides support for logical operations (including AND/OR/NOT/NULL, even on custom fields) when filtering issues:

For more on the new JQL search syntax, please see the documentation.

^Top

Dashboard Gadgets

The new-look JIRA dashboard now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

What's happened to your favourite JIRA portlets? Don't worry: every portlet that previously shipped with JIRA has been converted to a 'legacy gadget'. And if you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.

The following gadgets are available in Beta 2:

- 'Activity Stream' gadget (see below)
- 'Admin' gadget
- 'Create Issue' gadget
- 'Favourite Filters' gadget
- 'Filter Results' gadget
- 'Introduction' gadget
- 'Issue Completed This Iteration' gadget
- 'Login' gadget
- 'Quicklinks' gadget
- 'Voted Issues' gadget

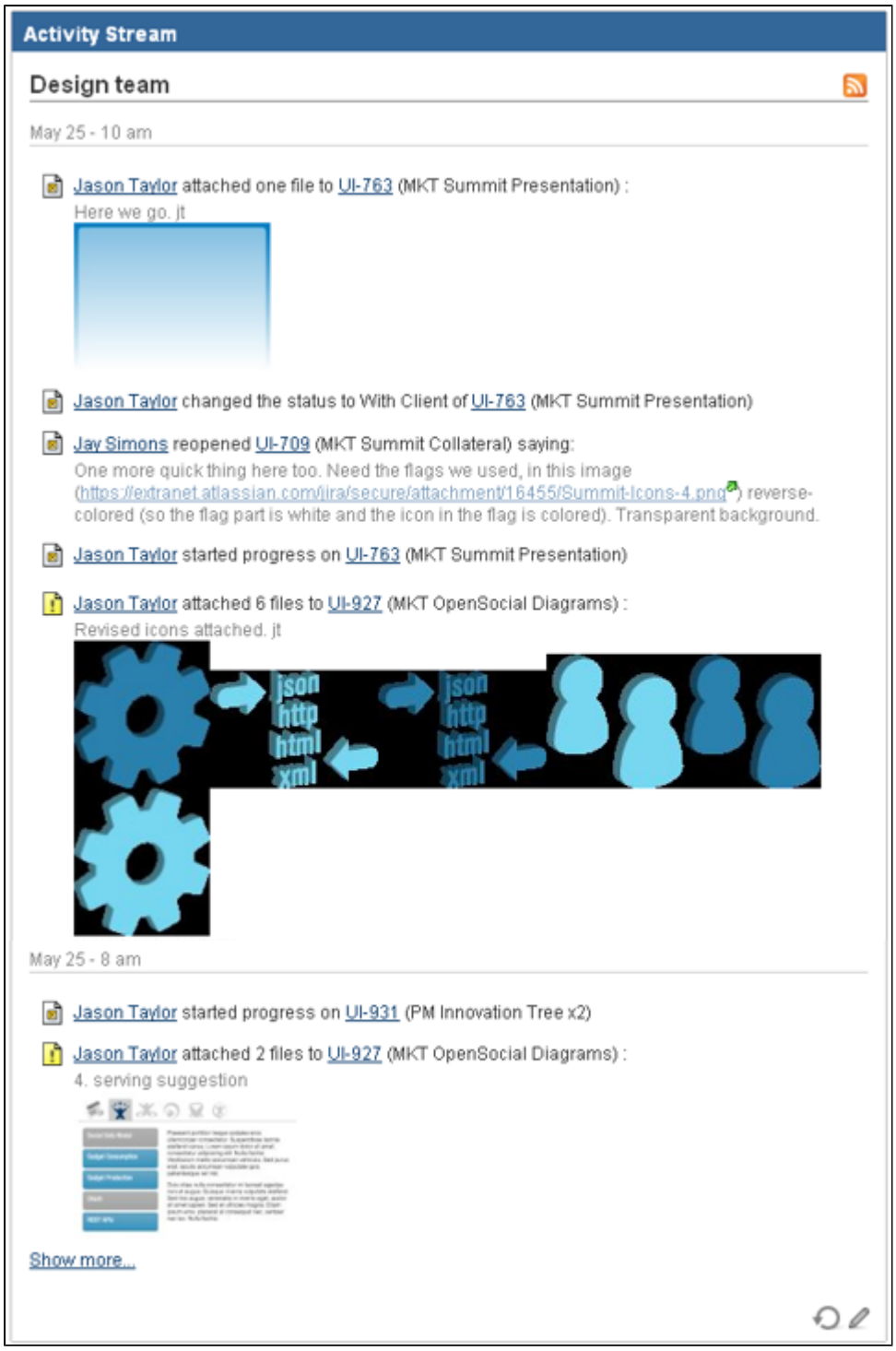
^Top

Activity Stream

The new '**Activity Stream**' gadget displays a summary of the latest activity in JIRA projects (and/or by particular people) in which you are interested.

The '**Activity Stream**' gadget also provides an RSS feed, allowing you to create very specific RSS feeds of only the information that is most relevant to you.

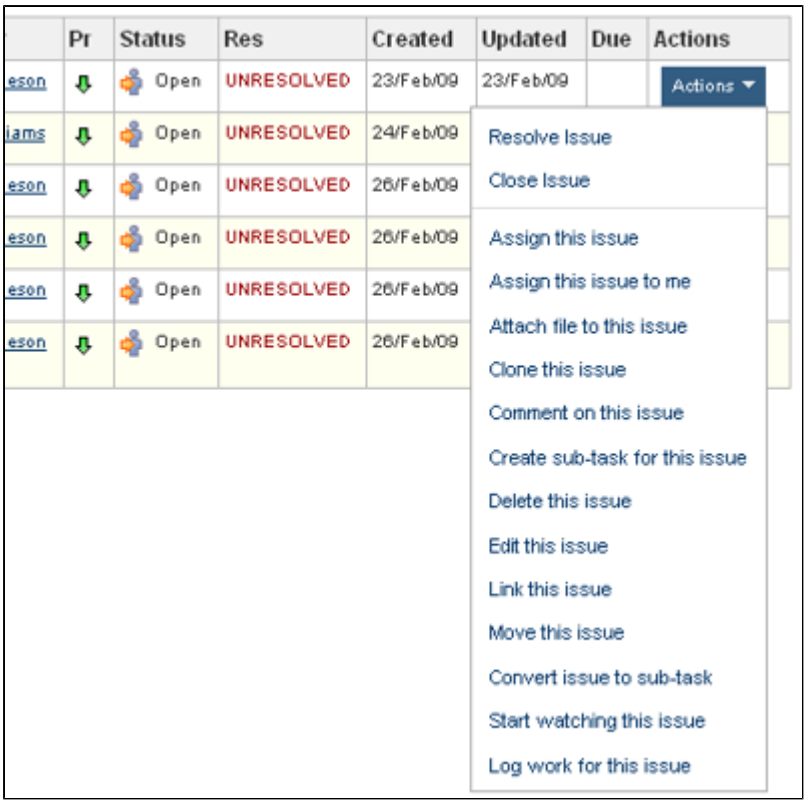
See the documentation for more details.



^Top

Issue Actions in the Issue Navigator

By popular request, issues can now be actioned directly from the Issue Navigator:

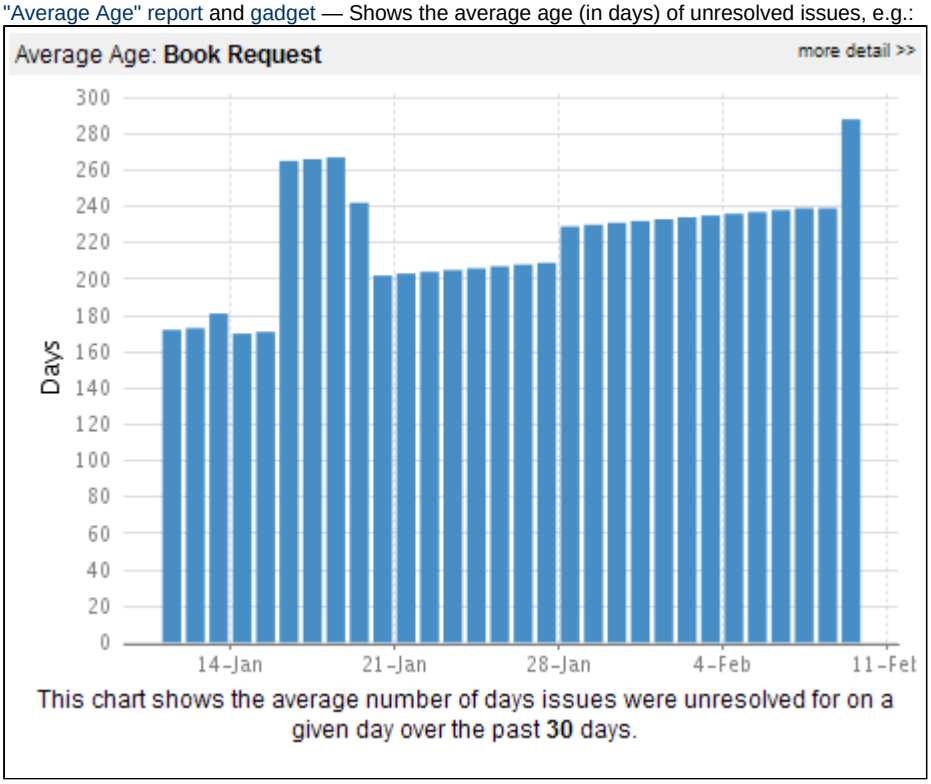


The "Actions" menu is also available for the list of sub-tasks within an issue.
^Top

Charting Now Comes Standard

The following reports and gadgets from the Charting plugin have now been integrated into JIRA:

- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over

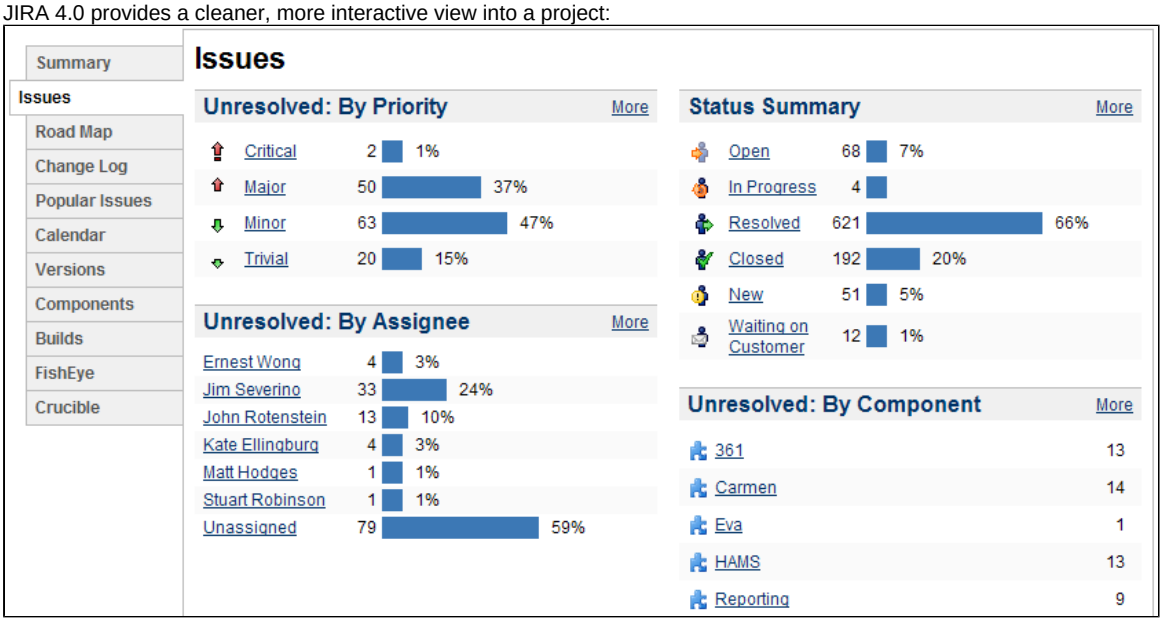
a given period of time.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.

Also, the "Resolved" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution date recorded.

^Top

New-look "Browse Project"

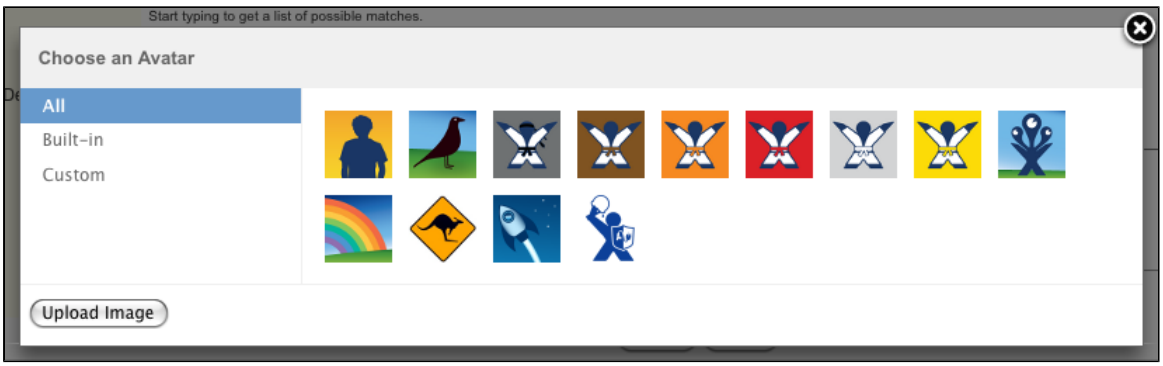
JIRA 4.0 provides a cleaner, more interactive view into a project:



See the documentation for more about browsing projects, versions and components.
^Top

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):

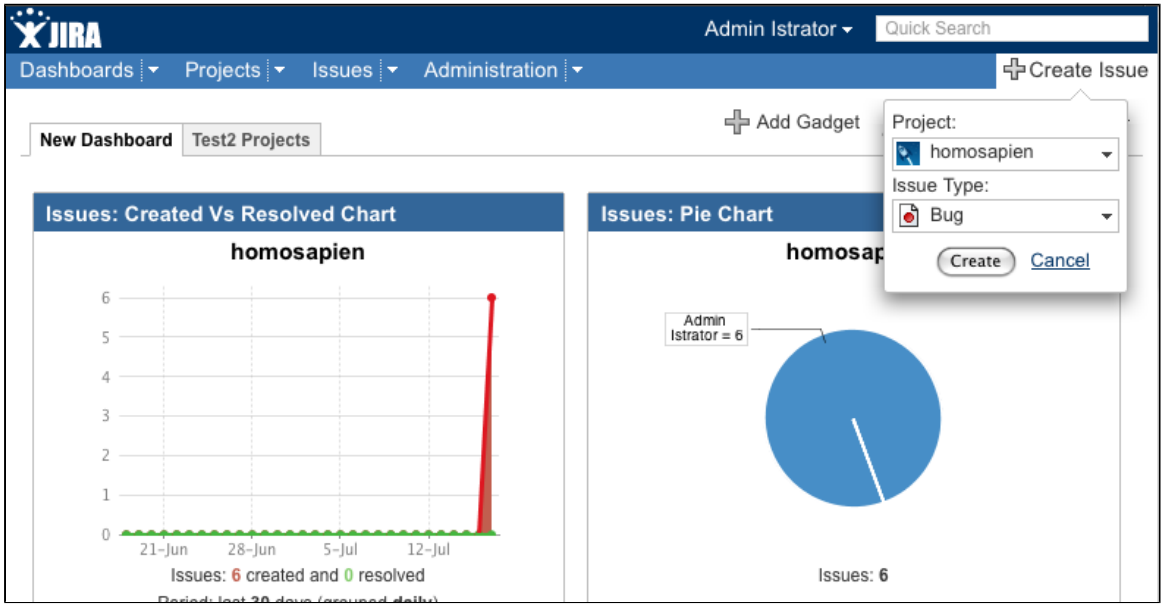


^Top

8

New-look Header

The new-look JIRA header gives you quick access to all the most commonly-used functions. Creating an issue just got super-fast!

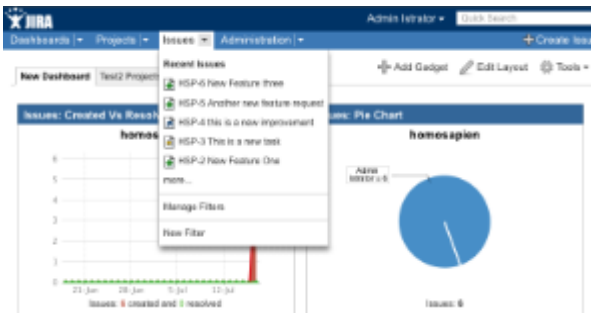


Click to zoom in:

The new 'Create Issue'

The new 'User' menu

The



The new 'Issues' menu

The new 'Projects' menu

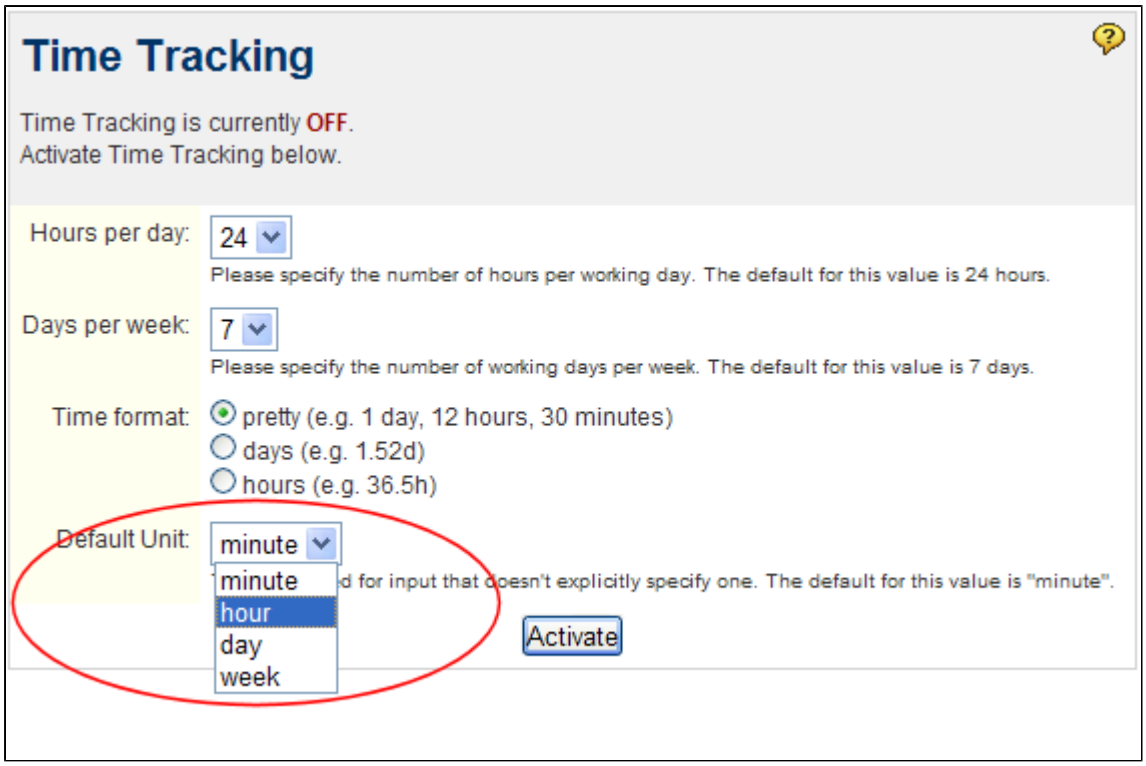
The

^Top

9

Default Unit for Time Tracking

You can now specify your preferred **Default Unit** (minutes/hours/days/weeks) for your JIRA system. This will be applied whenever users log work on an issue without specifying a unit.



^Top

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.

^Top

Engine Room

Beyond the 'Back' Button

When navigating away from a page where you have modified data, you will be prompted to see if you would like to save the data or discard your changes (see JRA-14911).

Index Queue

Index updates are now put in a queue. So even if the update takes longer than 30 seconds, the operation remains on the queue and is not lost. (See JRA-14220.)

^Top

Plus over 250 other fixes and improvements

Click here for full list.

JIRA Issues (200 issues)				
Key	**Type**	**Summary**	**Priority**	**Status**
JRA-923		Allow filter by "No Fix For" across projects		Resolved

JRA-1538	Filter on Versions and Components across Projects	Resolved
JRA-1560	Better support for logical operation (and/or/not) type of filters.	Resolved
JRA-1579	Create a portlet for the recent history	Resolved
JRA-1635	"not" qualifier on fields for searching	Resolved
JRA-1642	Create home directory instead of index & attachment directory	Resolved
JRA-1800	Improve the UI for browse project	Resolved
JRA-1844	Display attachment comments associated with their attachments	Resolved
JRA-1983	Enable filtering on "older than 1 month"	Resolved
JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-2033	Add an RSS feed query for comments to individual issues	Resolved
JRA-2469	It would be really nice to specficy several Asignee options in filters	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-2810	Recently viewed issues	Resolved
JRA-2852	search for issues on version lower or equal to a given version	Resolved
JRA-2916	Allow Previous version searching	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-3000	Add key NUMBER (only number) searching to default search filter.	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-3114	Request: add optional icon for each project	Resolved
JRA-3206	View issues without an estimate	Resolved
JRA-3451	Enable filtering by Date Resolved	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-3624	released/unreleased version filter	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-4227	Recent History Popup - persistance across sessions & more data	Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria	Resolved

JRA-4688		Browse Project: Within the tab panel, if components are hidden - the version info appears to be right aligned.	Resolved
JRA-5121		Filter Portlet with configurable columns	Resolved
JRA-5152		Show issue linked to another issue.	Resolved
JRA-5201		Enable filter to specify more than 1 user	Resolved
JRA-5310		Watchlist should be exportable	Resolved
JRA-5383		My Votes and My Watches as filters	Resolved
JRA-5435		Issue actions and operations on Issue Navigator	Resolved
JRA-5560		Improved query functionality	Resolved
JRA-5798		Project Portlet: needs multi project selection	Resolved
JRA-5844		allow multiple users when creating filters	Resolved
JRA-5965		Allow configure units of time tracking	Resolved
JRA-6010		Thought processing	Resolved
JRA-6164		'No Priority' count is not displayed in filter statistics portlet	Resolved
JRA-6170		Radio Buttons should support Select List Searcher template	Resolved
JRA-6180		Search for a custom field that is empty	Resolved
JRA-6344		Send to both previous and current assignees for all notifications	Resolved
JRA-6527		Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-6550		if text contains certain characters, cdata in xml based on this will be badly formed	Resolved
JRA-7068		Allow for list of issues to be saved as a filter	Resolved
JRA-7551		Provide capability to find issues by resolution date	Resolved
JRA-7626		Build search queries remotely	Resolved
JRA-7772		Ability to create advanced queries to search across all data	Resolved
JRA-7909		Search/ filter for "empty" fields	Resolved
JRA-8159		Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type	Resolved
JRA-8293		Import fails if searchrequest:request data too large	Resolved
JRA-8487		Bad logging from uk.ltd.getahead.dwr.util.CommonsLoggingOutput on startup	Resolved
JRA-8527		Put task actions directly in filter output	Resolved

JRA-8606	Need a way to find watched issues		Resolved
JRA-8686	Allow searching of issues by Full Name for all user fields		Resolved
JRA-8758	Cannot create filter for multiple projects all issues in version "Released Versions"		Resolved
JRA-8806	Allow "Released" & "Unreleased" Version search accross multiple projects		Resolved
JRA-8852	Sort filter results by non-visible field		Resolved
JRA-8973	RSS of Project Changes		Resolved
JRA-9048	Calendar week begins with sunday independently from locale		Resolved
JRA-9115	Ability to search for issues with no due date associated		Resolved
JRA-9278	New Field "Resolution Date" automatically filled with date of setting resolution		Resolved
JRA-9551	Search for all Sub-Tasks of one given issue		Resolved
JRA-9651	User Activity Log		Resolved
JRA-9658	Minor css bug (cursor)		Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.		Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"		Resolved
JRA-10405	Attachment ordering		Resolved
JRA-10422	Error in logs when nonexistent key used in wiki-rendered text		Resolved
JRA-10427	Changing field descriptions in "Field Configurations" for custom fields does not work		Resolved
JRA-10443	"Not Assigned to User" criteria in filters		Resolved
JRA-10492	Search for several users as Assignee or Reporter		Resolved
JRA-10554	Changing locale causes no translation change for 'Browse Projects' menu tab unless a project has been or is already selected		Resolved
JRA-10603	MultipleSelect searcher for cascading selection field		Resolved
JRA-10644	Make filters more accessible		Resolved
JRA-10658	More columns on Dashboards		Resolved
JRA-10854	'Restoring Data' Documentation incorrect or unclear		Closed
JRA-11134	Allow setting of column order/sort with no issues in result set		Resolved
JRA-11933	AutoTransitionListener - Reopen transition deletes issue summary		Resolved
JRA-12165	Unclear error message when bulk moving issues whose reporter cannot create issues		Resolved

JRA-12177	Time tracking by using setting "hours" - edit issue shows "Original/Remaining Estimate" -field value in "pretty" mode		Resolved
JRA-12200	Reporter system field throws ClassCastException after populateFromIssue() and validateFromParams()		Resolved
JRA-12525	Emails containing attachments with non-ASCII names lost		Resolved
JRA-12596	Enable cross-project filtering on special versions		Resolved
JRA-12656	Add paging/optimization for Change Log scope		Resolved
JRA-12816	OutOfMemoryError PermGen Space on Windows Func Test (under VMWare)		Resolved
JRA-12921	Ability to export Watched Issues to excel		Resolved
JRA-12976	AbstractMessageHandler might not be removing spaces from email addresses before using them to determine if a user exists when creating an issue from an email		Resolved
JRA-13003	Moving portlet up reults in IndexOutOfBoundsException		Resolved
JRA-13011	Component of a subtask is still component of original project after moving an issue		Resolved
JRA-13035	CSV import can not import resolution date.		Resolved
JRA-13426	Next/previous version links for 'Browse Version' screen		Resolved
JRA-13625	Implicit profiling functionality broken		Resolved
JRA-13689	Saved filters reverted to "All projects" when we deleted a project contained		Resolved
JRA-13711	Printer icon on 'Issue Navigator' does not show the Printable View of the issue navigator as it did in JIRA 3.6		Resolved
JRA-13745	Clean up top toolbar by moving Profile link to username and removing Filters link		Resolved
JRA-13793	Confusing "The 'Project Information' panel is not available" message when fields are disabled		Resolved
JRA-13801	Call method addWorklogAndAutoAdjustRemainingEstimate, the soap server reponse with this type IssueServiceImpl$RemoteWorklogImpl		Resolved
JRA-13850	Servlet Content Listeners should implement the catch / log / rethrow pattern		Resolved
JRA-14031	Form data lost when using back and forward web browser buttons		Resolved
JRA-14220	Ensure the index optimize operation does not cause index lock timeouts		Resolved
JRA-14416	Move Issue with SubTask between differents project, IssueType and SubIssueType		Resolved
JRA-14419	Warning for Websphere installation on validating entity-engine.xml		Closed
JRA-14490	Deleting project can cause filter to select all projects		Resolved
JRA-14513	JIRA Soap Service log and Access filter log footprint needs improving - Invoked Method would be handy		Resolved
JRA-14516	JIRA upgrade page should warn about possible character encoding issue if JIRA is moved between two servers.		Closed
JRA-14598	Add access key for administer project from browse project		Resolved

JRA-14613	Each project can have its own logo		Resolved
JRA-14616	Ability to query for issues that you are not watching		Resolved
JRA-14654	ColorPicker for LookAndFeel page does not work on all browsers		Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype		Resolved
JRA-14727	Cannot create a literal "backslash underscore" sequence		Resolved
JRA-14811	Deleting Group Does Not Remove Group From a Subscription		Resolved
JRA-14826	Dashboards with a large number of portal pages cause the page to become too wide		Resolved
JRA-14983	Fetch only updated or changed issues		Resolved
JRA-15018	Improved SOAP and HTTP access logging		Resolved
JRA-15112	Adding Update Issue Field workflow postfunction causes OutOfMemoryError		Resolved
JRA-15241	Single user picker field renrered incorrectly		Resolved
JRA-15247	Duplicate explanation of entityengine.xml in the upgrade guide.		Closed
JRA-15254	Browse Project Panel: Do not show Closed UNRESOLVED issues as Open in the version list		Resolved
JRA-15266	Would prefer if the "Worklog" heading under "Operations" when viewing an issue were a link, like the rest of the operations.		Closed
JRA-15309	Wrong assignee drop-down list sorting with non-ascii characters		Resolved
JRA-15327	On the login page the message at the bottom is off center if an error is displayed		Resolved
JRA-15383	Documentation on Two Dimentional Statistics Filter Portlet needs to specify suported custom fields		Closed
JRA-15445	RPC plugin needs to be cleaned up		Resolved
JRA-15484	Tokenizing java exceptions fails if the exception is terminated with a full-stop.		Resolved
JRA-15517	Upgrade JIRA to use the latest version of Lucene indexing framework - v2.3.2		Resolved
JRA-15543	Show release date next to version name in the list of versions on Browse Project screen		Resolved
JRA-15545	Export issues to Excel/Word format with non-ASCII filter name does not handle the file name properly		Resolved
JRA-15546	Versions no longer display descriptions when browsing project		Resolved
JRA-15548	If an attempt to get the Index lock times out, the indexing operation is discarded		Resolved
JRA-15564	JIRA displays error after a reindex.		Resolved
JRA-15575	Test and confim JIRA is compatible with Microsoft SQL Server 2008		Closed
JRA-15625	It is possible to disable plugins that then render jira incapable of restarting		Resolved

JRA-15631	jelly with invalid variables script returns blank page to user	↓	Resolved
JRA-15638	The new dropdown does not appear to render correctly under IE6/7	↓	Resolved
JRA-15646	Convert JIRA to jQuery		Resolved
JRA-15649	Sort list of plugins in Admin section alphabetically		Resolved
JRA-15665	Address issue of plugins pushing filter/report panel off screen		Resolved
JRA-15666	Add project information to the issue XML view		Resolved
JRA-15669	Drag and drop behaviour doesn't work on Modify Issues Type Scheme page	↑	Resolved
JRA-15700	Created VS Resolved cumulative + individual graphs' Y axis should should be independent		Resolved
JRA-15702	Migrate to licensing 2.0		Resolved
JRA-15723	Jelly AddComment tag changes the "updated" issue timestamp to execution script timestamp	↓	Resolved
JRA-15732	Update email documentation to highlight that OutOfMemoryError can stop email processing		Closed
JRA-15761	If issue key contains unicode characters, redirect on create issue doesn't work	↓	Resolved
JRA-15846	Allow changing license on Data Import in new JIRA instance.		Resolved
JRA-15872	"Browse Project" URL doesn't include current project		Resolved
JRA-15882	Notification Schemes image highlights the wrong function	↓	Closed
JRA-15886	Add logging notification for index optimization events		Resolved
JRA-15920	Include warning in EAR/WAR documentation not to edit anything directly on the application server		Closed
JRA-15927	Code samples on Web UI Plugin Module document are incorrect	↓	Closed
JRA-15962	Upgrade JIRA to Plugins 2.x		Resolved
JRA-15975	Couldn't log in separate log files if more then one instance of JIRA started on Tomcat	↓	Resolved
JRA-15991	Merge translations files into one to make translating JIRA easier!		Resolved
JRA-16058	Aggressive locking in JiraCachingPropertySet causes high contention		Resolved
JRA-16067	Provide field definition in XML issue view URL to customize XML view		Resolved
JRA-16074	Incorrect error warning message on navigator summary	↓	Resolved
JRA-16080	1px offset in Firefox	↓	Resolved
JRA-16088	Created VS Resolved Issues Report contains Old Filtter / Project Picker	↓	Resolved
JRA-16112	Bug in progressWorkflowAction method in SOAP	↓	Resolved

JRA-16113	Do not show negative values on Y axis in Created vs. Resolved chart		Resolved
JRA-16120	Dashboard rewrite		Resolved
JRA-16122	HTTP Basic auth should be enabled by default		Resolved
JRA-16138	Anonymous users should not be considered to "own" all anonymous comments.		Resolved
JRA-16151	Colon : in custom field search causes searching of wrong field		Resolved
JRA-16175	JIRA issues macro does not work with 4.0-m1 on EACJ		Resolved
JRA-16210	Display issue count on JQL execution		Resolved
JRA-16211	Enter / return should execute JQL		Resolved
JRA-16253	Source Build documentation is out of date and incomplete		Closed
JRA-16276	Adjust colours of Resoltion date chart to be more distinshable for the colour blind		Resolved
JRA-16278	Add ability to search for versions using regex or similar		Resolved
JRA-16316	Assigned To Me Portlet, selecting all columns to display causes error		Resolved
JRA-16339	The "Perm Gen" memory usage shown on the System Info page is incorrect.		Resolved
JRA-16351	Component plugin modules don't show up in plugins admin section in JIRA 4.0		Resolved
JRA-16363	ServiceProxyDestroyedException when you reactivate an OSGi plugin		Resolved
JRA-16379	Weblogic Deployment descriptor (weblogic.xml) has changed for Weblogic 9.x		Closed
JRA-16407	JiraModuleDescriptorFactory doesn't define some plugins2 descriptors		Resolved
JRA-16424	log4j output should contain more information		Resolved
JRA-16443	Create a jira.field.resolution.include transition attribute		Resolved
JRA-16451	JIRA home directory created in working directory in JIRA standalone		Resolved
JRA-16485	The long component name, on clicking overlaps the UI element		Resolved
JRA-16498	Version/s and Component/s not validated when updating an issue		Resolved
JRA-16502	Local helper has invalid HTML, causing styling issues		Resolved
JRA-16508	no attachments are returned when 'field=attachment' is specified in XML view		Resolved
JRA-16509	Check for javascript enabled in browser		Resolved
JRA-16510	Update to jQuery 1.3.2		Resolved
JRA-16518	Update readme.txt in war		Closed

JRA-16522	Searching according to multiple assignees should be provided.		Resolved
JRA-16526	Catalan translations needs to be changed		Resolved
JRA-16527	Indexing fails (or atleast error thrown) if index location is pointing to an invalid reference		Resolved
JRA-16529	Project Avatars: User is unable to upload PNG and JPEG file for project avatars on IE 7.0		Resolved
JRA-16530	Project Avatars: User is unable to crop the uploaded image in the project avatars dialog		Resolved
JRA-16531	Project avatar: One image is missing from "Choose an avatar" dialog		Resolved
JRA-16532	User gets a system error if after timeout tries to login from project avatar dialog		Resolved
JRA-16538	User Profile:User Summary area's formatting is not correct when logged in user view profile of other user.		Resolved
JRA-16560	Upgrading bunded plugins fails		Resolved
JRA-16568	Example for xmlrpc client uses a method that has been removed. Need to update		Closed

^Top

JIRA 4.0 Beta 1 Release Notes

JIRA 4.0 Beta 1 is a public development release leading up to **JIRA 4.0**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

Beta releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **Beta releases are not safe**— Beta releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because Beta releases represent work in progress, we **cannot** provide a supported upgrade path between Beta releases, or from any Beta to the eventual final release. Thus, any data you store in a JIRA Beta release may not be able to be migrated to a future JIRA release.

The **only** plugin that is compatible with JIRA 4.0 Beta is the latest JIRA Toolkit. Do not install any other plugins.

The Atlassian JIRA team is delighted to present a brand new version of one of the world's favourite issue-trackers.

Highlights of JIRA 4.0 Beta 1:

- Advanced Searching
- Issue Actions in the Issue Navigator
- Charting Now Comes Standard
- New-look "Browse Project"
- Project Icons
- Dashboard Gadgets
- Activity Stream
- "History" is now permanent
- Plus over 150 other fixes and improvements

Thank you for your interest in JIRA 4.0 Beta 1

Download Beta

Upgrading to JIRA 4.0 Beta
JIRA 4.0 Beta can be downloaded here. Before upgrading, please refer to the JIRA 4.0 Upgrade Guide.

Highlights of JIRA 4.0 Beta 1

Advanced Searching

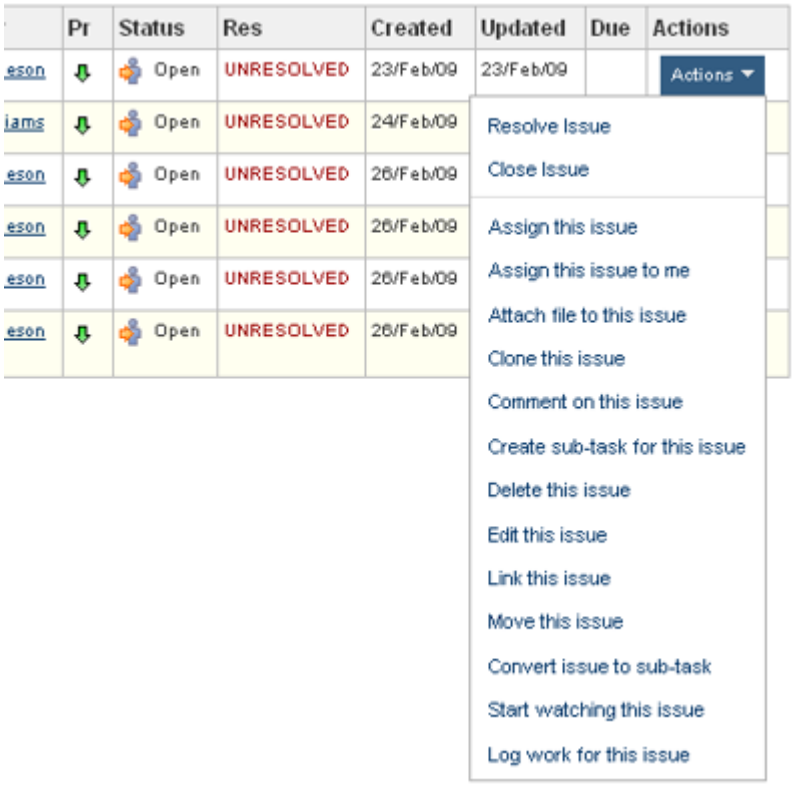
The new advanced search (JQL) provides support for logical operations (including AND/OR/NOT/NULL, even on custom fields) when filtering issues:

For more on the new JQL search syntax, please see the documentation.

^Top

Issue Actions in the Issue Navigator

By popular request, issues can now be actioned directly from the Issue Navigator:



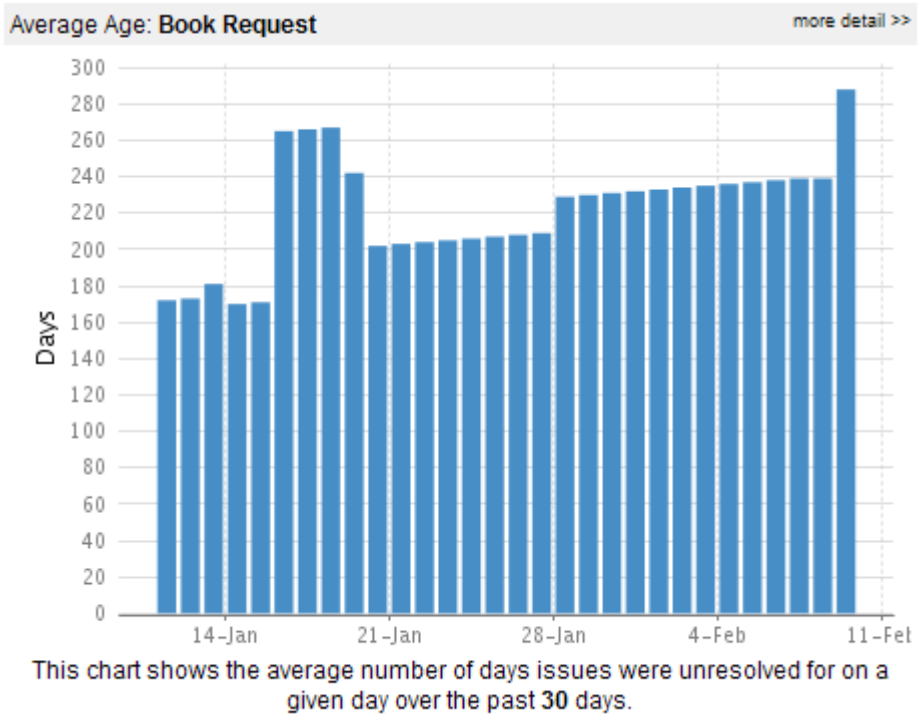
The "Actions" menu is also available for the list of sub-tasks within an issue.

^Top

Charting Now Comes Standard

The following reports and gadgets from the Charting plugin have now been integrated into JIRA:

- "Average Age" report and gadget — Shows the average age (in days) of unresolved issues, e.g.:



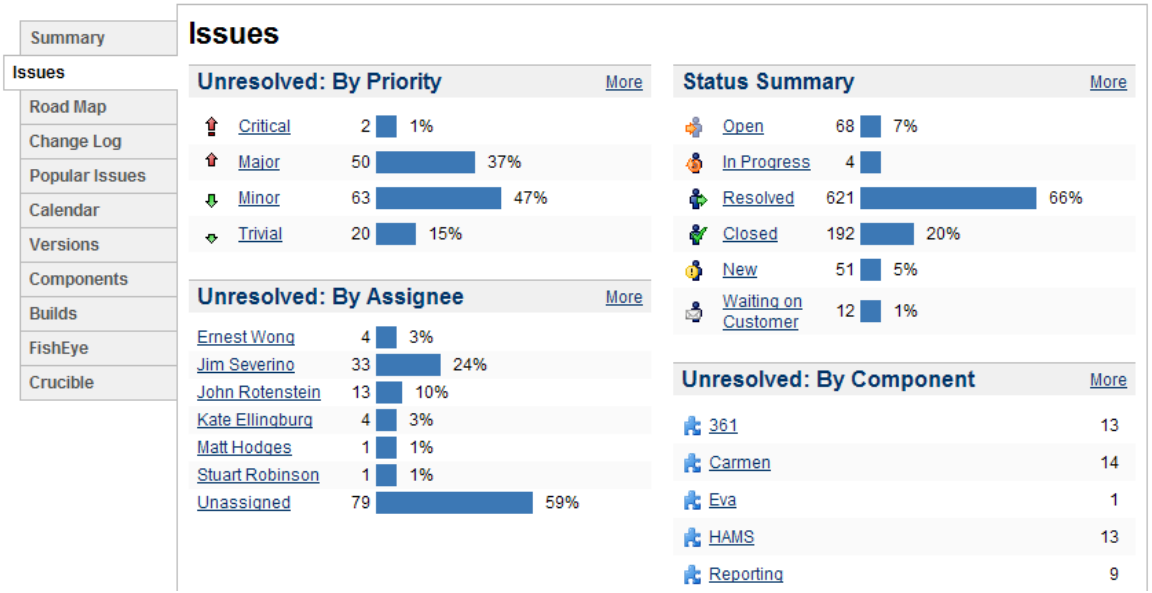
- "Created vs Resolved Issues" report and gadget — Shows the number of issues created vs number of issues resolved over a given period of time.
- "Pie Chart" report and gadget — Shows the search results from a specified issue filter (or project) in a pie-chart, based on a statistic of your choice.
- "Recently Created Issues" report and gadget — Shows the rate at which issues are being created.
- "Resolution Time" report and gadget — Shows the average time taken to resolve issues.
- "Time Since Issues" report and gadget — Shows the number of issues for which your chosen date field (e.g. 'Created') was set on a given date.

Also, the "Resolved" field from the Charting plugin is now part of JIRA, so every issue now automatically has its resolution date recorded.

^Top

New-look "Browse Project"

JIRA 4.0 provides a cleaner, more interactive view in to a project:



See the documentation for more about browsing projects, versions and components.

^Top

Project Icons

You can now give your project a visual identity, thanks to the introduction of project icons ('avatars'):

^Top

Dashboard Gadgets

The new-look JIRA dashboard now uses industry-standard 'gadgets'. So you can add external gadgets to your JIRA dashboard, as well as displaying JIRA gadgets in other places (such as iGoogle).

What's happened to your favourite JIRA portlets? Don't worry: every portlet that previously shipped with JIRA has been converted to a 'legacy gadget'. And if you are a plugin developer and have created your own portlets, see the instructions for converting your portlets to gadgets.

The following new gadgets are available in Beta 1:

- 'Activity Stream' gadget (see below)
- 'Create Issue' gadget
- 'Filter Results' gadget
- 'Issue Completed This Iteration' gadget

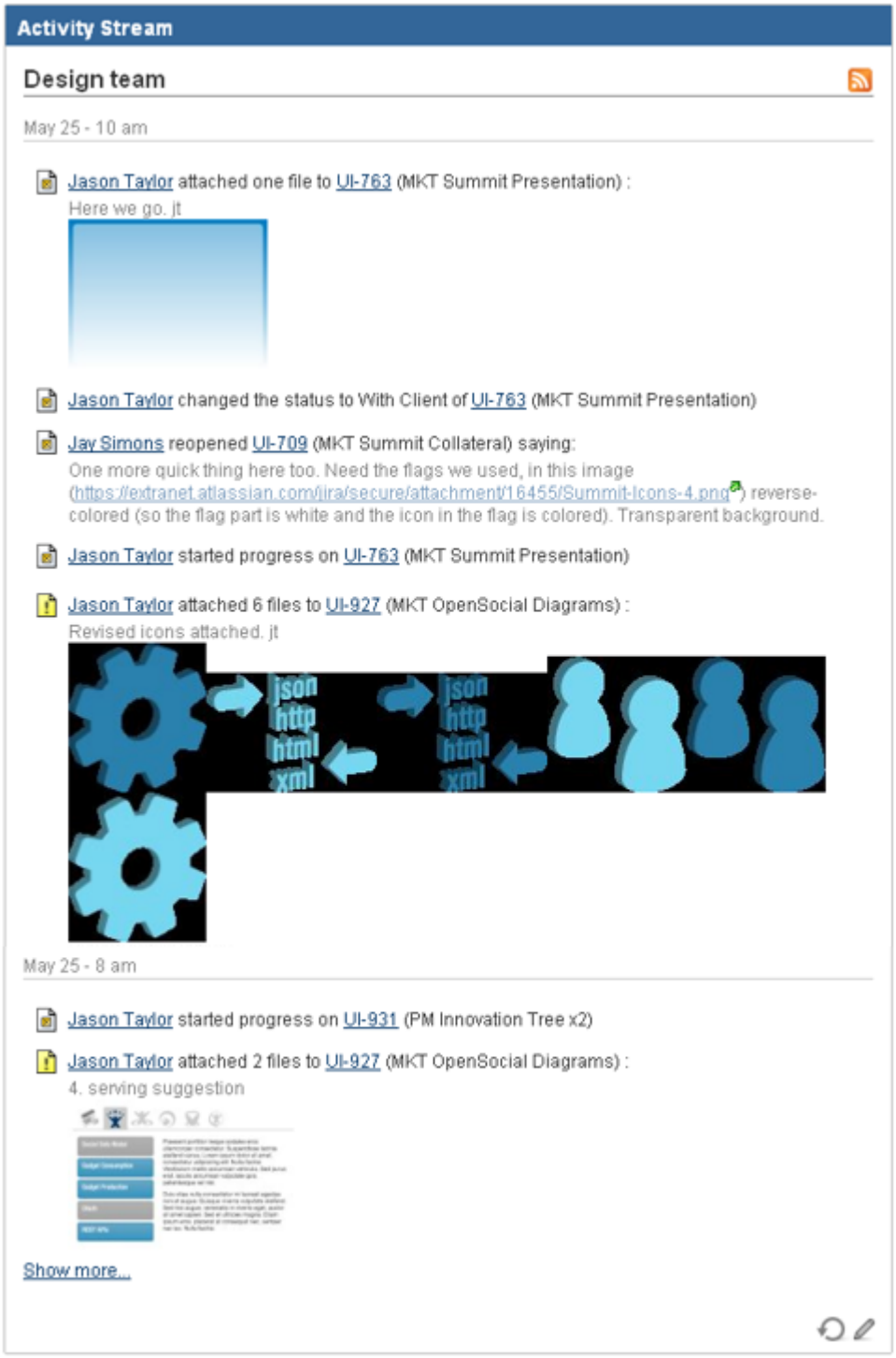
For optimal experience of the new dashboard, please use Firefox 3.x. Support for other browsers will be added prior to the launch of JIRA 4.0.

^Top

Activity Stream

The new 'Activity Stream' gadget displays a summary of the latest activity in JIRA projects (and/or by particular people) in which you are interested.

See the documentation for more details.



^Top

8

"History" is now permanent

Your list of recently-viewed issues is now stored in JIRA's database — so it's available after you log out and back in, even if you use a different machine.

^Top

Plus over 150 other fixes and improvements

JIRA Issues (200 issues)

Key	Type	Summary	Priority	Status
JRA-923		Allow filter by "No Fix For" across projects		Resolved
JRA-1538		Filter on Versions and Components across Projects		Resolved
JRA-1560		Better support for logical operation (and/or/not) type of filters.		Resolved
JRA-1579		Create a portlet for the recent history		Resolved
JRA-1635		"not" qualifier on fields for searching		Resolved
JRA-1642		Create home directory instead of index & attachment directory		Resolved
JRA-1800		Improve the UI for browse project		Resolved
JRA-1844		Display attachment comments associated with their attachments		Resolved
JRA-1983		Enable filtering on "older than 1 month"		Resolved

JRA-1994	Ability to filter on time tracking related fields	Resolved
JRA-2033	Add an RSS feed query for comments to individual issues	Resolved
JRA-2469	It would be really nice to specficy several Asignee options in filters	Resolved
JRA-2607	Would like to create a filter also with OR conditions	Resolved
JRA-2681	Extend filter capablities by adding negative clauses	Resolved
JRA-2810	Recently viewed issues	Resolved
JRA-2852	search for issues on version lower or equal to a given version	Resolved
JRA-2916	Allow Previous version searching	Resolved
JRA-2925	Can't filter by Security Level	Resolved
JRA-3000	Add key NUMBER (only number) searching to default search filter.	Resolved
JRA-3101	Jira - query / search / filter by issue links	Resolved
JRA-3114	Request: add optional icon for each project	Resolved
JRA-3206	View issues without an estimate	Resolved
JRA-3451	Enable filtering by Date Resolved	Resolved
JRA-3464	allow filtering by project category	Resolved
JRA-3624	released/unreleased version filter	Resolved
JRA-4059	Record last login time for a user	Resolved
JRA-4227	Recent History Popup - persistance across sessions & more data	Resolved
JRA-4605	new filter criteria: add NOT to all existing criteria	Resolved
JRA-4688	Browse Project: Within the tab panel, if components are hidden - the version info appears to be right aligned.	Resolved
JRA-5121	Filter Portlet with configurable columns	Resolved
JRA-5152	Show issue linked to another issue.	Resolved
JRA-5201	Enable filter to specify more than 1 user	Resolved
JRA-5310	Watchlist should be exportable	Resolved
JRA-5383	My Votes and My Watches as filters	Resolved
JRA-5435	Issue actions and operations on Issue Navigator	Resolved
JRA-5560	Improved query functionality	Resolved

JRA-5798		Project Portlet: needs multi project selection	Resolved
JRA-5844		allow multiple users when creating filters	Resolved
JRA-5965		Allow configure units of time tracking	Resolved
JRA-6010		Thought processing	Resolved
JRA-6164		'No Priority' count is not displayed in filter statistics portlet	Resolved
JRA-6170		Radio Buttons should support Select List Searcher template	Resolved
JRA-6180		Search for a custom field that is empty	Resolved
JRA-6344		Send to both previous and current assignees for all notifications	Resolved
JRA-6527		Allow filters to be built upon other shared filters (combined filters)	Resolved
JRA-6550		if text contains certain characters, cdata in xml based on this will be badly formed	Resolved
JRA-7068		Allow for list of issues to be saved as a filter	Resolved
JRA-7551		Provide capability to find issues by resolution date	Resolved
JRA-7626		Build search queries remotely	Resolved
JRA-7772		Ability to create advanced queries to search across all data	Resolved
JRA-7909		Search/ filter for "empty" fields	Resolved
JRA-8159		Add ability to issue navigator to find all issues linked to x issue - with option to constrain by link type	Resolved
JRA-8293		Import fails if searchrequest:request data too large	Resolved
JRA-8487		Bad logging from uk.ltd.getahead.dwr.util.CommonsLoggingOutput on startup	Resolved
JRA-8527		Put task actions directly in filter output	Resolved
JRA-8606		Need a way to find watched issues	Resolved
JRA-8686		Allow searching of issues by Full Name for all user fields	Resolved
JRA-8758		Cannot create filter for multiple projects all issues in version "Released Versions"	Resolved
JRA-8806		Allow "Released" & "Unreleased" Version search accross multiple projects	Resolved
JRA-8852		Sort filter results by non-visible field	Resolved
JRA-8973		RSS of Project Changes	Resolved
JRA-9048		Calendar week begins with sunday independently from locale	Resolved
JRA-9115		Ability to search for issues with no due date associated	Resolved

JRA-9278	New Field "Resolution Date" automatically filled with date of setting resolution		Resolved
JRA-9551	Search for all Sub-Tasks of one given issue		Resolved
JRA-9651	User Activity Log		Resolved
JRA-9658	Minor css bug (cursor)		Resolved
JRA-9823	Allow to optionally clone an issue's attachments when cloning an issue.		Resolved
JRA-10245	Ability to filter/view Issues upon "Versions" across multiple "Projects"		Resolved
JRA-10405	Attachment ordering		Resolved
JRA-10422	Error in logs when nonexistent key used in wiki-rendered text		Resolved
JRA-10427	Changing field descriptions in "Field Configurations" for custom fields does not work		Resolved
JRA-10443	"Not Assigned to User" criteria in filters		Resolved
JRA-10492	Search for several users as Assignee or Reporter		Resolved
JRA-10554	Changing locale causes no translation change for 'Browse Projects' menu tab unless a project has been or is already selected		Resolved
JRA-10603	MultipleSelect searcher for cascading selection field		Resolved
JRA-10644	Make filters more accessible		Resolved
JRA-10658	More columns on Dashboards		Resolved
JRA-10854	'Restoring Data' Documentation incorrect or unclear		Closed
JRA-11134	Allow setting of column order/sort with no issues in result set		Resolved
JRA-11933	AutoTransitionListener - Reopen transition deletes issue summary		Resolved
JRA-12165	Unclear error message when bulk moving issues whose reporter cannot create issues		Resolved
JRA-12177	Time tracking by using setting "hours" - edit issue shows "Original/Remaining Estimate" -field value in "pretty" mode		Resolved
JRA-12200	Reporter system field throws ClassCastException after populateFromIssue() and validateFromParams()		Resolved
JRA-12525	Emails containing attachments with non-ASCII names lost		Resolved
JRA-12596	Enable cross-project filtering on special versions		Resolved
JRA-12656	Add paging/optimization for Change Log scope		Resolved
JRA-12816	OutOfMemoryError PermGen Space on Windows Func Test (under VMWare)		Resolved
JRA-12921	Ability to export Watched Issues to excel		Resolved
JRA-12976	AbstractMessageHandler might not be removing spaces from email addresses before using them to determine if a user exists when creating an issue from an email		Resolved

JRA-13003	Moving portlet up reults in IndexOutOfBoundsException		Resolved
JRA-13011	Component of a subtask is still component of original project after moving an issue		Resolved
JRA-13035	CSV import can not import resolution date.		Resolved
JRA-13426	Next/previous version links for 'Browse Version' screen		Resolved
JRA-13625	Implicit profiling functionality broken		Resolved
JRA-13689	Saved filters reverted to "All projects" when we deleted a project contained		Resolved
JRA-13711	Printer icon on 'Issue Navigator' does not show the Printable View of the issue navigator as it did in JIRA 3.6		Resolved
JRA-13745	Clean up top toolbar by moving Profile link to username and removing Filters link		Resolved
JRA-13793	Confusing "The 'Project Information' panel is not available" message when fields are disabled		Resolved
JRA-13801	Call method addWorklogAndAutoAdjustRemainingEstimate, the soap server reponse with this type IssueServiceImpl$RemoteWorklogImpl		Resolved
JRA-13850	Servlet Content Listeners should implement the catch / log / rethrow pattern		Resolved
JRA-14031	Form data lost when using back and forward web browser buttons		Resolved
JRA-14220	Ensure the index optimize operation does not cause index lock timeouts		Resolved
JRA-14416	Move Issue with SubTask between differents project, IssueType and SubIssueType		Resolved
JRA-14419	Warning for Websphere installation on validating entity-engine.xml		Closed
JRA-14490	Deleting project can cause filter to select all projects		Resolved
JRA-14513	JIRA Soap Service log and Access filter log footprint needs improving - Invoked Method would be handy		Resolved
JRA-14516	JIRA upgrade page should warn about possible character encoding issue if JIRA is moved between two servers.		Closed
JRA-14598	Add access key for administer project from browse project		Resolved
JRA-14613	Each project can have its own logo		Resolved
JRA-14616	Ability to query for issues that you are not watching		Resolved
JRA-14654	ColorPicker for LookAndFeel page does not work on all browsers		Resolved
JRA-14701	OSPropertyText table should have the value column set to extremely-long datatype		Resolved
JRA-14727	Cannot create a literal "backslash underscore" sequence		Resolved
JRA-14811	Deleting Group Does Not Remove Group From a Subscription		Resolved
JRA-14826	Dashboards with a large number of portal pages cause the page to become too wide		Resolved
JRA-14983	Fetch only updated or changed issues		Resolved

JRA-15018	Improved SOAP and HTTP access logging		Resolved
JRA-15112	Adding Update Issue Field workflow postfunction causes OutOfMemoryError	↑	Resolved
JRA-15241	Single user picker field renrered incorrectly	↓	Resolved
JRA-15247	Duplicate explanation of entityengine.xml in the upgrade guide.	↑	Closed
JRA-15254	Browse Project Panel: Do not show Closed UNRESOLVED issues as Open in the version list	↓	Resolved
JRA-15266	Would prefer if the "Worklog" heading under "Operations" when viewing an issue were a link, like the rest of the operations.		Closed
JRA-15309	Wrong assignee drop-down list sorting with non-ascii characters	↓	Resolved
JRA-15327	On the login page the message at the bottom is off center if an error is displayed	↓	Resolved
JRA-15383	Documentation on Two Dimentional Statistics Filter Portlet needs to specify suported custom fields	↑	Closed
JRA-15445	RPC plugin needs to be cleaned up		Resolved
JRA-15484	Tokenizing java exceptions fails if the exception is terminated with a full-stop.	↓	Resolved
JRA-15517	Upgrade JIRA to use the latest version of Lucene indexing framework - v2.3.2		Resolved
JRA-15543	Show release date next to version name in the list of versions on Browse Project screen		Resolved
JRA-15545	Export issues to Excel/Word format with non-ASCII filter name does not handle the file name properly	↓	Resolved
JRA-15546	Versions no longer display descriptions when browsing project	↑	Resolved
JRA-15548	If an attempt to get the Index lock times out, the indexing operation is discarded	↑	Resolved
JRA-15564	JIRA displays error after a reindex.	↑	Resolved
JRA-15575	Test and confim JIRA is compatible with Microsoft SQL Server 2008		Closed
JRA-15625	It is possible to disable plugins that then render jira incapable of restarting	↑	Resolved
JRA-15631	jelly with invalid variables script returns blank page to user	↓	Resolved
JRA-15638	The new dropdown does not appear to render correctly under IE6/7	↓	Resolved
JRA-15646	Convert JIRA to jQuery		Resolved
JRA-15649	Sort list of plugins in Admin section alphabetically		Resolved
JRA-15665	Address issue of plugins pushing filter/report panel off screen		Resolved
JRA-15666	Add project information to the issue XML view		Resolved
JRA-15669	Drag and drop behaviour doesn't work on Modify Issues Type Scheme page	↑	Resolved
JRA-15700	Created VS Resolved cumulative + individual graphs' Y axis should should be independent		Resolved

JRA-15702	Migrate to licensing 2.0		Resolved
JRA-15723	Jelly AddComment tag changes the "updated" issue timestamp to execution script timestamp		Resolved
JRA-15732	Update email documentation to highlight that OutOfMemoryError can stop email processing		Closed
JRA-15761	If issue key contains unicode characters, redirect on create issue doesn't work		Resolved
JRA-15846	Allow changing license on Data Import in new JIRA instance.		Resolved
JRA-15872	"Browse Project" URL doesn't include current project		Resolved
JRA-15882	Notification Schemes image highlights the wrong function		Closed
JRA-15886	Add logging notification for index optimization events		Resolved
JRA-15920	Include warning in EAR/WAR documentation not to edit anything directly on the application server		Closed
JRA-15927	Code samples on Web UI Plugin Module document are incorrect		Closed
JRA-15962	Upgrade JIRA to Plugins 2.x		Resolved
JRA-15975	Couldn't log in separate log files if more then one instance of JIRA started on Tomcat		Resolved
JRA-15991	Merge translations files into one to make translating JIRA easier!		Resolved
JRA-16058	Aggressive locking in JiraCachingPropertySet causes high contention		Resolved
JRA-16067	Provide field definition in XML issue view URL to customize XML view		Resolved
JRA-16074	Incorrect error warning message on navigator summary		Resolved
JRA-16080	1px offset in Firefox		Resolved
JRA-16088	Created VS Resolved Issues Report contains Old Filtter / Project Picker		Resolved
JRA-16112	Bug in progressWorkflowAction method in SOAP		Resolved
JRA-16113	Do not show negative values on Y axis in Created vs. Resolved chart		Resolved
JRA-16120	Dashboard rewrite		Resolved
JRA-16122	HTTP Basic auth should be enabled by default		Resolved
JRA-16138	Anonymous users should not be considered to "own" all anonymous comments.		Resolved
JRA-16151	Colon : in custom field search causes searching of wrong field		Resolved
JRA-16175	JIRA issues macro does not work with 4.0-m1 on EACJ		Resolved
JRA-16210	Display issue count on JQL execution		Resolved
JRA-16211	Enter / return should execute JQL		Resolved

JRA-16253	Source Build documentation is out of date and incomplete		Closed
JRA-16276	Adjust colours of Resoltion date chart to be more distinshable for the colour blind		Resolved
JRA-16278	Add ability to search for versions using regex or similar		Resolved
JRA-16316	Assigned To Me Portlet, selecting all columns to display causes error		Resolved
JRA-16339	The "Perm Gen" memory usage shown on the System Info page is incorrect.		Resolved
JRA-16351	Component plugin modules don't show up in plugins admin section in JIRA 4.0		Resolved
JRA-16363	ServiceProxyDestroyedException when you reactivate an OSGi plugin		Resolved
JRA-16379	Weblogic Deployment descriptor (weblogic.xml) has changed for Weblogic 9.x		Closed
JRA-16407	JiraModuleDescriptorFactory doesn't define some plugins2 descriptors		Resolved
JRA-16424	log4j output should contain more information		Resolved
JRA-16443	Create a jira.field.resolution.include transition attribute		Resolved
JRA-16451	JIRA home directory created in working directory in JIRA standalone		Resolved
JRA-16485	The long component name, on clicking overlaps the UI element		Resolved
JRA-16498	Version/s and Component/s not validated when updating an issue		Resolved
JRA-16502	Local helper has invalid HTML, causing styling issues		Resolved
JRA-16508	no attachments are returned when 'field=attachment' is specified in XML view		Resolved
JRA-16509	Check for javascript enabled in browser		Resolved
JRA-16510	Update to jQuery 1.3.2		Resolved
JRA-16518	Update readme.txt in war		Closed
JRA-16522	Searching according to multiple assignees should be provided.		Resolved
JRA-16526	Catalan translations needs to be changed		Resolved
JRA-16527	Indexing fails (or atleast error thrown) if index location is pointing to an invalid reference		Resolved
JRA-16529	Project Avatars: User is unable to upload PNG and JPEG file for project avatars on IE 7.0		Resolved
JRA-16530	Project Avatars: User is unable to crop the uploaded image in the project avatars dialog		Resolved
JRA-16531	Project avatar: One image is missing from "Choose an avatar" dialog		Resolved
JRA-16532	User gets a system error if after timeout tries to login from project avatar dialog		Resolved
JRA-16538	User Profile:User Summary area's formatting is not correct when logged in user view profile of other user.		Resolved

JRA-16560	Upgrading bunded plugins fails		Resolved
JRA-16568	Example for xmlrpc client uses a method that has been removed. Need to update		Closed

^Top

EAP Releases

An **Early Access Preview (EAP) release** is a public development release leading up to the official release of a JIRA version. Development releases are a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release.

The following EAP releases are currently available for download. Your help with testing them is very appreciated! Please log the bugs you find on http://jira.atlassian.com in the "JIRA" project.

EAP Releases

- JIRA 5.0 EAP 5 Release Notes
- JIRA 5.0 EAP 4 Release Notes
- JIRA 5.0 EAP 3 Release Notes
- JIRA 5.0 EAP 2 Release Notes
- JIRA 4.4 EAP 6 Release Notes
- JIRA 4.4 EAP 5 Release Notes
- JIRA 4.4 EAP 4 Release Notes
- JIRA 4.4 EAP 3 Release Notes
- JIRA 4.4 EAP 2 Release Notes
- JIRA 4.3 EAP 5 Release Notes
- JIRA 4.3 EAP 3 Release Notes
- JIRA 4.3 EAP 2 Release Notes
- JIRA 4.3 EAP 1 Release Notes
- JIRA 4.2 EAP 4 Release Notes
- JIRA 4.2 EAP 3 Release Notes
- JIRA 4.2 EAP 2 Release Notes
- JIRA 4.2 EAP 1 Release Notes
- JIRA 3.11 EAP Release Notes

Beta Releases

Don't forget to check our JIRA Beta Releases too, which are more mature versions of the EAP releases above.

Latest Beta Release:

- JIRA 5.0 RC 3 Release Notes

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

JIRA 5.0 EAP 5 Release Notes

15 September 2011

JIRA 5.0 EAP 5 (a.k.a 5.0 milestone 5 or 'm5') is a public development release leading up to **JIRA 5.0**. An **Early Access Preview (EAP)**

release is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 5.0 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

A lot of the features in JIRA 5.0 focus on making JIRA easier to use and manage. With JIRA 5.0, you can link JIRA issues to other applications, search for issues whose fixVersion WAS a particular value at some point in the past, and administrators can manage shared filters and dashboards that were created by other people and issues can be copied between different JIRA sites.

There are a large number of improvements for the JIRA developer community (and more to come in future EAPs). In JIRA 5.0 EAP 5, new REST APIs have been added to create issues, a stable JIRA API is undergoing refinement and every block area on the 'View Issue' page is now a Web Panel.

Highlights of JIRA 5.0 EAP 5:

- Remote Issue Linking ***(improved in EAP 5)***
- More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements ***(improved in EAP 5)***
- Activity Streams Now Show External Content
- Manage Other Users' Shared Filters and Dashboards
- Administration User Interface Improvements ***(new in EAP 5)***
- REST API Improvements
- Java API Improvements
- Performance Improvements
- New Troubleshooting and Debugging Tools ***(new in EAP 5)***
- *New Plugin to Try Out* — Remote Issue Copying
- Other Enhancements and Fixes

Thank you for your interest in JIRA 5.0 EAP 5
Download EAP

Upgrading to JIRA 5.0 EAP 5

JIRA EAP releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Do not use in production

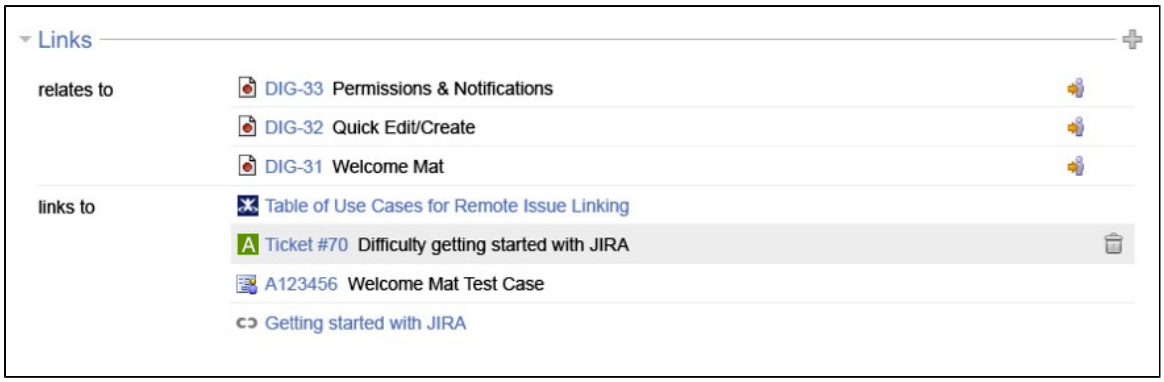
- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 5.0 EAP 5

Remote Issue Linking *(improved in EAP 5)*

The Remote Issue Linking feature provides a powerful way to link JIRA issues to another application. Along with a Java and REST API to add links, end users can also add a link in a JIRA issue to any URL, such as some documentation on another site, a technical note, or any relevant web page or URL.

For more information about this feature, see JIRA Remote Issue Links.



^Top

More Enhancements to JQL — New "CHANGED" Operator and "WAS" Improvements *(improved in EAP 5)*

Enhancements were introduced in JIRA 4.4 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

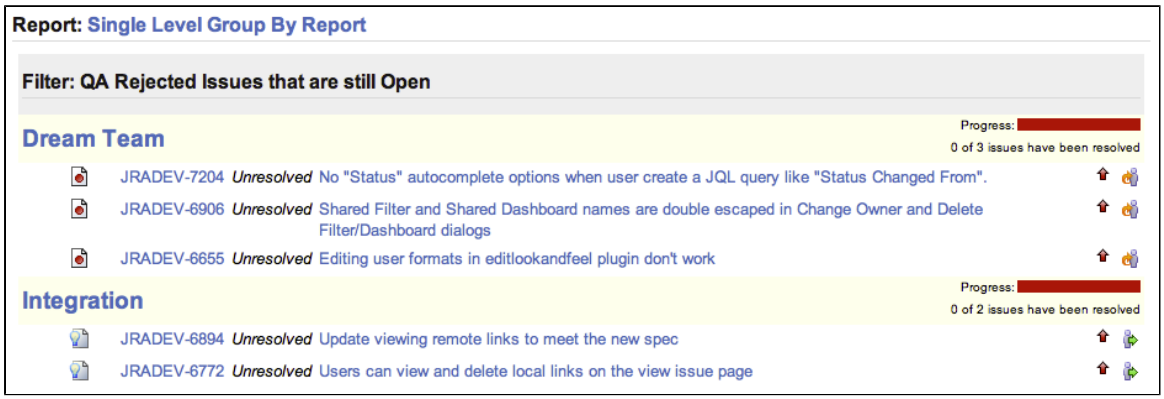
In JIRA 5.0, JQL supports the new "CHANGED" operator, which can accept the optional predicates "FROM", "TO", "ON", "DURING", "BEFORE", "AFTER" and "BY".

For example, the following JQL query:

```
status CHANGED FROM "In Progress" TO "Open" BY pslade BEFORE endOfWeek() AFTER startOfWeek()
```

Will find any issues whose **Status** field value was at some point "In Progress" but changed to "Open", by user 'pslade', and after the start and before the end of the current week.

The "CHANGED" operator can be used on the **Status**, **Assignee**, **Priority**, **Reporter**, **Resolution** and **Fix Version** fields.



The "WAS" operator can be used on the **Fix Version** field too. For example, the following JQL query:

```
fixVersion WAS 4.4
```

Will find any issues whose **Fix Version** field was at some point (or currently is) set to 4.4.

^Top

Activity Streams Now Show External Content

We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Plugin Developer Notes for JIRA 5.0 for more details.

^Top

Manage Other Users' Shared Filters and Dashboards

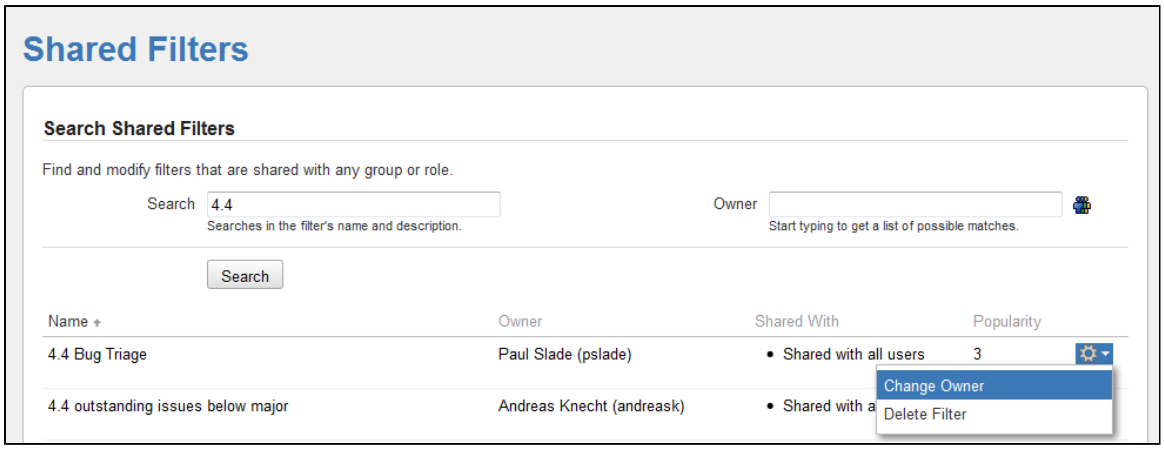
JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboards. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard:



^Top

Administration User Interface Improvements *(new in EAP 5)*

Following on with improvements to the Administration User Interface (UI) in JIRA 4.4, JIRA 5.0 will provide further improvements to the Administration UI by converting various forms and pages in this area to convenient dialog boxes.

In JIRA 5.0 EAP 5, the 'Create New User' form, accessed by clicking 'Add User' on the 'User Browser' page of JIRA's Administration area, is now a dialog box.

^Top

REST API Improvements

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues.
- Retrieve metadata for editing existing issues.
- Delete existing issues and their subtasks.
- Create remote issue links.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2'.

Refer to the Plugin Developer Notes for JIRA 5.0 for more details.

^Top

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Please see the Plugin Developer Notes for JIRA 5.0 for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

^Top

New Troubleshooting and Debugging Tools *(new in EAP 5)*

JIRA 5.0 adds several tools to help Administrators debug the configuration of their instance.

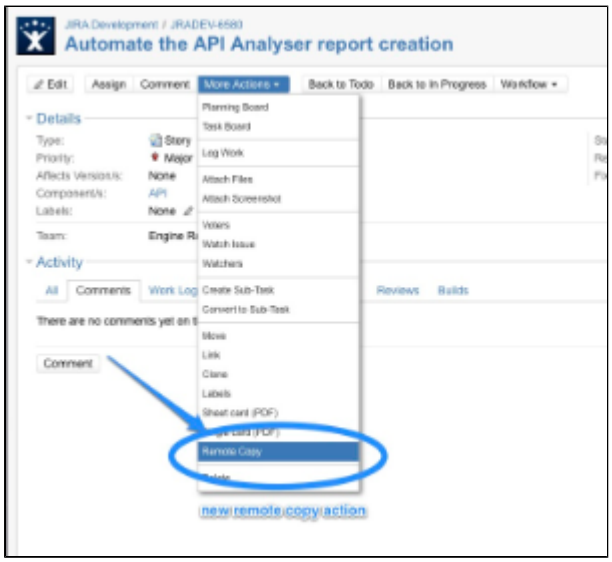
- A number of email debugging tools are now provided to System Administrators in the new 'Logging and Profiling' page under 'Troubleshooting and Support'
    - Enable or disable mail logging
    - Turn debug mail logging on or off
    - Configure a logging level for a new package easily in the default loggers section.
- For testing and troubleshooting LDAP connections, much more comprehensive testing is now provided, including basic connections, user retrieval, user membership, group retrieval, group membership and authentication.

^Top

New Plugin to Try Out — Remote Issue Copying

This new JIRA 5.0-compatible feature, currently undergoing development as a plugin, allows you to copy issues from one JIRA site to another.

Once you have an Application Link established between your JIRA site and another, a new issue action 'Remote Copy' will appear in the view issue page. You can limit this action to a particular user group, but by default everyone can use it.



You will be prompted to map field values by field names for JIRA's built-in (system) fields and/or to configure default values for required fields.
You will require the appropriate permissions to set the field value on the target site.

Custom fields are generally supported, although so far, we have only provided a mapper for the SelectCFType custom field type. Supporting more custom fields is a matter of writing more mappers (which we intend to make pluggable for the final JIRA 5.0 release).

The Remote Issue Copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between. You also need to configure the following before you can copy issues between your JIRA sites:

1. An Application Link from the source JIRA site to the target JIRA site — see Adding an Application Link.
2. A Project Link from the source JIRA Project to the target JIRA Project — Adding Project Links between Applications.

The Remote Issue Copying plugin is not yet bundled with JIRA. However, you can download it from from the following link:

- https://maven.atlassian.com/content/groups/public/com/atlassian/cpji/cpji-jira-plugin/2.1-beta1/cpji-jira-plugin-2.1-beta1.jar

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

JIRA 5.0 EAP 4 Release Notes

6 September 2011

JIRA 5.0 EAP 4 (a.k.a 5.0 milestone 4 or 'm4') is a public development release leading up to **JIRA 5.0**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 5.0 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

A lot of the features in JIRA 5.0 focus on making JIRA easier to use and manage. With JIRA 5.0, it is now possible to link JIRA issues to other applications, you can search for issues whose fixVersion WAS a particular value at some point in the past, administrators can manage shared filters and dashboards that were created by other people and issues can be copied between different JIRA sites.

There are a large number of improvements for the JIRA developer community (and more to come in future EAPs). In JIRA 5.0 EAP 4, new

REST APIs have been added to create issues, a stable JIRA API is undergoing refinement and every block area on the 'View Issue' page is now a Web Panel.

Highlights of JIRA 5.0 EAP 4:

- Remote Issue Linking ***(new in EAP 4)***
- More Enhancements to the JQL "WAS" Function ***(new in EAP 4)***
- Activity Streams Now Show External Content ***(improved since EAP 3)***
- Manage Other Users' Shared Filters and Dashboards
- REST API Improvements
- Java API Improvements
- Performance Improvements
- *New Plugin to Try Out* — Remote Issue Copying
- Other Enhancements and Fixes

Thank you for your interest in JIRA 5.0 EAP 4
Download EAP

Upgrading to JIRA 5.0 EAP 4

JIRA EAP releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Do not use in production

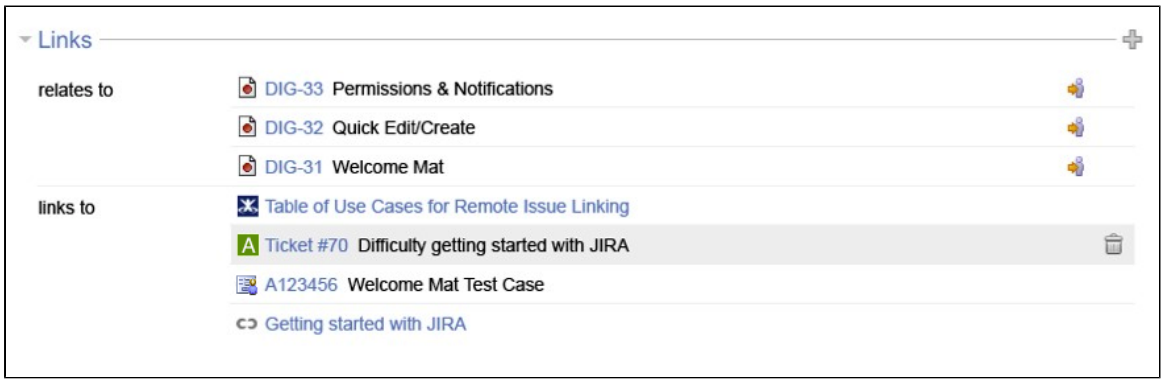
- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 5.0 EAP 4

Remote Issue Linking *(new in EAP 4)*

The Remote Issue Linking feature provides a powerful way to link JIRA issues to your application.

For more information about this feature, see JIRA Remote Issue Links.



^Top

More Enhancements to the JQL "WAS" Function *(new in EAP 4)*

Enhancements were introduced in JIRA 4.4 that allowed you to search the history of an issue's **Assignee** and **Reporter** fields.

In JIRA 5.0, you can now also search the history of the **fixVersion** field, e.g.

```
fixVersion WAS 4.4
```

This will find any issues whose **fixVersion** field was at some point (or currently is) set to 4.4.

^Top

Activity Streams Now Show External Content *(improved since EAP 3)*

We are expanding the Activity Stream features introduced in JIRA 4.4 with:

- The ability to show external activity from another Atlassian application (such as Confluence, FishEye/Crucible and Bamboo) via an Application Link.
- The ability to combine this external activity into an Activity Stream gadget on a JIRA dashboard.
- An API for creating entries in activity streams from remote applications via the REST API or locally via Java.

Refer to the Plugin Developer Notes for JIRA 5.0 for more details.

^Top

4

Manage Other Users' Shared Filters and Dashboards

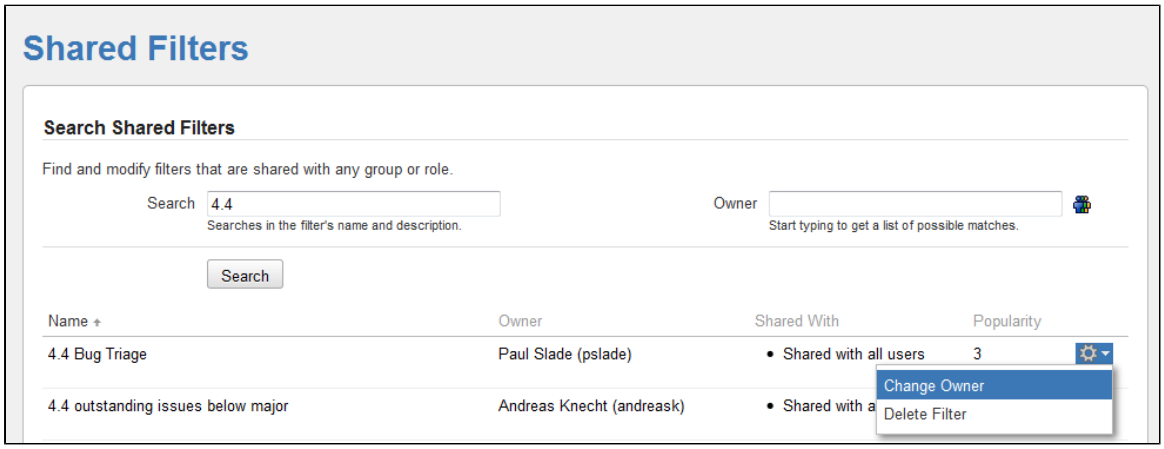
JIRA administrators have the ability to change the ownership of or delete other user's shared filters and dashboards. A shared filter or dashboard is a filter/dashboard created (and hence, owned) by a user, which the user has then shared with others.

Since JIRA only allows the editing or modification of shared filters/dashboards by their owners, this new JIRA feature is especially helpful in situations where a user has left an organisation, but the shared filters or dashboards they created continue to be used by others within the organisation.

Only users with the **Create Shared Objects** global permission can share their filters and dashboards with other JIRA users.

You can access these features by selecting **'Administration' > 'Users' > 'Shared Filters'** or **'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'** or **'shared dashboard'**).

On the 'Shared Filters' or 'Shared Dashboard' pages, you can search for any shared filters/dashboards, or use the cog icon to change the owner of a shared filter/dashboard to another user or delete the shared filter/dashboard:



^Top

REST API Improvements

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues.
- Retrieve metadata for editing existing issues.
- Delete existing issues and their subtasks.
- Create remote issue links.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2'.

Refer to the Plugin Developer Notes for JIRA 5.0 for more details.

^Top

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Please see the Plugin Developer Notes for JIRA 5.0 for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

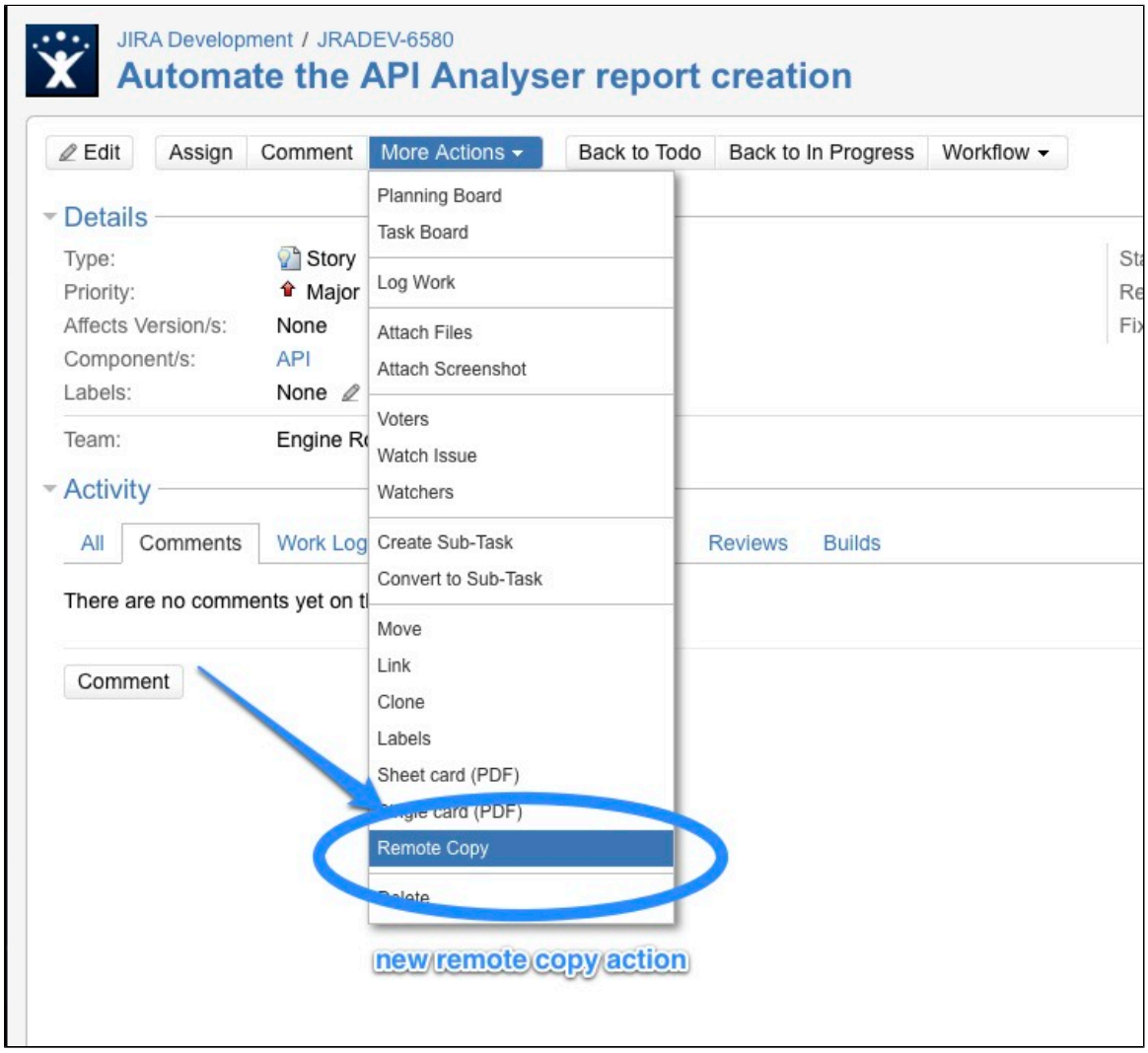
^Top

8

New Plugin to Try Out — Remote Issue Copying

This new JIRA 5.0-compatible feature, currently undergoing development as a plugin, allows you to copy issues from one JIRA site to another.

Once you have an Application Link established between your JIRA site and another, a new issue action 'Remote Copy' will appear in the view issue page. You can limit this action to a particular user group, but by default everyone can use it.



You will be prompted to map field values by field names for JIRA's built-in (system) fields and/or to configure default values for required fields.
You will require the appropriate permissions to set the field value on the target site.

Custom fields are generally supported, although so far, we have only provided a mapper for the SelectCFType custom field type. Supporting more custom fields is a matter of writing more mappers (which we intend to make pluggable for the final JIRA 5.0 release).

The Remote Issue Copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between. You also need to configure the following before you can copy issues between your JIRA sites:

1. An Application Link from the source JIRA site to the target JIRA site — see Adding an Application Link.
2. A Project Link from the source JIRA Project to the target JIRA Project — Adding Project Links between Applications.

The Remote Issue Copying plugin is not yet bundled with JIRA. However, you can download it from from the following link:

- https://maven.atlassian.com/content/groups/public/com/atlassian/cpji/cpji-jira-plugin/2.1-beta1/cpji-jira-plugin-2.1-beta1.jar

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

JIRA 5.0 EAP 3 Release Notes

26 August 2011

JIRA 5.0 EAP 3 (a.k.a 5.0 milestone 3 or 'm3') is a public development release leading up to **JIRA 5.0**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 5.0 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

A lot of the features in JIRA 5.0 focus on making JIRA easier to use and manage. With JIRA 5.0, administrators can manage shared filters and dashboards that were created by other people and issues can be copied between different JIRA sites.

There are a large number of improvements for the JIRA developer community (and more to come in future EAPs). In JIRA 5.0 EAP 3, new REST APIs have been added to create issues, a stable JIRA API is undergoing refinement and every block area on the 'View Issue' page is now a Web Panel.

Highlights of JIRA 5.0 EAP 3:

- Manage Other Users' Shared Dashboards ***(new since EAP 2)***
- Manage Other Users' Shared Filters
- Activity Stream API ***(new since EAP 2)***
- REST API Improvements ***(improved since EAP 2)***
- Java API Improvements
- Performance Improvements ***(improved since EAP 2)***
- *New Plugin to Try Out* — Remote Issue Copying
- Other Enhancements and Fixes

Thank you for your interest in JIRA 5.0 EAP 3
Download EAP

Upgrading to JIRA 5.0 EAP 3

JIRA EAP releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Do not use in production

- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

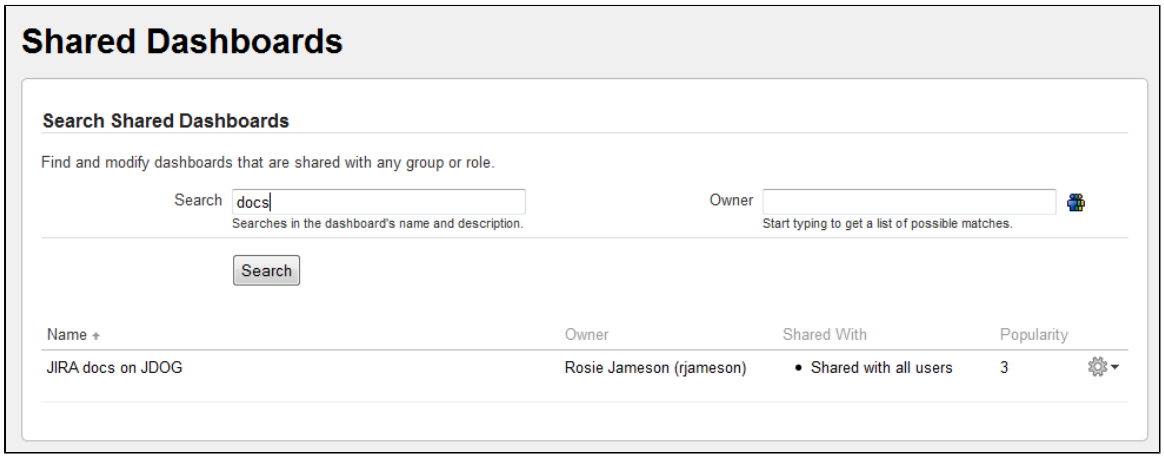
Highlights of JIRA 5.0 EAP 3

Manage Other Users' Shared Dashboards *(new since EAP 2)*

JIRA administrators have the ability to manage other people's shared dashboards. This is especially helpful in situations where a user has

left an organisation, but the shared dashboards they owned continue to be used by others within the organisation.

You can access this feature by selecting **'Administration' > 'Users' > 'Shared Dashboards'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared dashboards'**). On the 'Shared Dashboards' page, you can search for any shared dashboards, or use the cog icon to change the owner of a dashboard to another user or delete a dashboard:



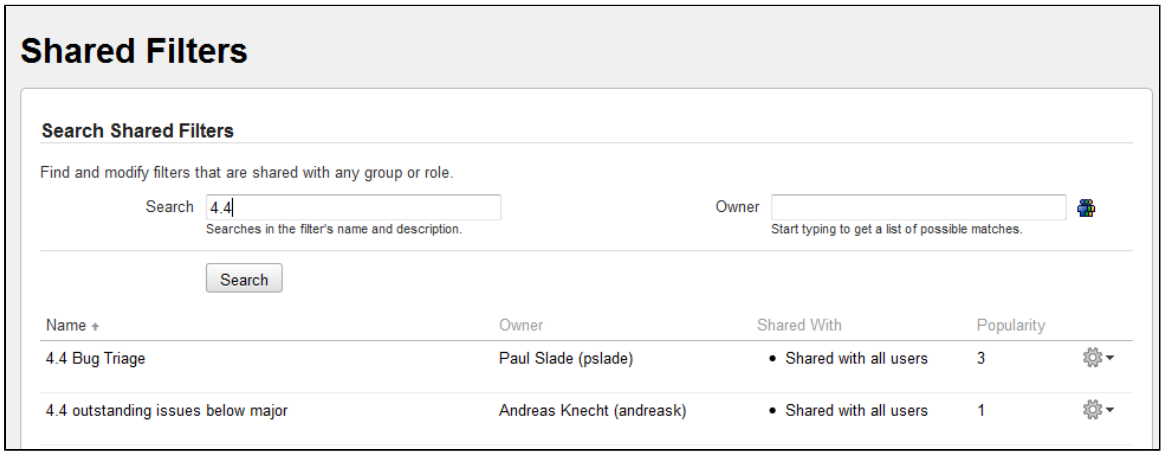
^Top

Manage Other Users' Shared Filters

JIRA also gives administrators the ability to manage other people's shared filters. Like Shared Dashboards (above), this is also useful in situations where a user has left an organisation, but the shared filters they owned continue to be used by others within the organisation.

You can access this feature by selecting **'Administration' > 'Users' > 'Shared Filters'** (or using the keyboard shortcut **'g' + 'g' +** start typing **'shared filters'**).

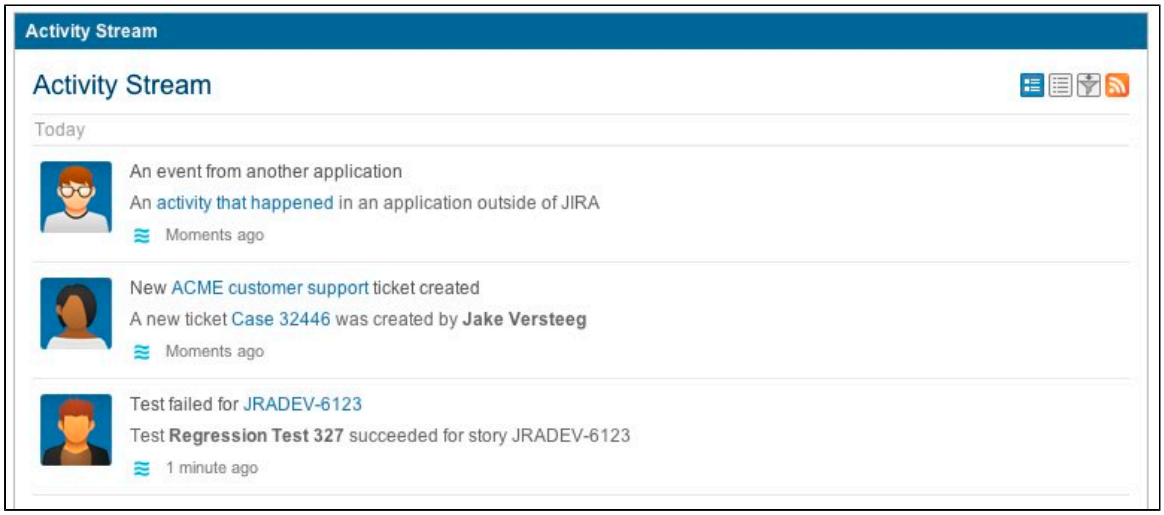
On the 'Shared Filters' page, you can search for any shared filter, or use the cog icon to change the owner of a filter to another user or delete a filter:



^Top

Activity Stream API *(new since EAP 2)*

We are expanding the Activity Stream features introduced in JIRA 4.4 with a new Activity Stream API in JIRA 5.0 that makes it easy for any application to post activities into JIRA's activity streams.



Refer to the Plugin Developer Notes for JIRA 5.0 for more details.

^Top

REST API Improvements *(improved since EAP 2)*

JIRA's REST API is undergoing a significant number of changes and improvements to provide the following:

- Create new issues.
- Retrieve metadata for creating new issues.
- Retrieve metadata for editing existing issues.
- Delete existing issues and their subtasks.
- Create remote issue links.

Please also note that the we have changed the `api-version` name component of URLs for JIRA's REST API calls from '`2.0.alpha1`' to simply '2'.

Refer to the Plugin Developer Notes for JIRA 5.0 for more details.

^Top

Java API Improvements

JIRA's Java API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Please see the Plugin Developer Notes for JIRA 5.0 for more details.

Please also be aware that the JIRA's Java API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements *(improved since EAP 2)*

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

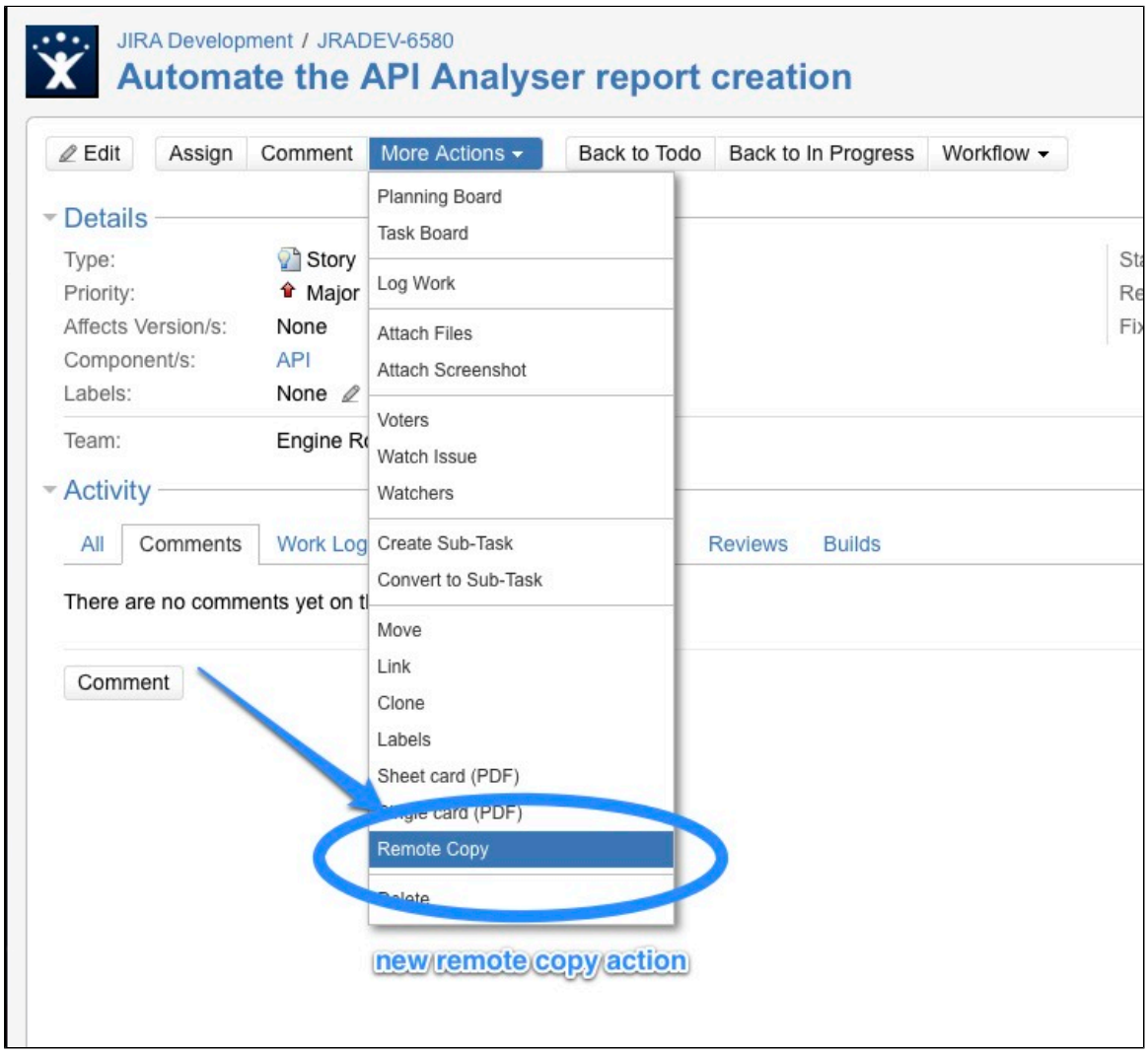
The 'Activity' tabs on the 'View Issue' page are now loaded in the background when this page is first viewed, allowing the information on these tabs to be displayed more rapidly.

^Top

New Plugin to Try Out — Remote Issue Copying

This new JIRA 5.0-compatible feature, currently undergoing development as a plugin, allows you to copy issues from one JIRA site to another.

Once you have an Application Link established between your JIRA site and another, a new issue action 'Remote Copy' will appear in the view issue page. You can limit this action to a particular user group, but by default everyone can use it.



You will be prompted to map field values by field names for JIRA's built-in (system) fields and/or to configure default values for required fields.

You will require the appropriate permissions to set the field value on the target site.

Custom fields are generally supported, although so far, we have only provided a mapper for the SelectCFType custom field type. Supporting more custom fields is a matter of writing more mappers (which we intend to make pluggable for the final JIRA 5.0 release).

The Remote Issue Copy feature is currently available as a plugin that needs to be installed on each JIRA server you wish to copy issues between. You also need to configure the following before you can copy issues between your JIRA sites:

1. An Application Link from the source JIRA site to the target JIRA site — see Adding an Application Link.
2. A Project Link from the source JIRA Project to the target JIRA Project — Adding Project Links between Applications.

The Remote Issue Copying plugin is not yet bundled with JIRA. However, you can download it from from the following link:

- https://maven.atlassian.com/content/groups/public/com/atlassian/cpji/cpji-jira-plugin/2.1-beta1/cpji-jira-plugin-2.1-beta1.jar

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

JIRA 5.0 EAP 2 Release Notes

10 August 2011

JIRA 5.0 EAP 2 (a.k.a 5.0 milestone 2 or 'm2') is a public development release leading up to **JIRA 5.0**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

Please note that JIRA 5.0 EAP 1 was not released to the public.

While development work on JIRA 5.0 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

A lot of the features in JIRA 5.0 focus on making JIRA easier to use and manage. With JIRA 5.0, administrators can manage filters that were created by other people.

There are a large number of improvements for the JIRA developer community (and more to come in future EAPs). In JIRA 5.0 EAP 2, new REST APIs have been added to create issues, a stable JIRA API is being defined and every block area on the 'View Issue' page is now a Web Panel.

Highlights of JIRA 5.0 EAP 2:

- Manage Other Users' Shared Filters
- JIRA 5.0 API Improvements
- Performance Improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 5.0 EAP 2

Download EAP

Upgrading to JIRA 5.0 EAP 2

JIRA EAP releases are available here. When upgrading, please follow the JIRA 5.0 Upgrade Notes.

Do not use in production

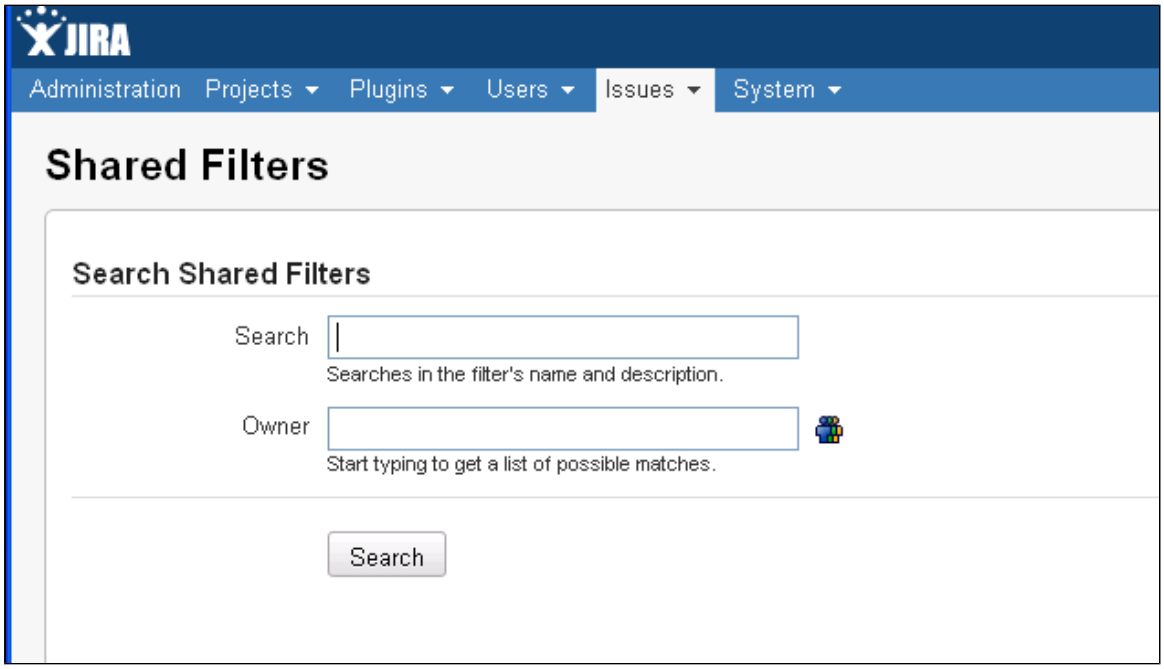
- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 5.0 EAP 2

Manage Other Users' Shared Filters

JIRA 5.0 will give administrators the ability to manage other people's shared filters. This is especially helpful in situations where a user has left an organisation, but the shared filters they owned continue to be used by others within the organisation.

As the first step toward this, EAP 2 provides the ability for JIRA administrators to search for any shared filters:



^Top

JIRA 5.0 API Improvements

For plugin developers, JIRA's API is undergoing a significant number of changes and improvements to provide the following:

- More stability and reliability with future versions of JIRA.
- A more functional REST API with the ability to create new issues in JIRA 5.0 EAP 2.
- Removal of deprecated OSUser classes.
- Removal of deprecated portlets (replaced by gadgets in JIRA 4.0) and their related APIs.

Please see the Plugin Developer Notes for JIRA 5.0 for more details.

Please also be aware that JIRA's API is likely to undergo a rapid number of changes from one JIRA 5.0 EAP release to the next.

^Top

Performance Improvements

Lucene 3.2 is now fully integrated into JIRA. Initial benchmarking shows performance improvements across a number of JIRA features.

^Top

Other Enhancements and Fixes

For a list of more issues resolved in JIRA 5.0 so far, click here.

^Top

JIRA 4.4 EAP 6 Release Notes

24 May 2011

JIRA 4.4 EAP 6 (a.k.a 4.4 milestone 6 or 'm6') is a public development release leading up to **JIRA 4.4**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 4.4 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer. We are also very pleased to announce that the JIRA installation process has been improved and simplified.

Note to developers: Much of the View Issue page is now rendered via Web Panel Plugin Modules. Also please see the Plugin Developer Notes for JIRA 4.4.

Highlights of JIRA 4.4 EAP 6:

- Visual Workflow Designer For Administrators
- Workflow Viewer on the 'View Issue' Screen
- User Time Zones ***(improved since EAP 5)***
- Editable Options for Custom Fields
- Issue Linking when Resolving an Issue
- Multiple File Selection and Upload from the 'File Upload' Dialog Box
- New-Look Administration Area ***(new since EAP 5)***
- Improved Setup Wizard
- Database Configuration Now Part of the Setup Wizard
- Improved JIRA Standalone Linux Installer with Uninstall and Upgrade Capabilities ***(improved since EAP 5)***
- Improved JIRA Standalone Windows Installer with Upgrade Capabilities ***(improved since EAP 5)***
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 EAP 6
Download EAP

Upgrading to JIRA 4.4 EAP 6

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

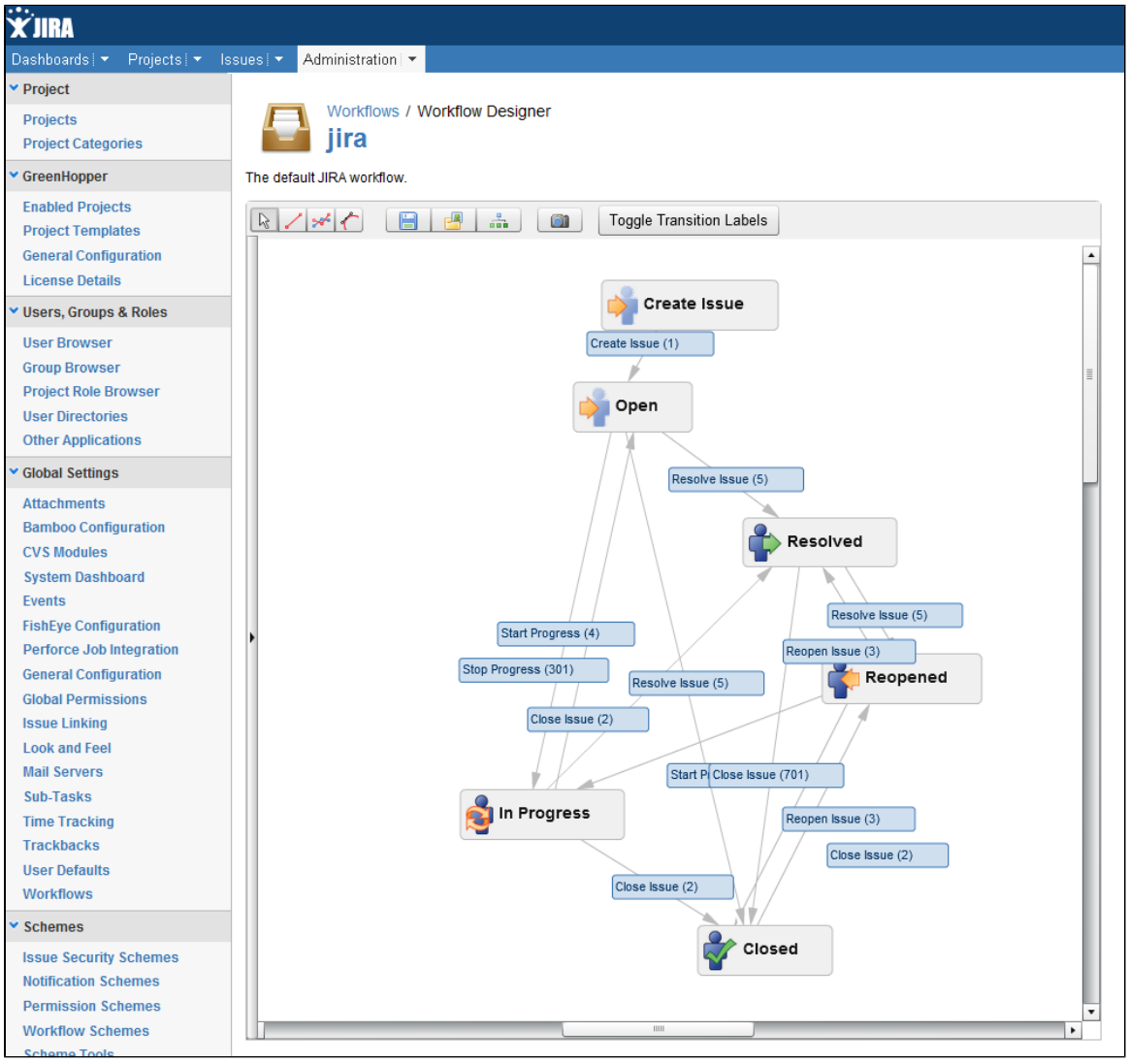
Highlights of JIRA 4.4 EAP 6

Visual Workflow Designer For Administrators

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to '**Workflows**' in JIRA administration as usual, and click the '**Design**' link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.

In EAP 5, the layout of a workflow is now preserved whenever you 'Copy' or 'Create a Draft' of an existing workflow. We have also implemented a few improvements to the status editor.



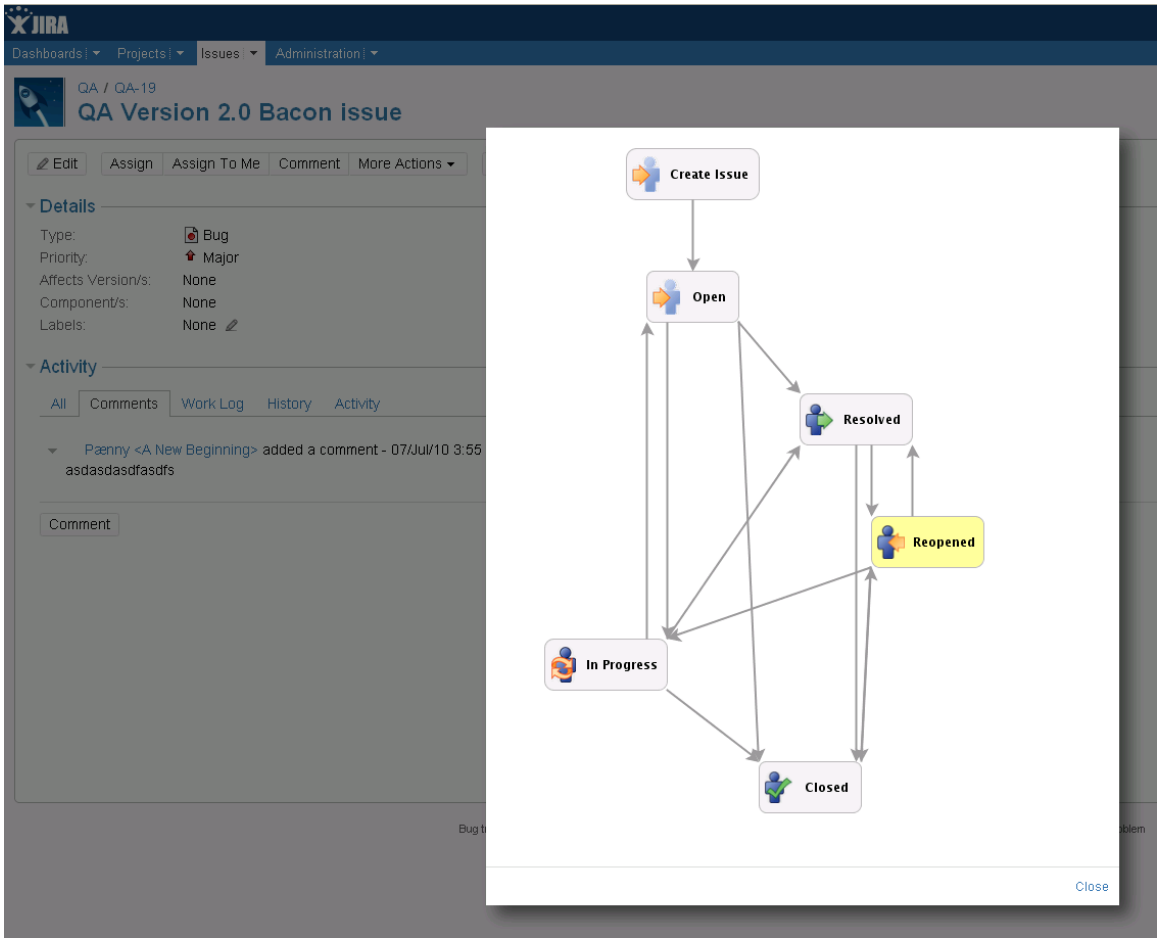
^Top

Workflow Viewer on the 'View Issue' Screen

You can also see a read-only view of the workflow from the 'View Issue' page — just select '**View Workflow**' from the '**More Actions**' menu. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View Read-Only Workflow' is required to access the workflow viewer feature from the 'View Issue' page.



^Top

User Time Zones *(improved since EAP 5)*

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: times will be displayed to a user in their local time zone, rather than the server's time zone.

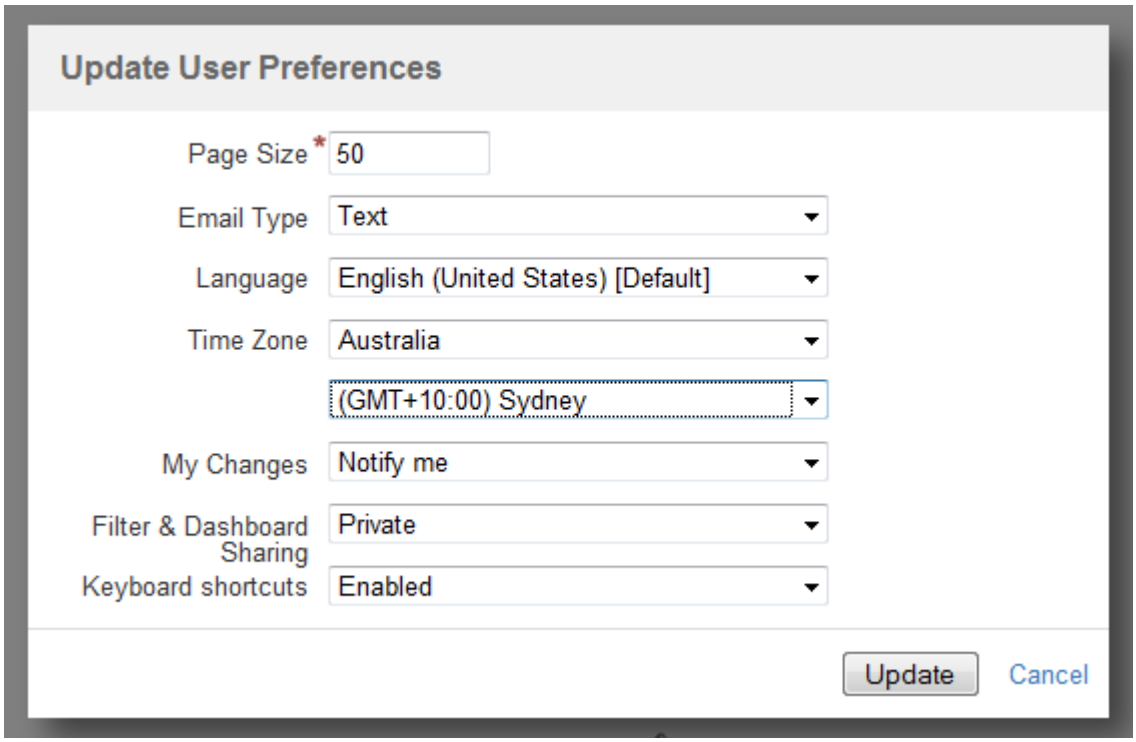
You can set a default user time zone at an administration level, and individual users have the ability to choose their own time zone. In EAP 6 each user's time zone is displayed in their hover profile.

Time zone support has been implemented for quick searching, simple searching and advanced searching, chart and report gadgets, date/time-based custom fields, as well as issue histories, work logs and source code check-ins (via the JIRA FishEye Plugin).

Note: Date fields, which have no time component, such as due dates, release dates (associated with versions) and custom date fields, solely record date information (and no time zone-related information).

JIRA administrators can change the default time zone by going to '**Administration**' => '**General Configuration**' (under '**Global Settings**'), and editing the '**Default user time zone**'.

Users can also change their individual time zone setting via their user profile:



Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

^Top

Editable Options for Custom Fields

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

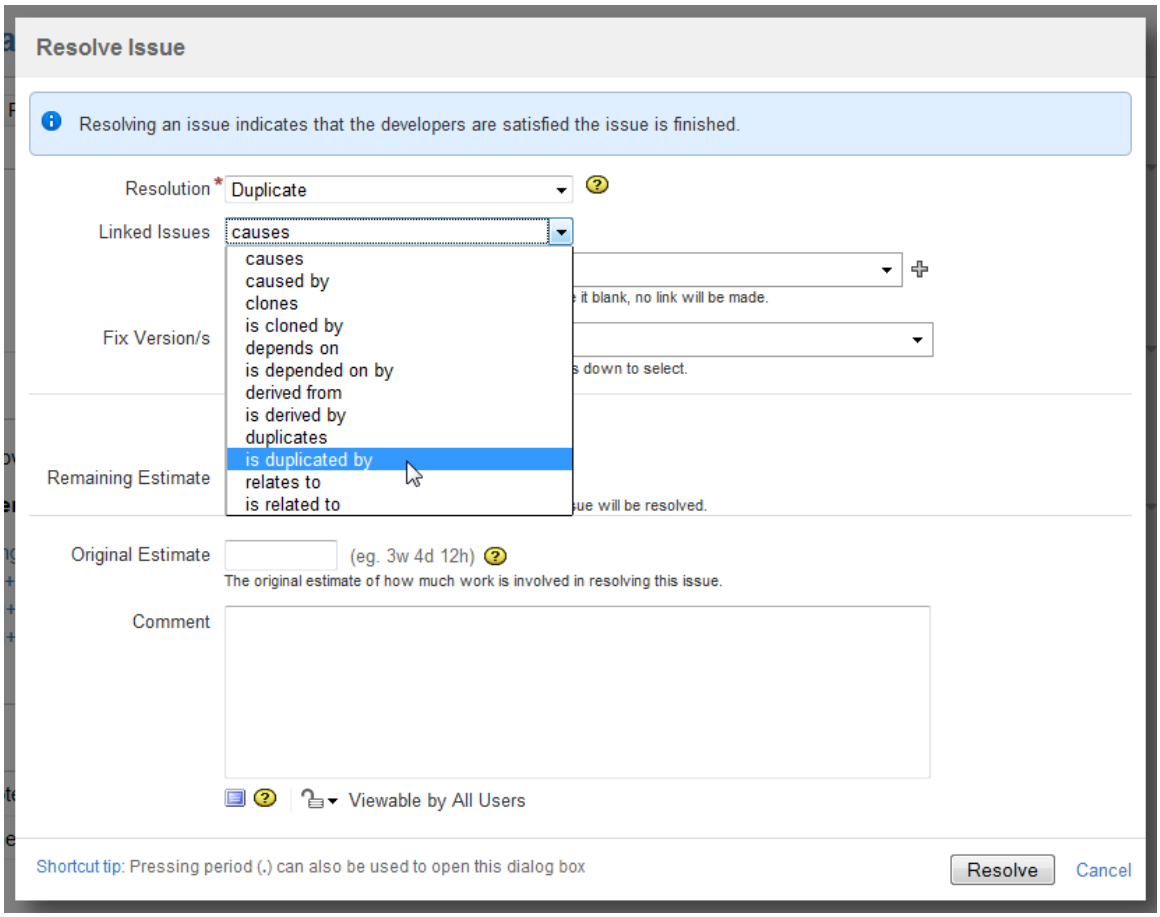
^Top

Issue Linking when Resolving an Issue

When resolving an issue, you can create links to other issues on an issue resolution screen. This is handy when you want to resolve an issue as a 'duplicate' of another *and at the same time* link to the duplicate issue.

For convenience, your last 10 issue links and resolutions are readily accessible from the 'Linked Issues' fields (in the screenshot below). The 'Linked Issues' fields can also be added to any JIRA screen via the new 'Issue Linking' field in JIRA 4.4. See Defining a Screen for more information.

If you are upgrading from an earlier version of JIRA, you need to configure this feature manually through JIRA's administration area. Please refer to the Upgrade Notes for details.

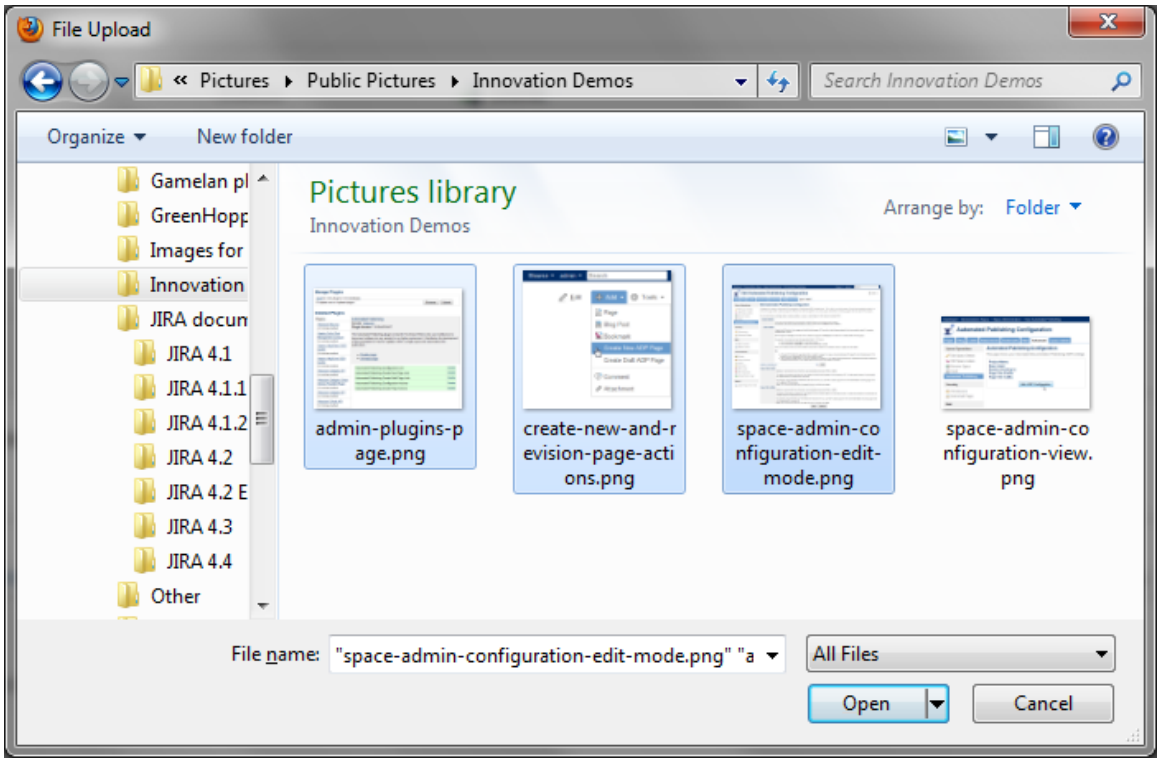


^Top

6

Multiple File Selection and Upload from the 'File Upload' Dialog Box

When using JIRA's 'Attach Files' dialog box, you can now select multiple files in the 'File Upload' dialog box that appears after clicking the 'Browse' button.



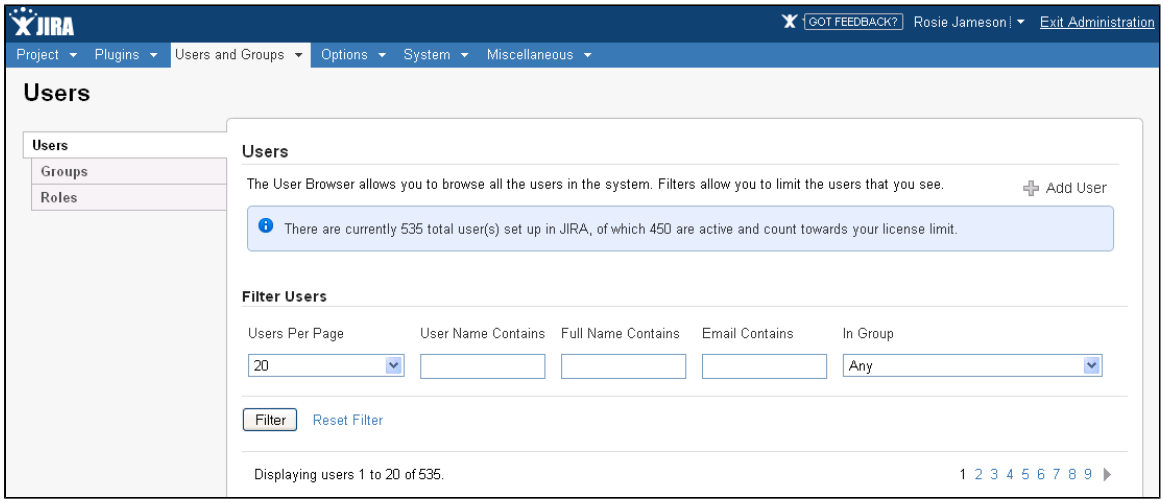
^Top

New-Look Administration Area *(new since EAP 5)*

JIRA 4.4 brings you a dedicated 'administration mode', which replaces the left-hand column of the JIRA's administration console with a series of drop-down menus across the top navigation bar.

To leave JIRA's 'administration mode', click the '**Exit Administration**' link at the top-right of the screen to return JIRA to its standard user mode.

We are still refining this, but here's the first version:



^Top

Improved Setup Wizard

Our trusty wizard has had a makeover:

The new Setup Wizard

^Top

Database Configuration Now Part of the Setup Wizard

In JIRA 4.4, configuring a connection to an external database is now part of the standard setup wizard. Upon completing the setup wizard, JIRA will create a direct JDBC connection (whose entire configuration is stored within your JIRA home directory).

The database configuration step of the setup wizard will change before the final 4.4 release, but here is how it looks so far:

^Top

Improved JIRA Standalone Linux Installer with Uninstall and Upgrade Capabilities *(improved since EAP 5)*

Console Installer

A simple console (command line) wizard is now available for Linux operating systems. The console wizard:

- Installs JIRA under a dedicated user account 'jira' with restricted write access to your JIRA installation directory.
- Can install JIRA as a service, so that JIRA automatically re-starts whenever your Linux operating system must be rebooted.

To install JIRA, simply download the Linux `.bin` installer file and at a shell prompt, execute this file and follow the remaining prompts!

The console wizard can install JIRA as either the 'root' user or a non-root user. However, to install JIRA as a service, the console wizard must be executed as the 'root' user.

See Installing JIRA on Linux for details.

Unattended Installation

After installing JIRA on Linux using the `.bin` installer file above, a configuration file called `response.varfile` is generated in the `.install4j` subdirectory of your JIRA Installation Directory.

See Performing an Unattended Installation (on Linux) for details.

Uninstaller

After installing JIRA on Linux using the `.bin` installer file above, an executable file called `uninstall` (located in your JIRA Installation Directory) is available to conveniently uninstall JIRA.

See Uninstalling JIRA from Linux for details.

Upgrade Feature *(new in EAP 6)*

The console wizard includes an option that allows you to upgrade an existing JIRA Standalone installation from version 4.3.x or later.

While the upgrade feature installs a new version of JIRA, it automates the following tasks for you:

1. Backs up your existing JIRA installation and home directories.
2. Migrates database configurations used in JIRA 4.3.x and earlier to the new database configuration used in JIRA 4.4.
3. Migrates port values in your existing JIRA installation's `server.xml` file to your new version of JIRA.
4. Migrates custom values in your existing JIRA installation's `jira-application.properties` and `setenv.sh` files to your new version of JIRA.
   In the `setenv.sh` file, only the following values are migrated:
   - `JVM_SUPPORT_RECOMMENDED_ARGS`
   - `JVM_MINIMUM_MEMORY`
   - `JVM_MAXIMUM_MEMORY`
   - `JIRA_MAX_PERM_SIZE`

Please Note:

- The upgrade process requests that you conduct a backup of your database using your database's backup utilities. If your database does not support online backups, you can stop the upgrade process, shut down JIRA, perform your database backup and then restart the upgrade process to continue on.
- If you have made customisations to your `seraph-config.xml` file or any other files in your JIRA installation directory, these must be migrated manually.
- If your attachments and index files are located outside your JIRA home directory, then backups of these directories must be performed manually.

^Top

Improved JIRA Standalone Windows Installer with Upgrade Capabilities *(improved since EAP 5)*

The installation wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

See Installing JIRA on Windows for details.

Unattended Installation

When you execute the installer file at the Windows command prompt and complete your installation of JIRA, a file called `response.varfile` is generated in the `.install4j` subdirectory of your JIRA Installation Directory.

See Performing an Unattended Installation (on Windows) for details.

Upgrade Feature *(new in EAP 6)*

The installation wizard includes an option that allows you to upgrade an existing JIRA Standalone installation from version 4.3.x or later.

While the upgrade feature installs a new version of JIRA, it automates the following tasks for you:

1. Backs up your existing JIRA installation and home directories.
2. Migrates database configurations used in JIRA 4.3.x and earlier to the new database configuration used in JIRA 4.4.
3. Migrates port values in your existing JIRA installation's `server.xml` file to your new version of JIRA.
4. Migrates custom values in your existing JIRA installation's `jira-application.properties` and `setenv.bat` files to your new version of JIRA.
   In the `setenv.bat` file, only the following values are migrated:
   - `JVM_SUPPORT_RECOMMENDED_ARGS`
   - `JVM_MINIMUM_MEMORY`
   - `JVM_MAXIMUM_MEMORY`
   - `JIRA_MAX_PERM_SIZE`

Please Note:

- The upgrade process requests that you conduct a backup of your database using your database's backup utilities. If your database does not support online backups, you can stop the upgrade process, shut down JIRA, perform your database backup and then restart the upgrade process to continue on.
- If you have made customisations to your `seraph-config.xml` file or any other files in your JIRA installation directory, these must be migrated manually.
- If your attachments and index files are located outside your JIRA home directory, then backups of these directories must be performed manually.

^Top

REST API improvements

There are new REST APIs for

- Listing and managing Project Components.
- Listing and managing Project Versions.
- Listing and managing Project Roles.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.4 EAP 5 Release Notes

9 May 2011

JIRA 4.4 EAP 5 (a.k.a 4.4 milestone 5 or 'm5') is a public development release leading up to **JIRA 4.4**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 4.4 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer. We are also very pleased to announce that the JIRA installation process has been improved and simplified.

Note to developers: Much of the View Issue page has been converted to Web Panel Plugin Modules — a new type of plugin module in JIRA 4.4. Also please see the Plugin Developer Notes for JIRA 4.4.

Highlights of JIRA 4.4 EAP 5:

- Visual Workflow Designer For Administrators ***(improved since EAP 4)***
- Workflow Viewer on the 'View Issue' Screen
- User Time Zones ***(improved since EAP 4)***
- Editable Options for Custom Fields
- Issue Linking when Resolving an Issue ***(new in EAP 5)***
- Multiple File Selection and Upload from the 'File Upload' Dialog Box ***(new in EAP 5)***
- Improved Setup Wizard
- Database Configuration Now Part of the Setup Wizard
- Improved Linux Installers and Uninstaller
- Improved Windows Installers
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 EAP 5

Download EAP

Upgrading to JIRA 4.4 EAP 5

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.4 EAP 5

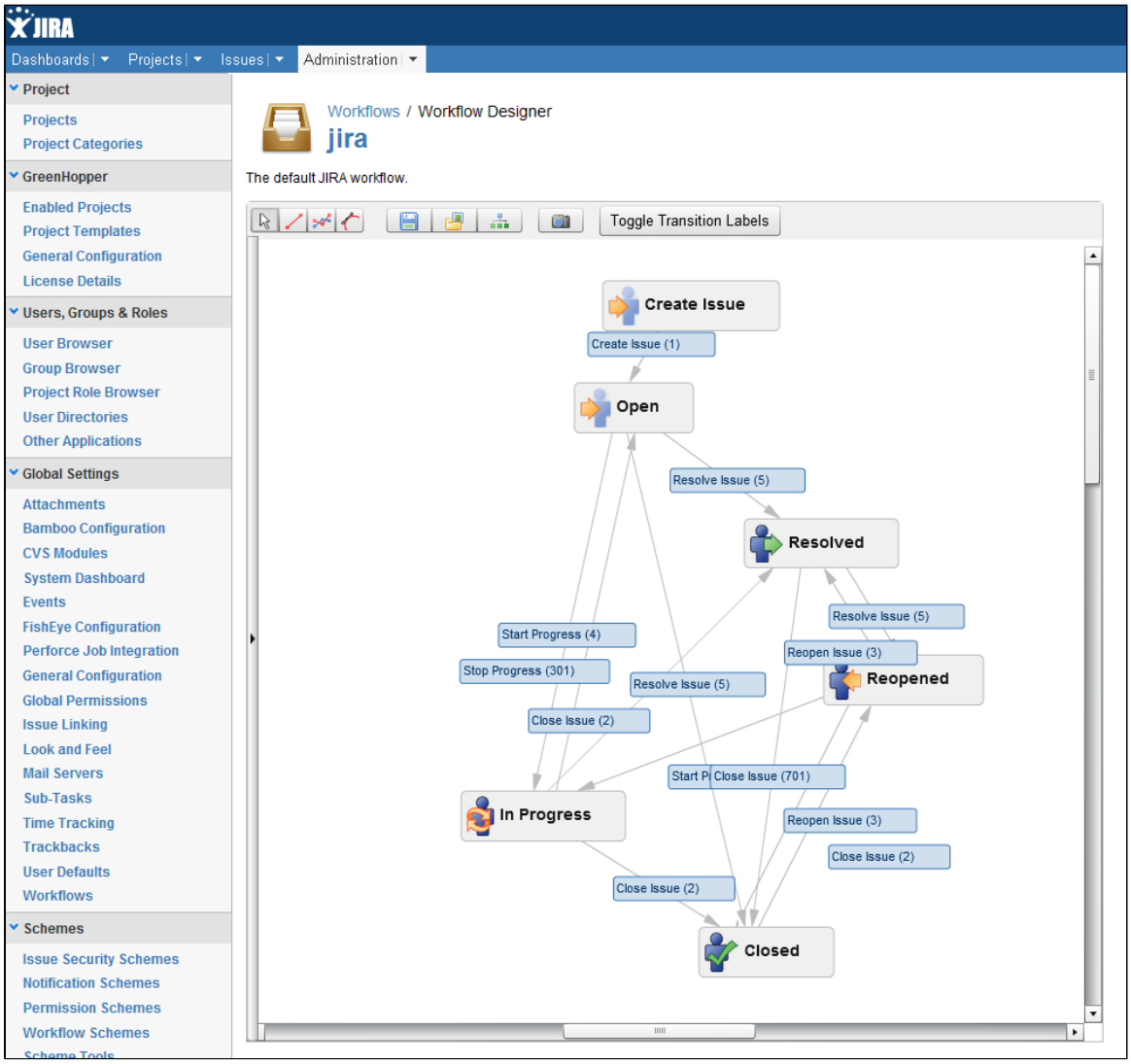
1

Visual Workflow Designer For Administrators *(improved since EAP 4)*

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to '**Workflows**' in JIRA administration as usual, and click the '**Design**' link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.

In EAP 5, the layout of a workflow is now preserved whenever you 'Copy' or 'Create a Draft' of an existing workflow. We have also implemented a few improvements to the status editor.



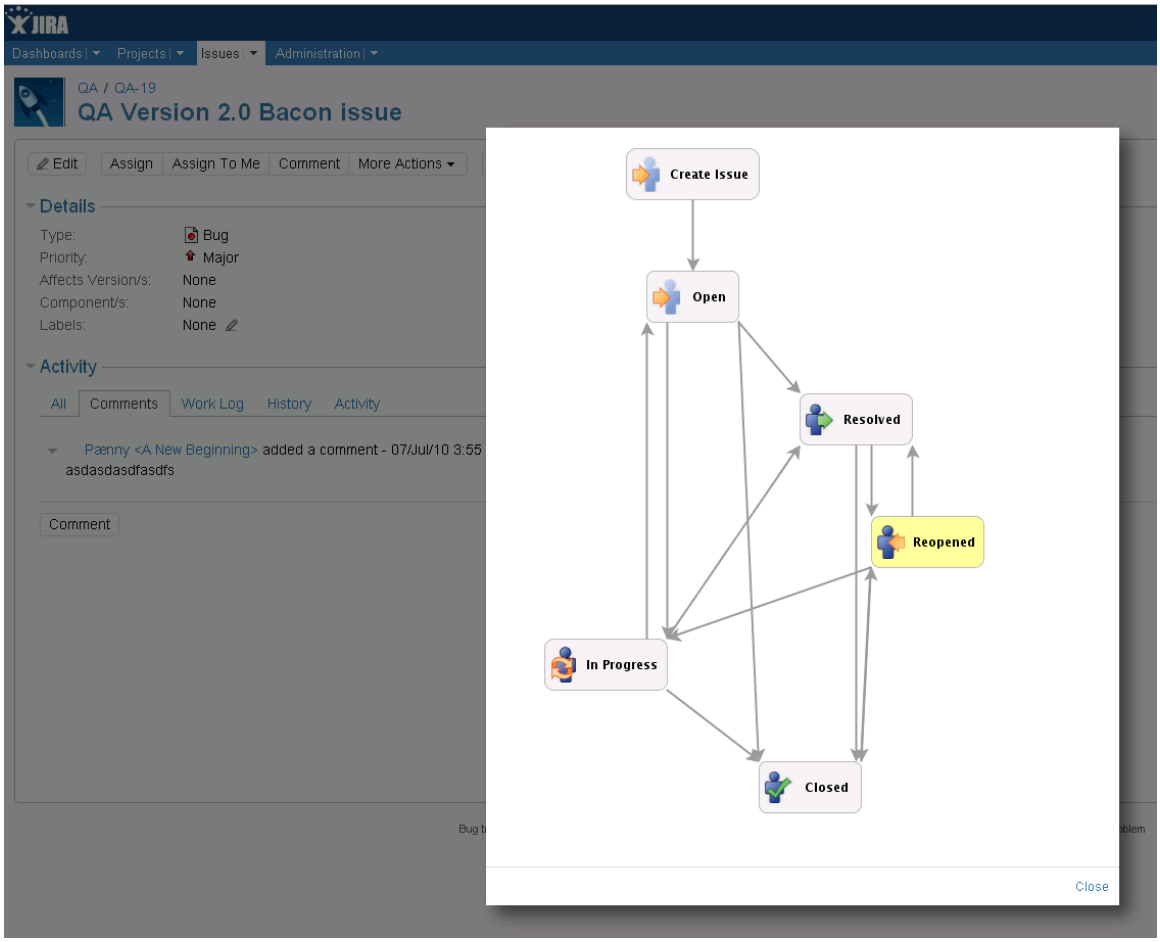
^Top

Workflow Viewer on the 'View Issue' Screen

You can also see a read-only view of the workflow from the 'View Issue' page — just select '**View Workflow**' from the '**More Actions**' menu. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View Read-Only Workflow' is required to access the workflow viewer feature from the 'View Issue' page.



^Top

User Time Zones *(improved since EAP 4)*

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: times will be displayed to a user in their local time zone, rather than the server's time zone.

You can set a default user time zone at an administration level, and individual users have the ability to choose their own time zone.

Time zone support has been implemented for quick searching, simple searching and advanced searching, chart and report gadgets, date/time-based custom fields, as well as issue histories, work logs and source code check-ins (via the JIRA FishEye Plugin).

In EAP 5, all times are now associated with user time zones. Date fields, which have no time component, such as due dates, release dates

(associated with versions) and custom date fields, solely record date information (and no time zone-related information).

Be aware that EAP 5 shipped with a bug (JRA-24525) that affects the date selection calendar. To resolve this bug, please install the patch file (zip archive) attached to this JIRA issue after you install or upgrade to EAP 5. Patch instructions are included in the patch file.

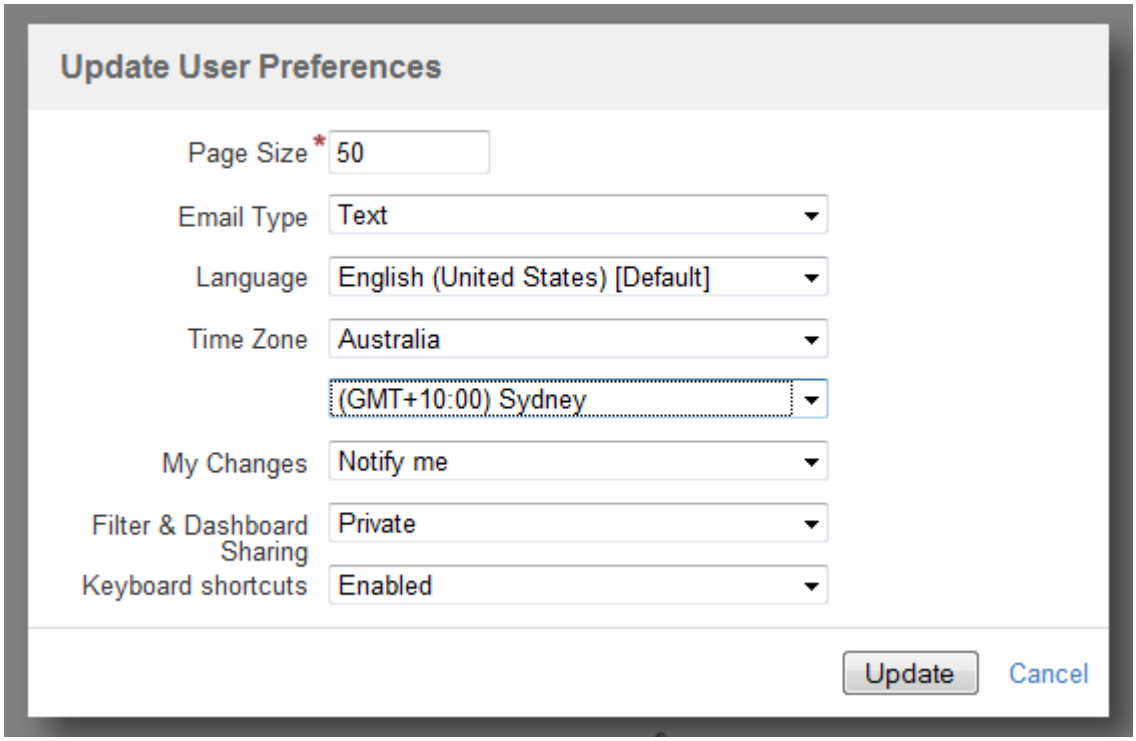
Date/Time Formats

These time and date formats are used throughout JIRA. To specify them, use the format described at http://java.sun.com/j2se/1.4.2/docs/api/java/text/SimpleDateFormat.html.

	Format	Example
Time Format	h:mm a z	2:02 PM EST
Day Format	EEEE h:mm a z	Monday 2:02 PM EST
Complete Date/Time Format	dd/MMM/yy h:mm a z	11/Apr/11 2:02 PM EST
Day/Month/Year Format	dd/MMM/yy z	11/Apr/11 EST
Date Picker Format (javascript format)	d/MMM/yy (%e/%b/%y)	11/Apr/11
Date Time Picker Format (javascript format)	dd/MMM/yy h:mm a (%e/%b/%y %I:%M %p)	11/Apr/11 2:02 PM
Use ISO8601 standard in Date Picker	OFF	

JIRA administrators can change the default time zone by going to '**Administration**' => '**General Configuration**' (under '**Global Settings**'), and editing the '**Default user time zone**'.

Users can also change their individual time zone setting via their user profile:



Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

^Top

Editable Options for Custom Fields

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

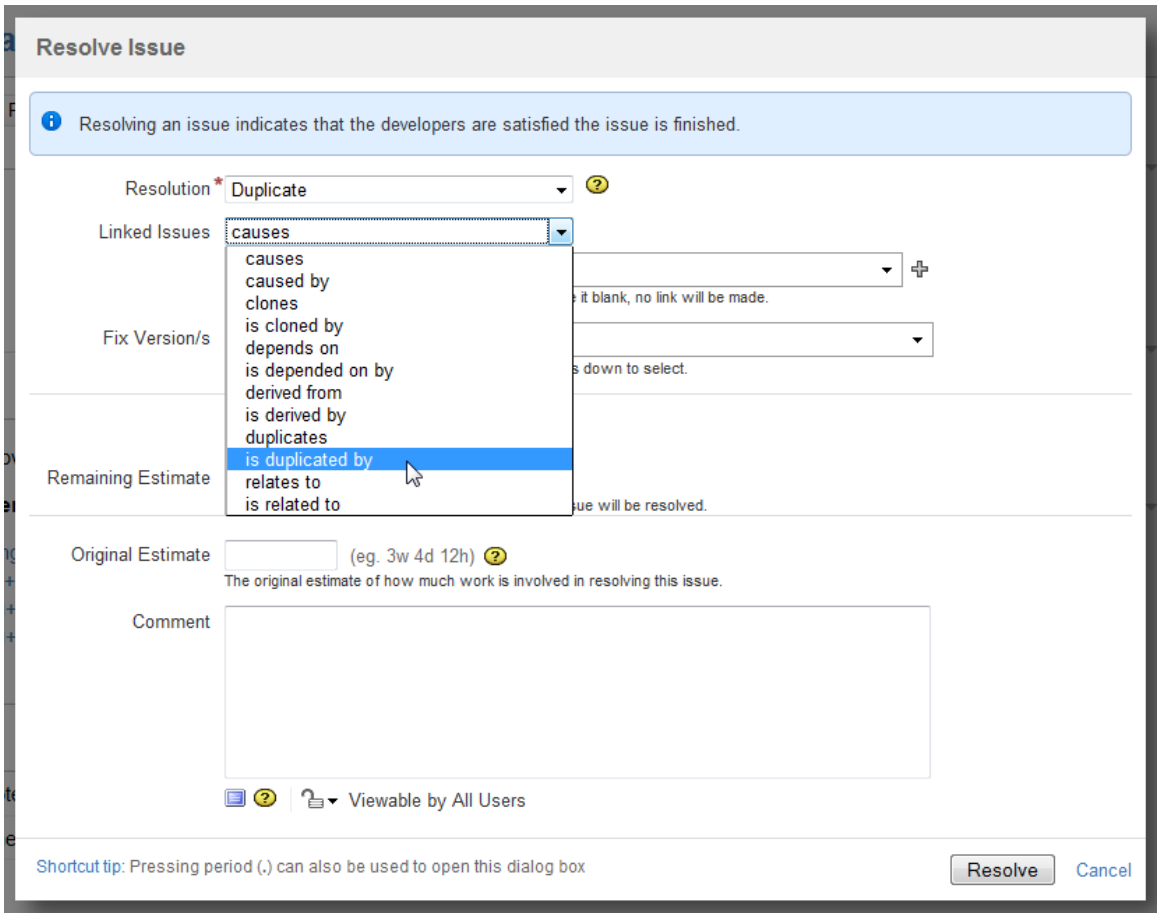
- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

^Top

Issue Linking when Resolving an Issue *(new in EAP 5)*

When resolving an issue, you can create links to other issues on an issue resolution screen. This is handy when you want to resolve an issue as a 'duplicate' of another *and at the same time* link to the duplicate issue.

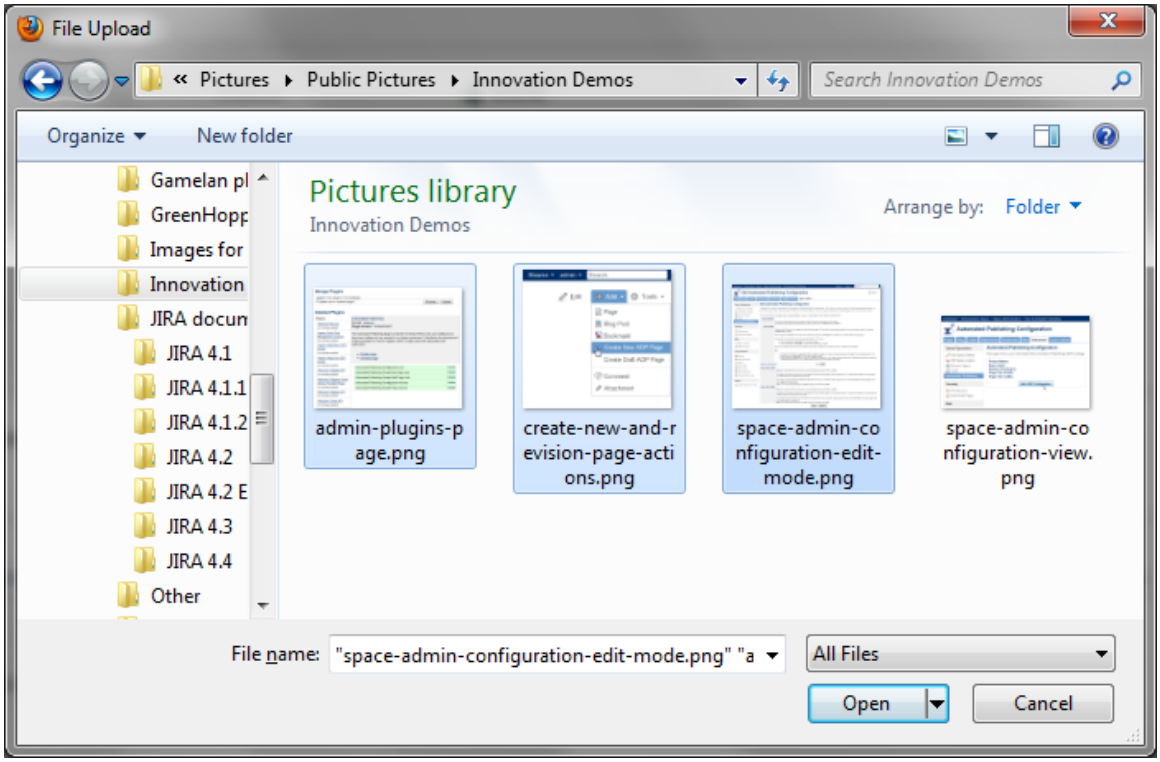


If you are upgrading from an earlier version of JIRA, you need to configure this feature manually through the JIRA administration console. Please refer to the Upgrade Notes for details.

^Top

Multiple File Selection and Upload from the 'File Upload' Dialog Box *(new in EAP 5)*

When using JIRA's 'Attach Files' dialog box, you can now select multiple files in the 'File Upload' dialog box that appears after clicking the 'Browse' button.



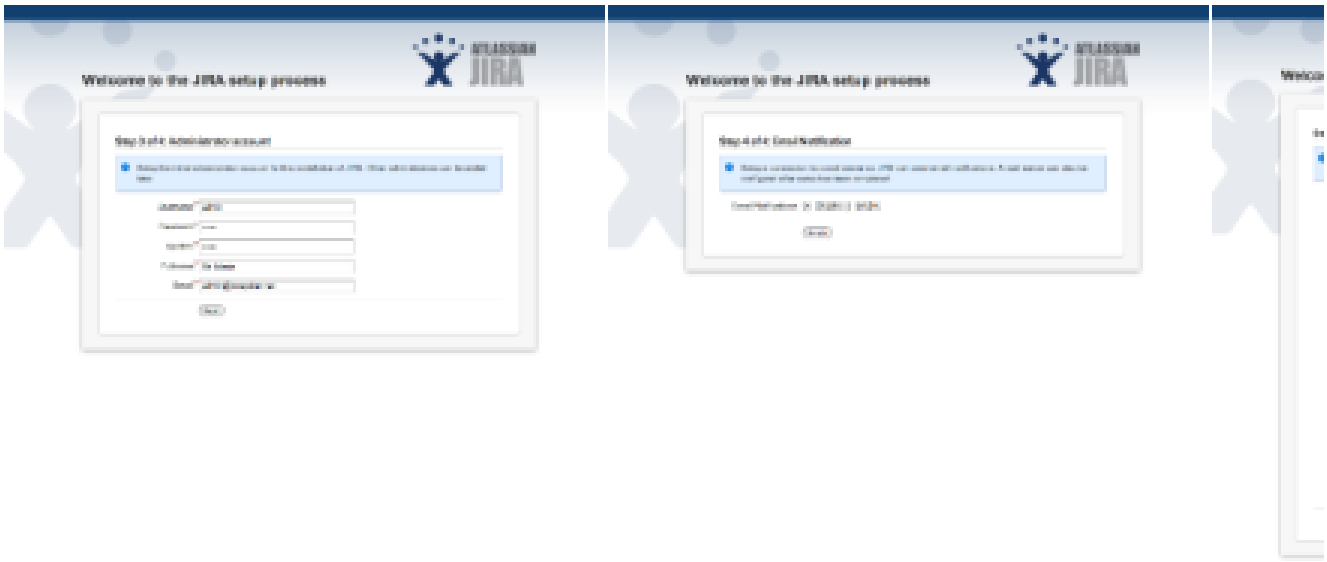
^Top

7

Improved Setup Wizard

Our trusty wizard has had a makeover:

The new Setup Wizard



^Top

Database Configuration Now Part of the Setup Wizard

In JIRA 4.4, configuring a connection to an external database is now part of the standard setup wizard. Customers can now connect JIRA to their database directly via JDBC, rather than having to rely on a JNDI-based connection to a datasource specified in the application server.

The database configuration step of the setup wizard will change before the final 4.4 release, but here is how it looks so far:

^Top

Improved Linux Installers and Uninstaller

Console Installer

A simple command line/console installer wizard is now available for Linux operating systems.

To install JIRA, simply download the Linux `.bin` installer file and at a shell prompt, execute this file and follow the remaining prompts!

You can install JIRA as either the 'root' user or a non-root user under Linux. Currently, however, JIRA will be *run as the same Linux user account* which is used to install JIRA via this wizard. We intend to change this behaviour in a future EAP.

If you use the console installer wizard to install JIRA as the `root` user on your Linux operating system, then for security reasons, you should not use this JIRA installation in a public-facing environment.

Silent Installation

After installing JIRA on Linux using the `.bin` installer file above, a configuration file called `response.varfile` is generated in the `.install4j` subdirectory of your `<JIRA Installation Directory>`. This file contains all the settings you specified manually while installing JIRA at the shell prompt, such as the Tomcat port numbers for JIRA and the location of your JIRA Installation and Home directories

You can use this `response.varfile` file to re-install JIRA again 'silently' (without the need for any user input) by doing the following:

1. Before deleting your original JIRA installation, copy (`cp`) the `<JIRA Installation Directory>/.install4j/response.varfile` to the same directory containing the Linux `.bin` installer file.
2. Enter the following command at the shell prompt:
   ```
   ./atlassian-jira-4.4-m3.bin -q -varfile response.varfile
   ```

Uninstaller

After installing JIRA on Linux using the `.bin` installer file above, an executable file called `uninstall` (located in `<JIRA Installation Directory>`) is available to conveniently uninstall JIRA.

To uninstall JIRA:

1. Log in as the user who installed JIRA.
2. At the shell prompt, change directory (`cd`) into your `<JIRA Installation Directory>`.
3. Execute the `uninstall` command.
   No additional flags or parameters are required.

Please note:

- The uninstaller will not delete the JIRA Home Directory.
- All log files that were generated while JIRA was running will not be deleted.
- All files within the JIRA Installation Directory will be deleted (with the exception of the Tomcat log folder located in `<JIRA Installation Directory>/logs`).

^Top

Improved Windows Installers

The installer wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

Silent Installation

When you execute the installer file at a shell prompt and complete your installation of JIRA, a file called `response.varfile` is generated ir the `.install4j` subdirectory of your `<JIRA Installation Directory>`.

This file contains all the settings you specified manually while installing JIRA at the command prompt, such as the Tomcat port numbers for JIRA and the location of your JIRA Installation and Home directories. You can use `response.varfile` to re-install JIRA again 'silently'

(without the need for any user input) by doing the following:

1. Before deleting your original JIRA installation, `copy` the `<JIRA Installation Directory>\.install4j\response.varfile` to the same directory containing the Windows `.exe` installer file.
2. Open a command prompt and enter the following command:
   `atlassian-jira-4.4-m2.exe -q -varfile response.varfile`

^Top

REST API improvements

There are new REST APIs for

- Listing and managing Project Components.
- Listing and managing Project Versions.
- Listing and managing Project Roles.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.4 EAP 4 Release Notes

2 May 2011

JIRA 4.4 EAP 4 (a.k.a 4.4 milestone 4 or 'm4') is a public development release leading up to **JIRA 4.4**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 4.4 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer. We are also very pleased to announce that the JIRA installation process has been improved and simplified.

Note to developers: In JIRA 4.4 EAP 4 all plugin points are now reloadable — yes, all 51 of them! Also please see the Plugin Developer Notes for JIRA 4.4, plus note that JIRA 4.4 includes the Atlassian Plugin Framework version 2.8.

Highlights of JIRA 4.4 EAP 4:

- Visual Workflow Designer For Administrators
- Workflow Viewer on the 'View Issue' Screen
- User Time Zones ***(improved since EAP 3)***
- Editable Options for Custom Fields
- Improved Setup Wizard
- Database Configuration Now Part of the Setup Wizard
- Improved Linux Installers and Uninstaller
- Improved Windows Installers
- REST API improvements ***(improved since EAP 3)***
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 EAP 4

Download EAP

Upgrading to JIRA 4.4 EAP 4

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

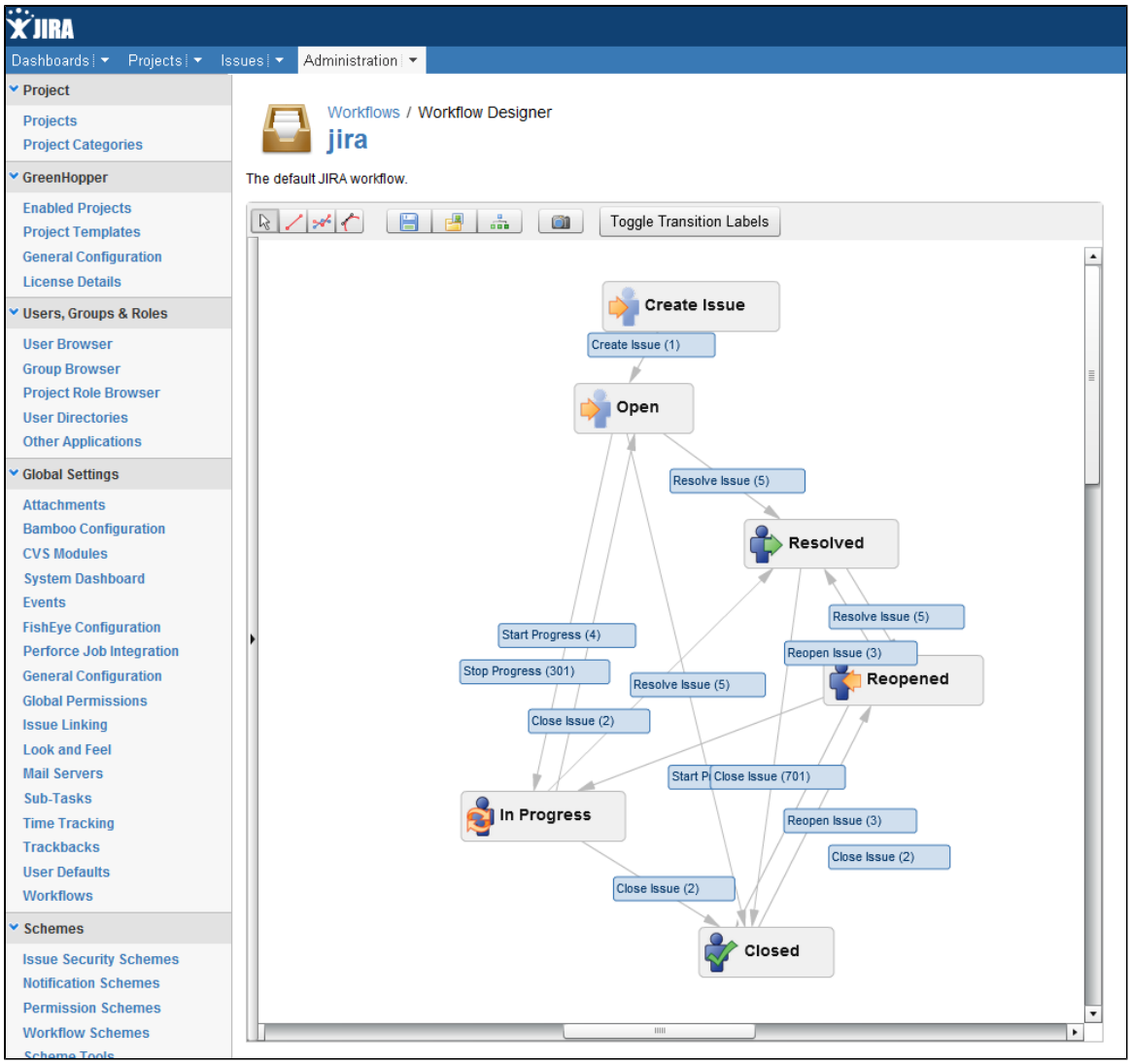
- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.4 EAP 4

Visual Workflow Designer For Administrators

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to '**Workflows**' in JIRA administration as usual, and click the '**Design**' link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.



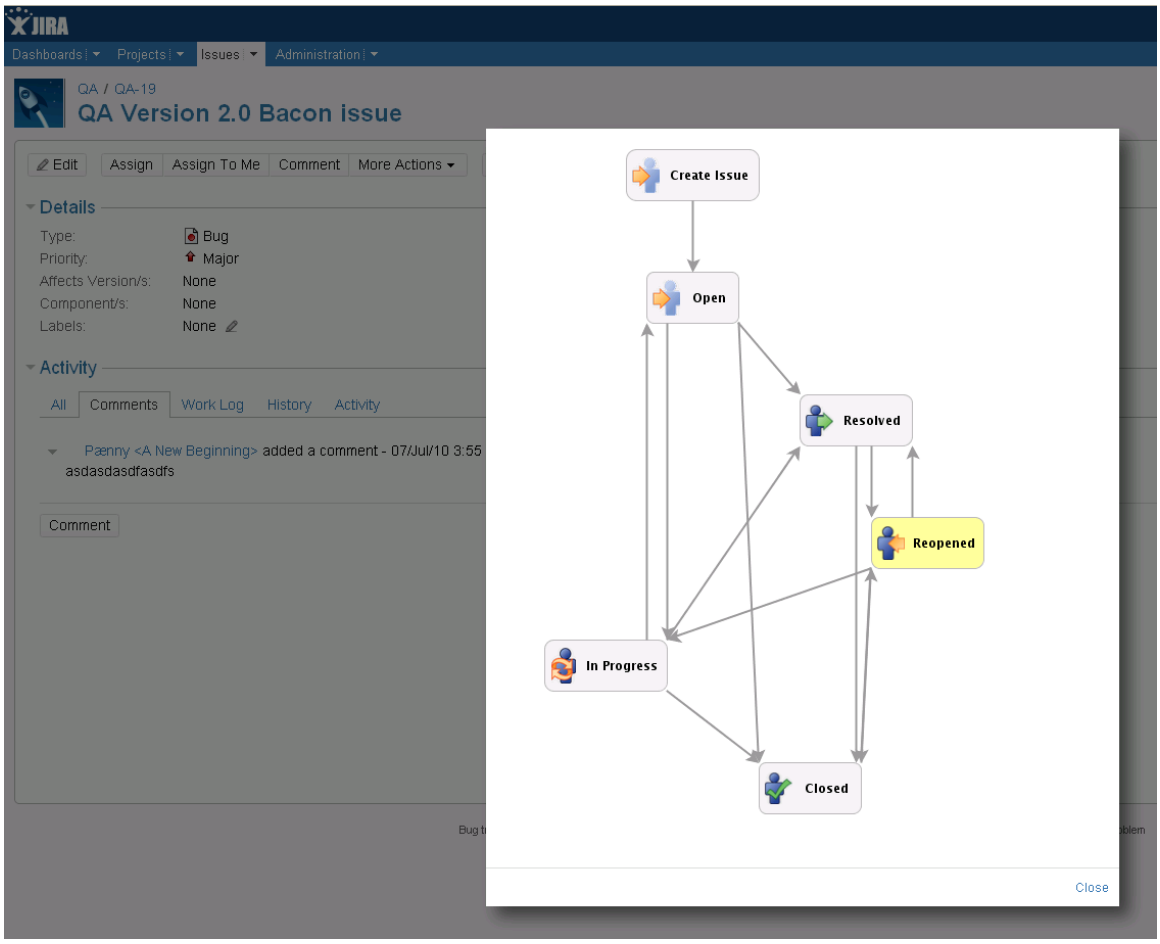
^Top

Workflow Viewer on the 'View Issue' Screen

You can also see a read-only view of the workflow from the 'View Issue' page — just select '**View Workflow**' from the '**More Actions**' menu. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View Read-Only Workflow' is required to access the workflow viewer feature from the 'View Issue' page.



^Top

User Time Zones *(improved since EAP 3)*

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: dates will be displayed to a user in their local time zone, rather than the server's time zone.

You can set a default user time zone at an administration level, and individual users have the ability to choose their own time zone.

In EAP 4, time zone support has been implemented for quick and advanced searching, chart and report gadgets, date/time-based custom fields, as well as issue histories, work logs and source code check-ins (via the JIRA FishEye Plugin). We are still in the process of implementing time zone support for the date/time formats used by all relevant fields and features in JIRA.

In order to see which fields honour an individual's time zone, you must first configure JIRA to use a Date Format that includes the time zone. To do this, go to '**Administration**' => '**Look and Feel**' and edit the '**Date/Time Formats**', adding the letter 'z' or 'Z' in JIRA's Date and Time pattern format. For example, the format `'yyyy.MM.dd G 'at' HH:mm:ss z'` includes the abbreviated time zone as indicated in http://java.sun.com/j2se/1.4.2/docs/api/java/text/SimpleDateFormat.html. You will now see some dates in JIRA display the individual's time zone while other dates still display the default time zone set at the administrative level.

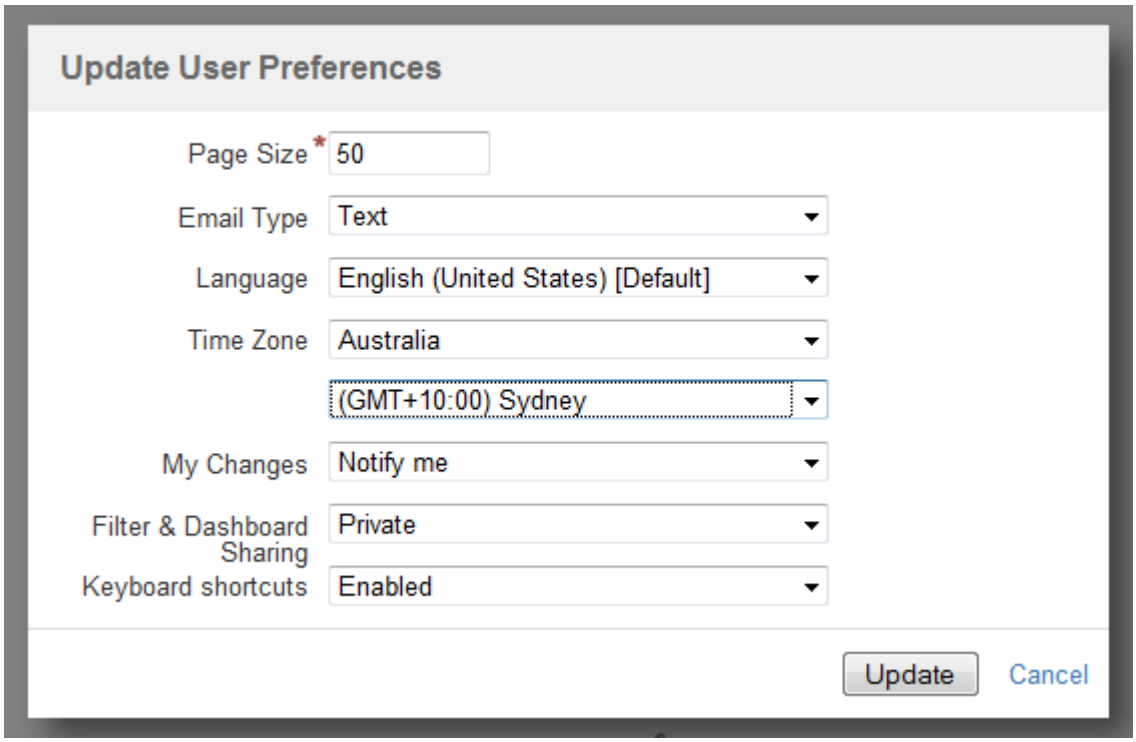
Date/Time Formats

These time and date formats are used throughout JIRA. To specify them, use the format described at http://java.sun.com/j2se/1.4.2/docs/api/java/text/SimpleDateFormat.html.

	Format	Example
Time Format	h:mm a z	2:02 PM EST
Day Format	EEEE h:mm a z	Monday 2:02 PM EST
Complete Date/Time Format	dd/MMM/yy h:mm a z	11/Apr/11 2:02 PM EST
Day/Month/Year Format	dd/MMM/yy z	11/Apr/11 EST
Date Picker Format (javascript format)	d/MMM/yy (%e/%b/%y)	11/Apr/11
Date Time Picker Format (javascript format)	dd/MMM/yy h:mm a (%e/%b/%y %I:%M %p)	11/Apr/11 2:02 PM
Use ISO8601 standard in Date Picker	OFF	

JIRA administrators can change the default time zone by going to '**Administration**' => '**General Configuration**' (under '**Global Settings**'), and editing the '**Default user time zone**'.

Users can also change their individual time zone setting via their user profile:



Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

^Top

Editable Options for Custom Fields

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

^Top

Improved Setup Wizard

Our trusty wizard has had a makeover:

The new Setup Wizard

^Top

6

Database Configuration Now Part of the Setup Wizard

In JIRA 4.4, configuring a connection to an external database is now part of the standard setup wizard. Customers can now connect JIRA to their database directly via JDBC, rather than having to rely on a JNDI-based connection to a datasource specified in the application server.

The database configuration step of the setup wizard will change before the final 4.4 release, but here is how it looks so far:

^Top

Improved Linux Installers and Uninstaller

Console Installer

A simple command line/console installer wizard is now available for Linux operating systems.

To install JIRA, simply download the Linux `.bin` installer file and at a shell prompt, execute this file and follow the remaining prompts!

You can install JIRA as either the 'root' user or a non-root user under Linux. Currently, however, JIRA will be *run as the same Linux user account* which is used to install JIRA via this wizard. We intend to change this behaviour in a future EAP.

If you use the console installer wizard to install JIRA as the `root` user on your Linux operating system, then for security reasons, you should not use this JIRA installation in a public-facing environment.

Silent Installation

After installing JIRA on Linux using the `.bin` installer file above, a configuration file called `response.varfile` is generated in the `.install4j` subdirectory of your `<JIRA Installation Directory>`. This file contains all the settings you specified manually while installing JIRA at the shell prompt, such as the Tomcat port numbers for JIRA and the location of your JIRA Installation and Home directories

You can use this `response.varfile` file to re-install JIRA again 'silently' (without the need for any user input) by doing the following:

1. Before deleting your original JIRA installation, copy (cp) the `<JIRA Installation Directory>/.install4j/response.varfile` to the same directory containing the Linux `.bin` installer file.
2. Enter the following command at the shell prompt:
   ```
   ./atlassian-jira-4.4-m3.bin -q -varfile response.varfile
   ```

Uninstaller

After installing JIRA on Linux using the `.bin` installer file above, an executable file called `uninstall` (located in `<JIRA Installation Directory>`) is available to conveniently uninstall JIRA.

To uninstall JIRA:

1. Log in as the user who installed JIRA.

2. At the shell prompt, change directory (`cd`) into your `<JIRA Installation Directory>`.
3. Execute the `uninstall` command.
   No additional flags or parameters are required.

Please note:

- The uninstaller will not delete the JIRA Home Directory.
- All log files that were generated while JIRA was running will not be deleted.
- All files within the JIRA Installation Directory will be deleted (with the exception of the Tomcat log folder located in `<JIRA Installation Directory>/logs`).

^Top

Improved Windows Installers

The installer wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

Silent Installation

When you execute the installer file at a shell prompt and complete your installation of JIRA, a file called `response.varfile` is generated ir the `.install4j` subdirectory of your `<JIRA Installation Directory>`.

This file contains all the settings you specified manually while installing JIRA at the command prompt, such as the Tomcat port numbers for JIRA and the location of your JIRA Installation and Home directories. You can use `response.varfile` to re-install JIRA again 'silently' (without the need for any user input) by doing the following:

1. Before deleting your original JIRA installation, `copy` the `<JIRA Installation Directory>\.install4j\response.varfile` to the same directory containing the Windows `.exe` installer file.
2. Open a command prompt and enter the following command:
   `atlassian-jira-4.4-m2.exe /q /varfile response.varfile`

^Top

REST API improvements *(improved since EAP 3)*

There are new REST APIs for

- Listing and managing Project Components *(improved since EAP 3)*.
- Listing and managing Project Versions.
- Listing and managing Project Roles *(new in EAP 4)*.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.4 EAP 3 Release Notes

12 April 2011

JIRA 4.4 EAP 3 (a.k.a 4.4 milestone 3 or 'm3') is a public development release leading up to **JIRA 4.4**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

While development work on JIRA 4.4 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer. We are also very pleased to announce that the JIRA installation process has been improved and simplified.

Note to developers: In JIRA 4.4 EAP 3 we have now implemented the majority of the reloadable plugin points — we are aiming to have them all done for EAP 4. Also please see the Plugin Developer Notes for JIRA 4.4.

Highlights of JIRA 4.4 EAP 3:

- Visual Workflow Designer For Administrators
- Workflow Viewer on the "View Issue" Screen
- User Time Zones ***(new in EAP 3)***
- Editable Options for Custom Fields ***(new in EAP 3)***
- Improved Setup Wizard ***(new in EAP 3)***
- Database Connection Now Included In Setup
- Improved Linux Installer + Uninstaller ***(New in EAP 3)***
- Improved Windows Installer
- REST API improvements ***(new in EAP 3)***
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 EAP 3
Download EAP

Upgrading to JIRA 4.4 EAP 3

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

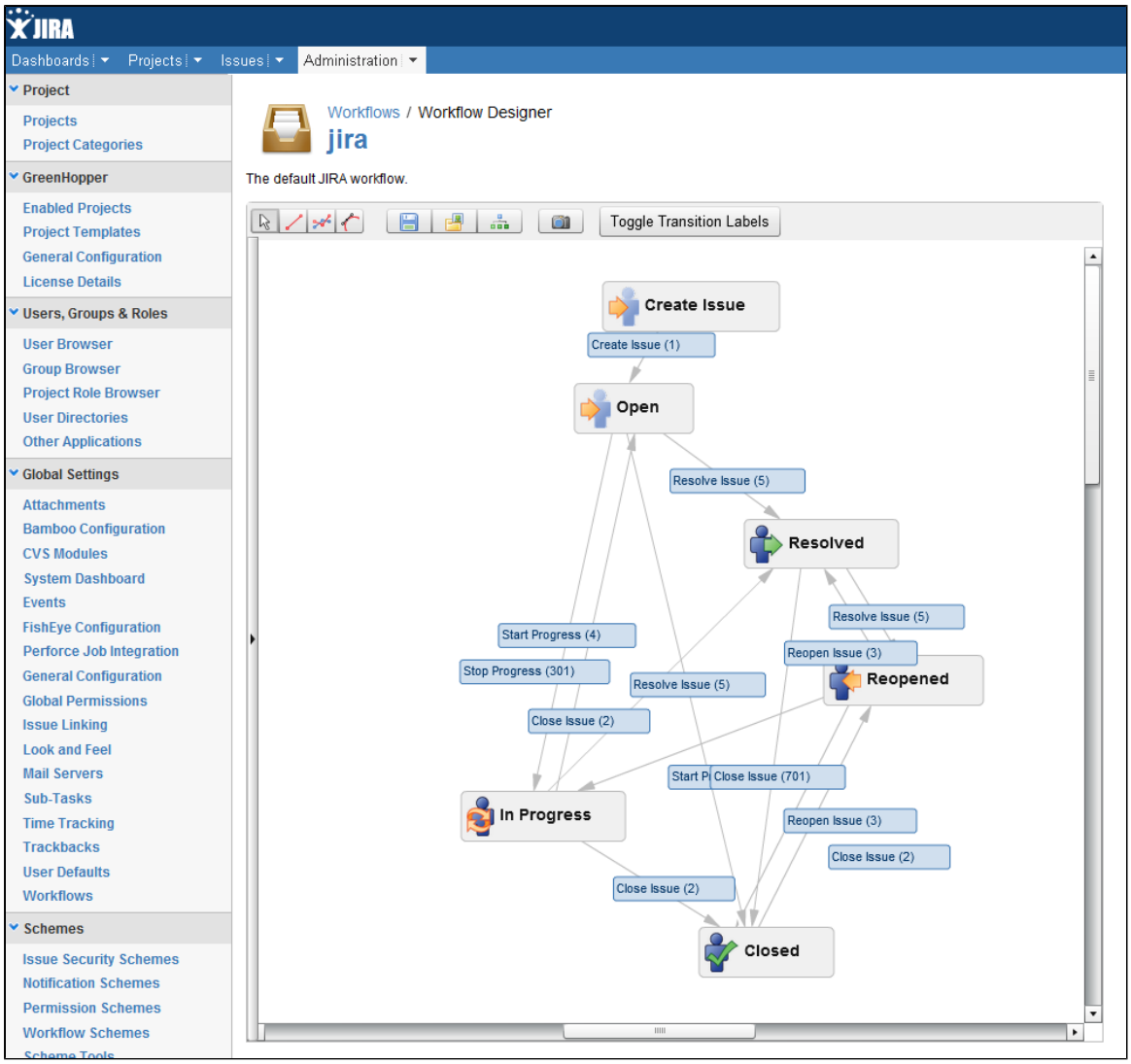
- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.4 EAP 3

Visual Workflow Designer For Administrators

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to "**Workflows**" in JIRA administration as usual, and click the "**Design**" link for the workflow you wish to edit. If you hover your mouse over an individual Step or Transition, a 'cog' icon appears, which you can click to access more functions.



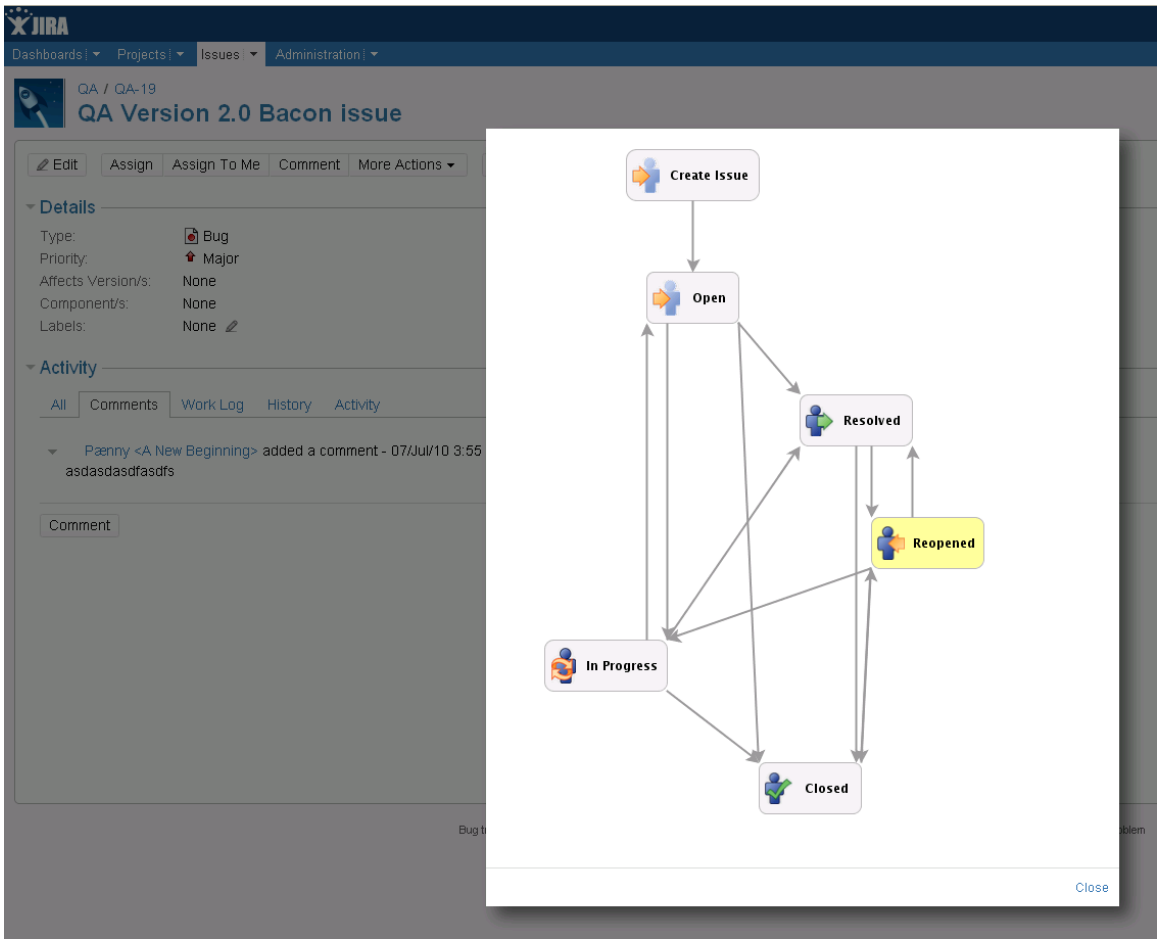
^Top

Workflow Viewer on the "View Issue" Screen

You can also see a read-only view of the workflow from the "**View Issue**" page — just select "**View Workflow**" from the "**More Actions**" menu. We've made it simple for everyone to use, without requiring Flash to be installed.

The workflow viewer also highlights the current status of your issue in the workflow.

A new permission called 'View Read-Only Workflow' is required to access the workflow viewer feature from a view issue page.



^Top

User Time Zones *(new in EAP 3)*

We are very pleased to announce progress on JRA-9, one of the most highly-voted requests for JIRA: dates will be displayed to a user in their local time zone, rather than the server's time zone.

In EAP 3, you have the ability to set a default user time zone at an administration level, and individual users have the ability to choose their own time zone. We are in the process of implementing this for the date/time formats used by all relevant fields and features in JIRA. For example, time zone support has not yet been implemented for searching.

In order to see which fields honour an individual's time zone you must first configure JIRA to use a Date Format that includes the time zone. To do this, go to "**Administration**" => "**Look and Feel**" and edit the "**Date/Time Formats**", adding the letter "z" or "Z" in JIRA's Date and Time pattern format. For example, the format "yyyy.MM.dd G 'at' HH:mm:ss z" includes the abbreviated time zone as indicated in http://java.sun.com/j2se/1.4.2/docs/api/java/text/SimpleDateFormat.html. You will now see some dates in JIRA display the individual's time zone while other dates still display the default time zone set at the administrative level.

Date/Time Formats

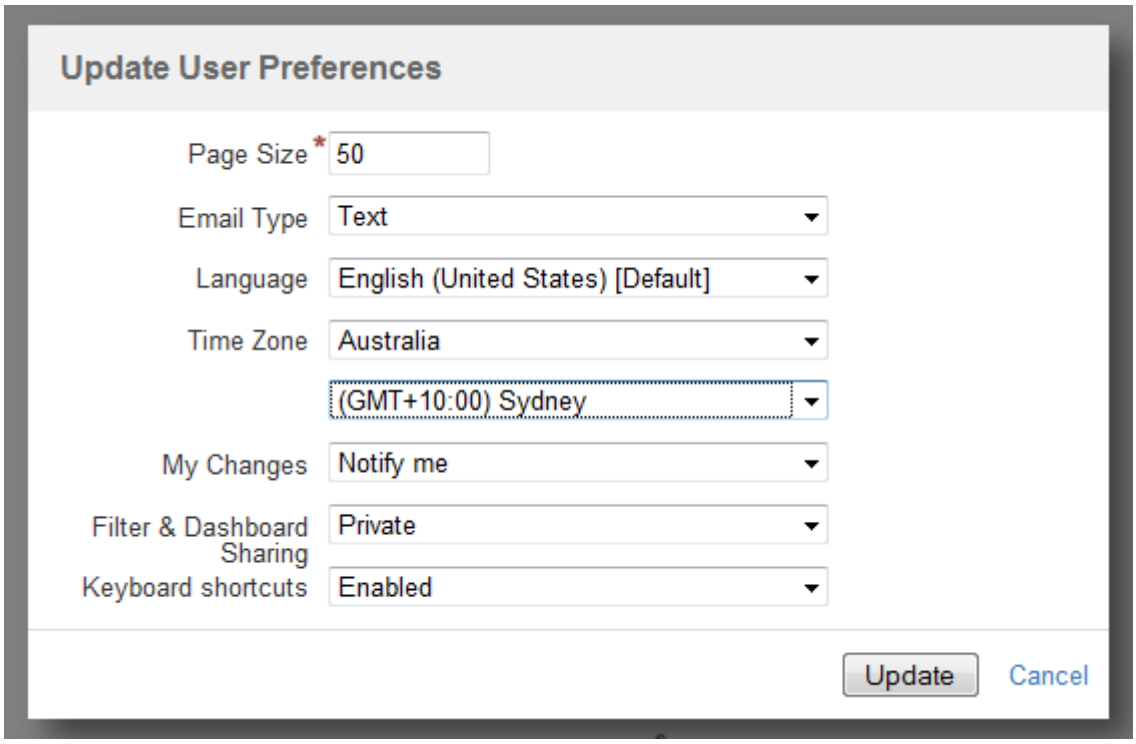
These time and date formats are used throughout JIRA. To specify them, use the format described at http://java.sun.com/j2se/1.4.2/docs/api/java/text/SimpleDateFormat.html.

	Format	Example
Time Format	h:mm a z	2:02 PM EST
Day Format	EEEE h:mm a z	Monday 2:02 PM EST
Complete Date/Time Format	dd/MMM/yy h:mm a z	11/Apr/11 2:02 PM EST
Day/Month/Year Format	dd/MMM/yy z	11/Apr/11 EST
Date Picker Format (javascript format)	d/MMM/yy (%e/%b/%y)	11/Apr/11
Date Time Picker Format (javascript format)	dd/MMM/yy h:mm a (%e/%b/%y %I:%M %p)	11/Apr/11 2:02 PM
Use ISO8601 standard in Date Picker	OFF	

JIRA Administrators can change the default time zone by going to "**Administration**" => "**General Configuration**" (under "Global Settings"),

and editing the "**Default user time zone**".

Users can also change their individual time zone setting via their user profile:



Note to developers: If you develop JIRA plugins that handle dates and times, please be aware of the Formatting and Parsing Dates Using the Appropriate Time Zone section of the Plugin Developer Notes for JIRA 4.4.

^Top

Editable Options for Custom Fields *(new in EAP 3)*

We are also pleased to announce progress on JRA-2983. You can now edit the options for custom fields of the following field types:

- Select List
- Multi Select
- Cascading Select
- Radio Buttons
- Multi Checkboxes

Because your custom fields may change over time, JIRA now gives you the ability to:

- edit a field's options — that is, change the text of an option.
- disable an option — that is, hide an option so that it is no longer available for selection. Options that have been used cannot be removed (to preserve data integrity), but due to changing business requirements, they may become invalid over time and so you may wish to make them unavailable for new issues.

Note to developers: If you develop JIRA plugins that handle single- and multi-select custom fields, please be aware of the Single- and Multi-Select Custom Field Changes section of the Plugin Developer Notes for JIRA 4.4.

^Top

Improved Setup Wizard *(new in EAP 3)*

Our trusty wizard has had a makeover:

The new Setup Wizard

^Top

Database Connection Now Included In Setup

In JIRA 4.4, connecting to a database is now part of the standard setup wizard. It will change before the final 4.4 release, but here is how it looks in M3:

^Top

Improved Linux Installer + Uninstaller *(New in EAP 3)*

A simple command line/console installer wizard is now available for Linux operating systems.

To install JIRA, simply download the Linux `.bin` installer file and at a shell prompt, execute this file and follow the remaining prompts!

You can install JIRA as either the 'root' user or a non-root user under Linux. Currently, however, JIRA will be *run as the same Linux user account* which is used to install JIRA via this wizard. We intend to change this behaviour in a future EAP.

If you use the console installer wizard to install JIRA as the `root` user on your Linux operating system, then for security reasons, you should not use this JIRA installation in a public-facing environment.

Uninstaller for Linux *(New in EAP 3)*

After installing JIRA on Linux using the `.bin` installer file above, an executable file called `uninstall` (located in `<JIRA Installation Directory>`) is available to conveniently uninstall JIRA.

To uninstall JIRA:

1. Log in as the user who installed JIRA.
2. At the shell prompt, change directory (`cd`) into your `<JIRA Installation Directory>`.
3. Execute the `uninstall` command.
   No additional flags or parameters are required.

Please note:

- The uninstaller will not delete the JIRA Home Directory.
- All log files that were generated while JIRA was running will not be deleted.
- All files within the JIRA Installation Directory will be deleted (with the exception of the Tomcat log folder located in `<JIRA Installation Directory>/logs`).

Silent Installations *(New in EAP 3)*

After installing JIRA on Linux using the `.bin` installer file above, a configuration file called `response.varfile` is generated in the `.install4j` subdirectory of your `<JIRA Installation Directory>`. This file contains all the settings you specified manually while

installing JIRA at the shell prompt, such as the Tomcat port numbers for JIRA and the location of your JIRA Installation and Home directories

You can use this `response.varfile` file to re-install JIRA again 'silently' (without the need for any user input) by doing the following:

1. Before deleting your original JIRA installation, copy (`cp`) the `<JIRA Installation Directory>/.install4j/response.varfile` to the same directory containing the Linux `.bin` installer file.
2. Enter the following command at the shell prompt:
   `./atlassian-jira-4.4-m3.bin -q -varfile response.varfile`

^Top

Improved Windows Installer

The installer wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and the wizard now allows you to specify the port numbers used to run and access JIRA.

Silent Installations *(New in EAP 3)*

When you execute the installer file at a shell prompt and complete your installation of JIRA, a file called `response.varfile` is generated ir the `.install4j` subdirectory of your `<JIRA Installation Directory>`.

This file contains all the settings you specified manually while installing JIRA at the command prompt, such as the Tomcat port numbers for JIRA and the location of your JIRA Installation and Home directories. You can use `response.varfile` to re-install JIRA again 'silently' (without the need for any user input) by doing the following:

1. Before deleting your original JIRA installation, `copy` the `<JIRA Installation Directory>\.install4j\response.varfile` to the same directory containing the Windows `.exe` installer file.
2. Open a command prompt and enter the following command:
   `atlassian-jira-4.4-m2.exe /q /varfile response.varfile`

^Top

REST API improvements *(new in EAP 3)*

There are new REST APIs for

- Listing and managing Project Components.
- Listing and managing Project Versions.

Have a look at the reference documentation.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.4 EAP 2 Release Notes

31 March 2011

JIRA 4.4 EAP 2 (a.k.a 4.4 milestone 2 or 'm2') is a public development release leading up to **JIRA 4.4**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

Please note that JIRA 4.4 EAP 1 was not released to the public.

While development work on JIRA 4.4 commenced relatively recently, we want your involvement from the earliest days. Please provide feedback here.

JIRA 4.4 brings you a visual Workflow Designer and Workflow Viewer. We are also very pleased to announce that the JIRA installation process has been improved and simplified.

Highlights of JIRA 4.4 EAP 2:

- Visual Workflow Designer For Administrators
- Workflow Viewer on the "View Issue" Screen
- Database Connection Now Included In Setup
- Console Installer Wizard for Linux
- Improvements to the Windows Installer Wizard
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.4 EAP 2
Download EAP

Upgrading to JIRA 4.4 EAP 2

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.4 Upgrade Notes.

Do not use in production

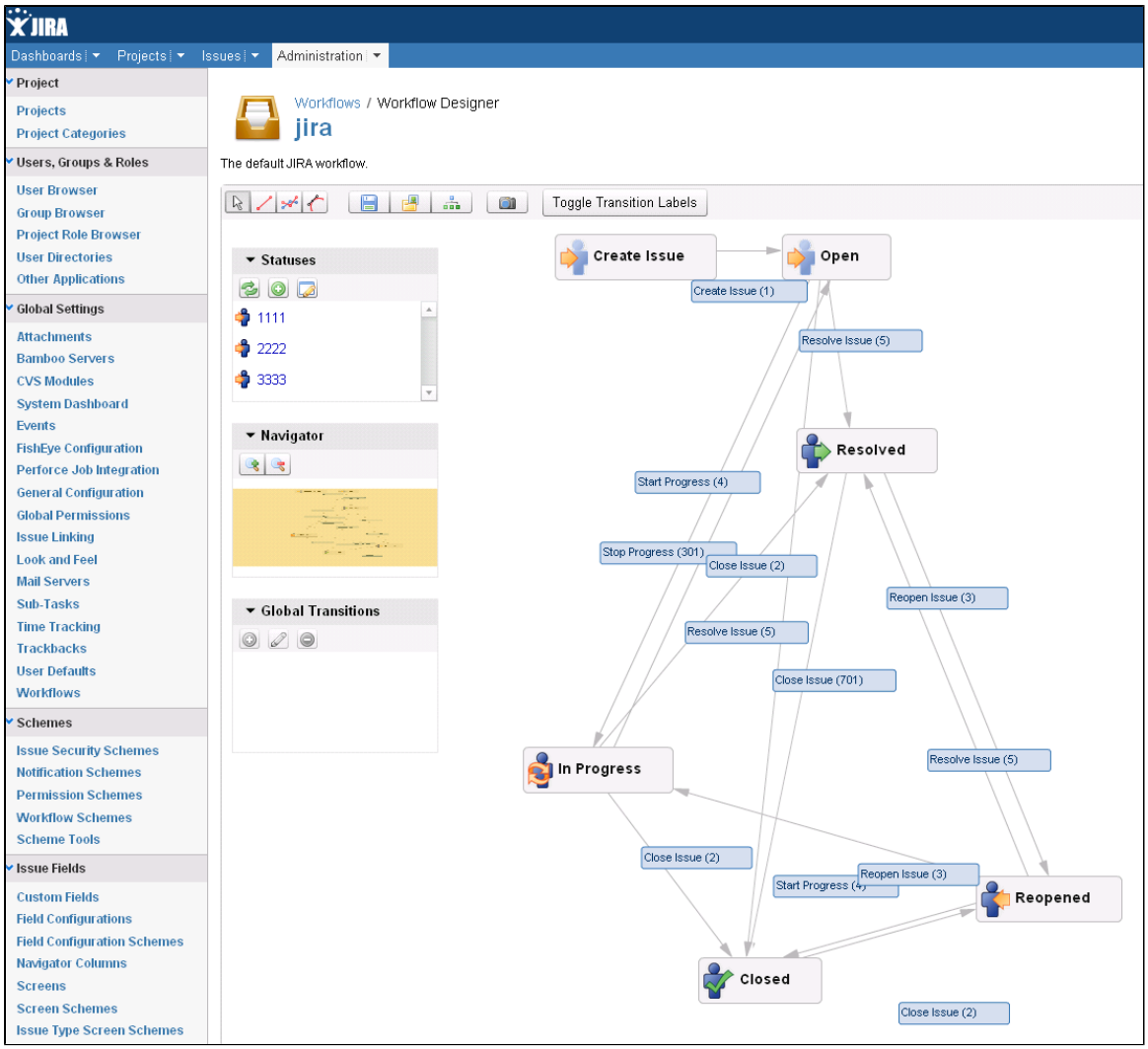
- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.4 EAP 2

Visual Workflow Designer For Administrators

JIRA 4.4 provides a visual representation of your workflow, making it easy to map out your business processes.

Simply go to "**Workflows**" in JIRA administration as usual, and click the "**Design**" link for the workflow you wish to edit. You can right-click or individual Steps and Transitions to view more information.

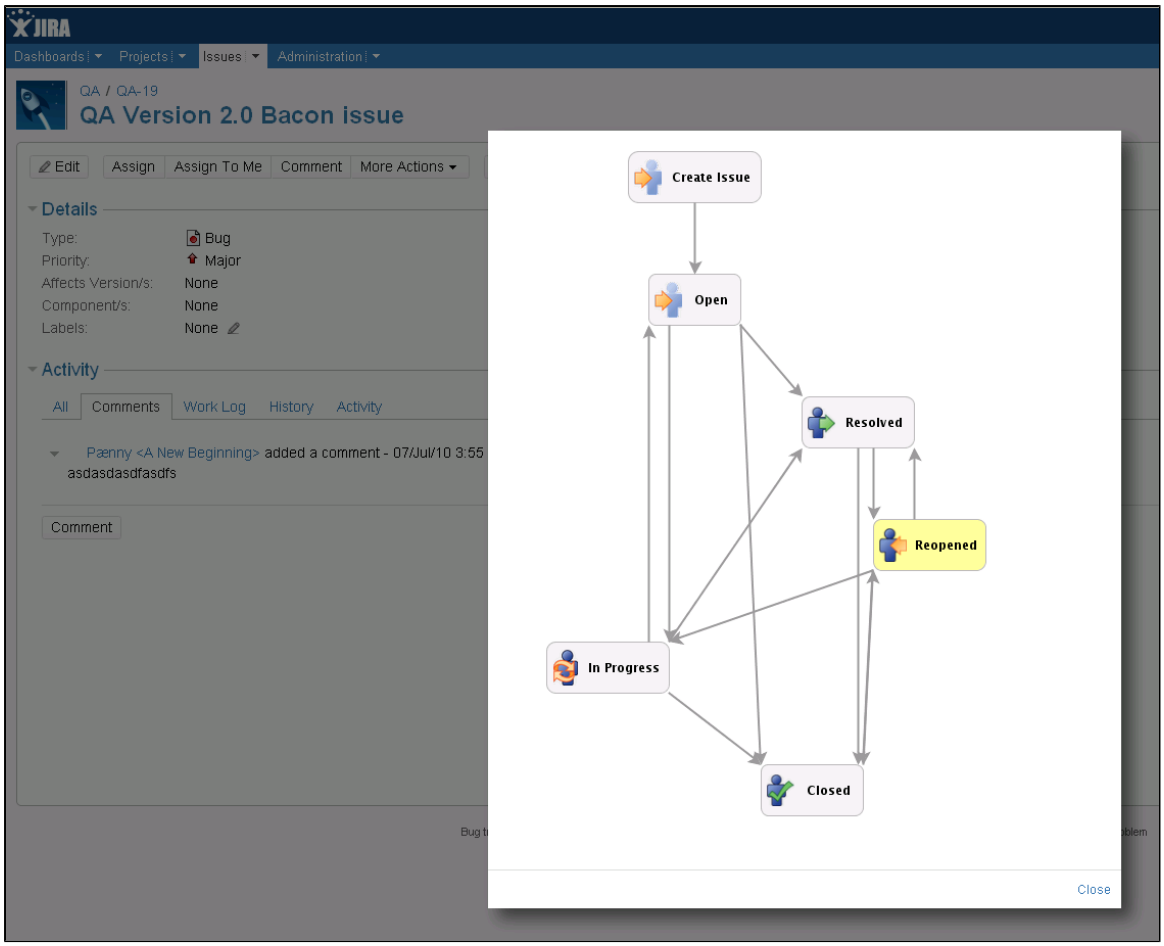


^Top

Workflow Viewer on the "View Issue" Screen

You can also see a read-only view of the workflow from the "**View Issue**" page — just select "**View Workflow**" from the "**More Actions**" menu. We've made it simple for everyone to use, without requiring Flash to be installed.

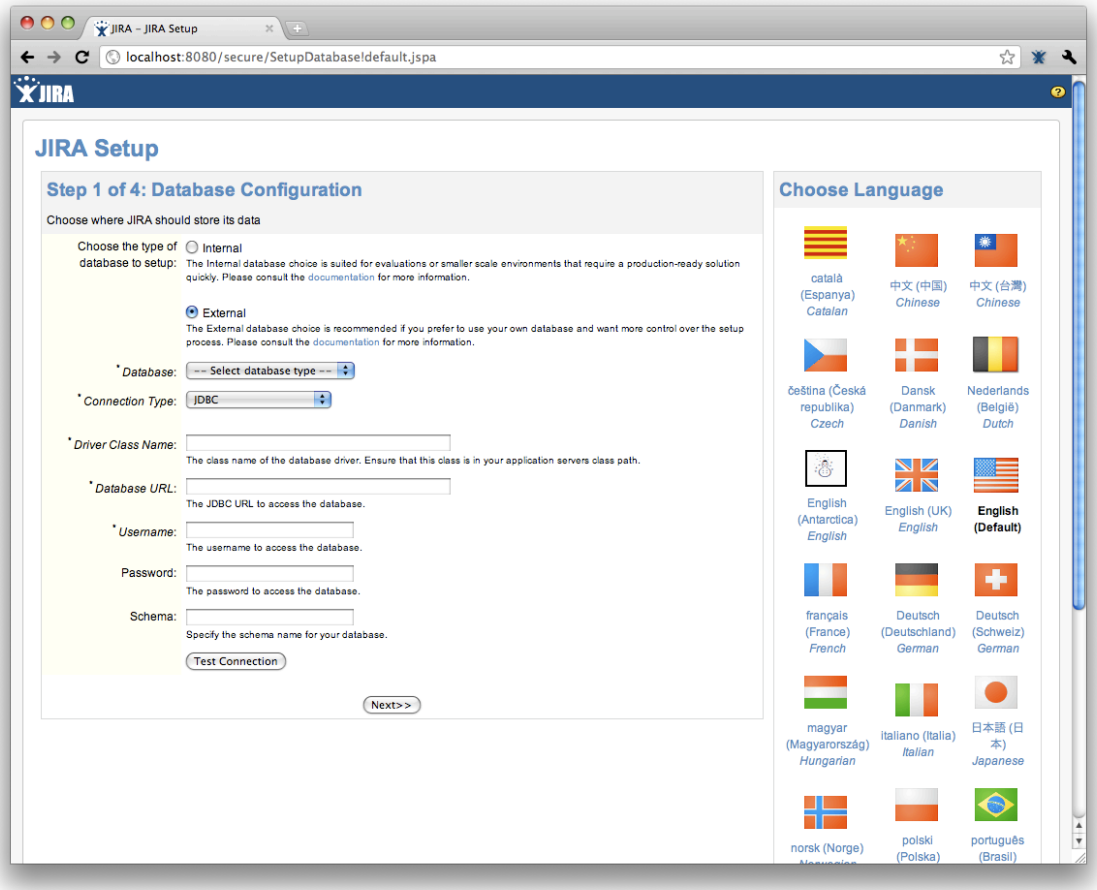
The workflow viewer also highlights the current status of your issue in the workflow.



^Top

Database Connection Now Included In Setup

In JIRA 4.4, connecting to an external database is now part of the standard setup wizard. It will change before the final 4.4 release, but here is how it looks in M2:



^Top

Console Installer Wizard for Linux

A simple command line/console installer wizard is now available for Linux operating systems.

To install JIRA, simply download the Linux `.bin` installer file and at the command line or shell prompt, execute this file and follow the remaining prompts!

You can install JIRA as either the 'root' user or a non-root user under Linux. Currently, however, JIRA will be *run as the same Linux user account* which is used to install JIRA via this wizard. We intend to change this behaviour in a future EAP.

If you use the console installer wizard to install JIRA as the `root` user on your Linux operating system, then for security reasons, you should not use this JIRA installation in a public-facing environment.

^Top

Improvements to the Windows Installer Wizard

The installer wizard for Windows now works smoothly for Windows 7 and Vista.

The process for installing JIRA as a Windows service has also improved and wizard now allows you to specify the port numbers used to run and access JIRA.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.4 so far, click here.

^Top

JIRA 4.3 EAP 5 Release Notes

14 January 2011

JIRA 4.3 EAP 5 (a.k.a 4.3 milestone 5 or 'm5') is a public development release leading up to **JIRA 4.3**. An **Early Access Preview (EAP) release** is a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

- A note for the curious: we skipped the EAP 4 release due to the holiday season.

The Atlassian team is proud to bring you the JIRA 4.3 EAP 5 release. Thank you for your feedback so far, and please keep providing it here.

Identity management comes of age in JIRA 4.3, with complete LDAP integration. Additionally, you now have the ability to search an issue's change history. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins.

Dashboard Publish/Subscribe with Confluence (i.e. Gadgets 2.0) is also included, so if you are using Confluence as well as JIRA, your Confluence gadgets will now appear in the JIRA Gadget Directory (and vice versa).

Note to developers: JIRA 4.3 EAP 5 includes Unified Application Links (UAL), Gadgets 2.0, and the Atlassian Plugin Framework version 2.6 . Also please see the Plugin Developer Notes for JIRA 4.3, plus note that the REST API will continue to evolve through the 4.3 EAP releases — please watch the Developer blog for the latest news on the REST API.

Highlights of JIRA 4.3 EAP 5:

- Add another application's Gadgets to your JIRA Gadget Directory
- Search the Change History with JQL 'WAS' Function ***(improved since EAP 3)***
- Full LDAP integration ***(upgrade note added since EAP 3)***
- User Directory management via JIRA Admin UI
- Revamped User Avatars ***(new since EAP 3)***
- Mail Server Configuration Improvements ***(improved since EAP 3)***
- User Email Change is now Password Protected ***(new since EAP 3)***
- New Plugin Management System
- AppLinks pre-installed and configurable from the JIRA Admin UI
- Enhancements to Quick Search ***(improved since EAP 3)***
- Faster Dashboards
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.3 EAP 5
Download EAP

Upgrading to JIRA 4.3 EAP 5

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.3 Upgrade Guide.

Do not use in production

- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.3 EAP 5

Add another application's Gadgets to your JIRA Gadget Directory

JIRA 4.3 EAP 2 includes Atlassian Gadgets 2.0, allowing you to quickly add all gadgets from your Confluence, Bamboo, FishEye or Crucible instance — or from another JIRA instance — to your JIRA Gadget Directory, for easy addition to your JIRA dashboard:

In the JIRA Gadget Directory , you can now click 'Gadget Subscriptions'. There you can provide the URL for the other application (or other JIRA instance), and all the gadgets from that instance will be added to your JIRA Gadget Directory.

^Top

Search the Change History with JQL 'WAS' Function *(improved since EAP 3)*

In EAP 2 we released the first cut of Change History searching. There is much more functionality to follow, but for now you can search for changes to the Status field.

For example, the following will return all issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

Autocomplete has been implemented in EAP 5.

^Top

Full LDAP integration *(upgrade note added since EAP 3)*

The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3. This provides a number of additional capabilities, mainly the ability to use an LDAP server for all user information. Key features are:

- Ability to connect to an LDAP server, including to Microsoft Active Directory, for user management. This includes:
  - Read-write access
  - Read-only access
  - Read-only access with local groups
  - Use LDAP for authentication only — *Previous functionality*
- Ability to connect to a Crowd server for user management — *Previous functionality*
- Ability to manage users fully within JIRA — *Previous functionality*
- Ability to use 2 or more LDAP and/ or Crowd servers simultaneously for user management.

Note when upgrading from EAP 3

If you are upgrading to EAP 5 from EAP 3, please note that there have been changes to Crowd that require extra data in the cwd_applicatior table.

You will need to update the table and add 'CROWD' into the application_type column:

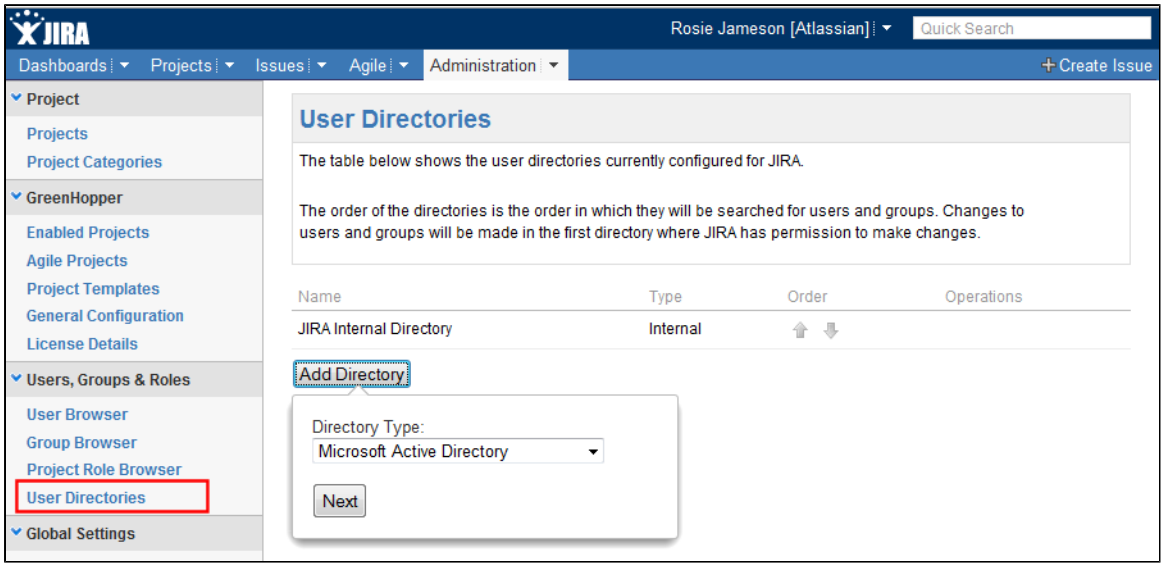
```
UPDATE cwd_application SET application_type = 'CROWD';
```

(This will not be necessary when upgrading to the final release of JIRA 4.3.)

^Top

User Directory management via JIRA Admin UI

In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.



^Top

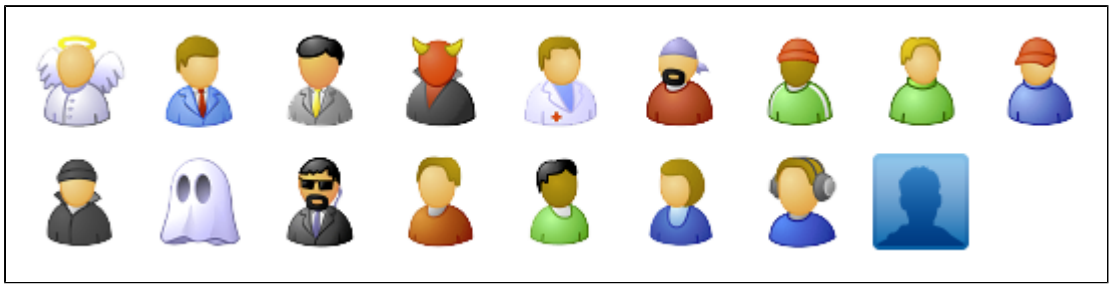
Revamped User Avatars *(new since EAP 3)*

JIRA 4.3 introduces the new-look Atlassian avatars:

- User avatars are displayed as the icon for your profile, and to illustrate your comments on an issue. See the documentation on

Adding a User Avatar.

We were fond of the old avatars, but think you'll agree they were looking a little dated by comparison:

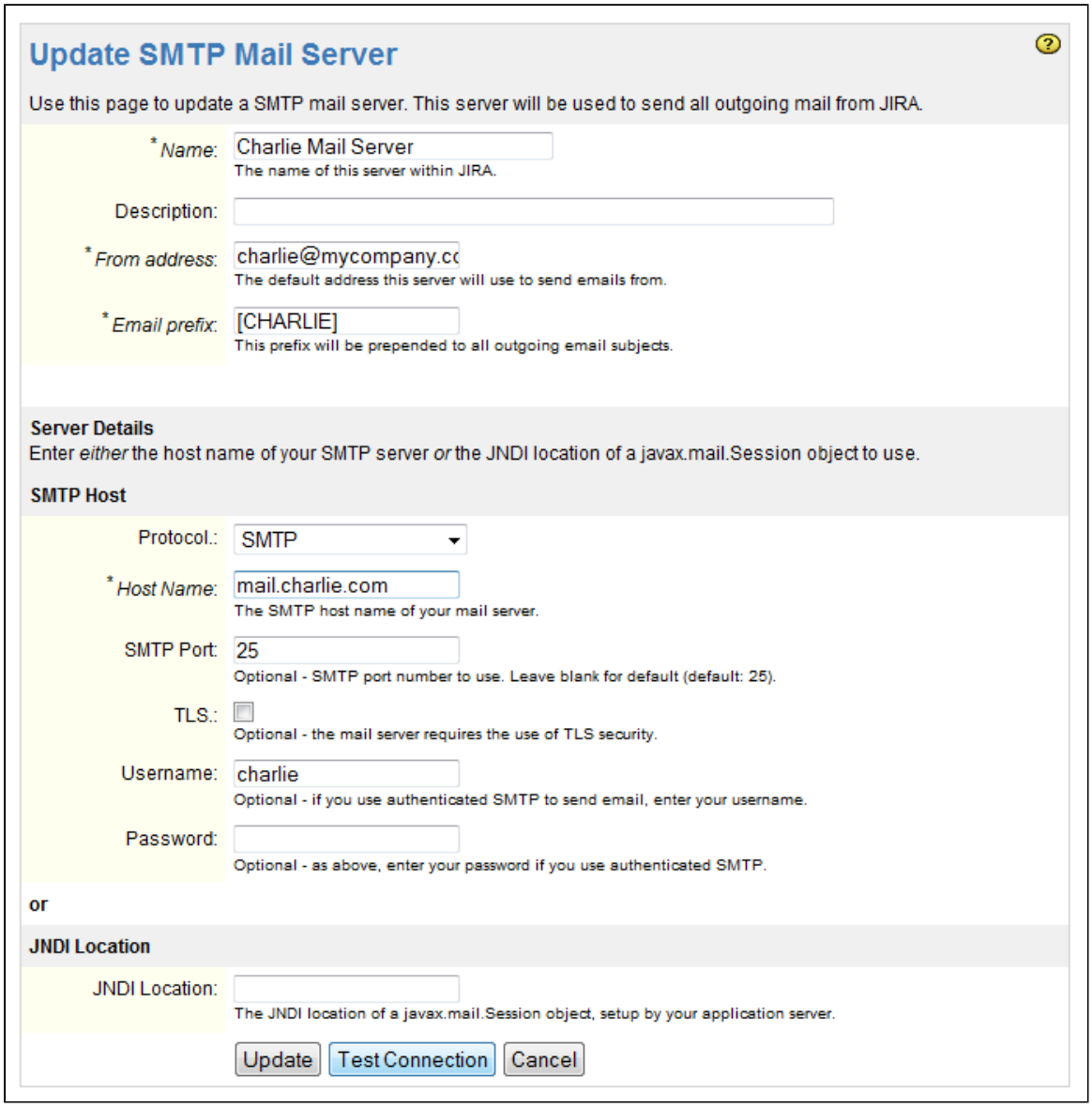


^Top

6

Mail Server Configuration Improvements *(improved since EAP 3)*

In JIRA 4.3, administrators can now test their mail server configuration with the "Test Connection" button.



A '**Timeout**' field has also been added in EAP 5.

^Top

7

User Email Change is now Password Protected *(new since EAP 3)*

For enhanced security, users will now be prompted to enter their password when changing their email address.

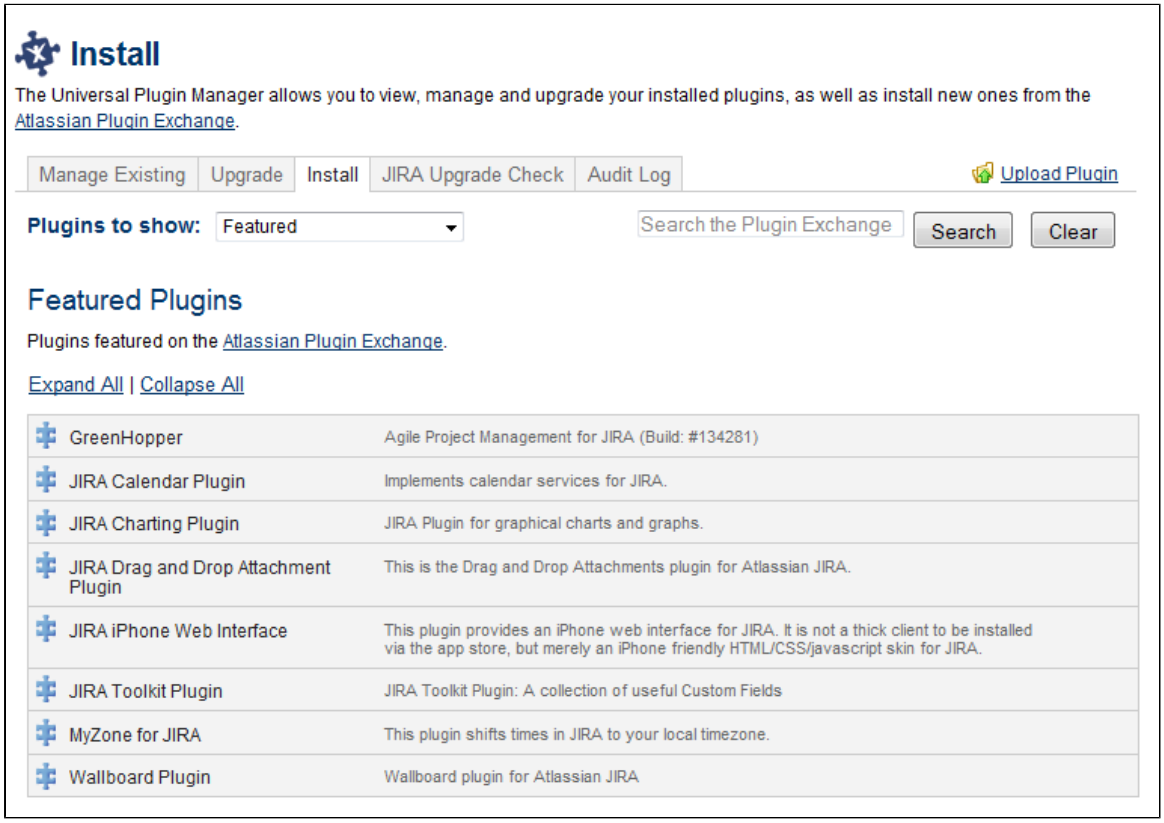
^Top

8

New Plugin Management System

JIRA 4.3 includes the Universal Plugin Manager (UPM), which provides you with a simple way of adding and managing plugins:

- Auto-discovery of available plugins (see **Featured Plugins** in the screenshot below)
- Point-and-click installation — no more downloading JAR files



^Top

9

AppLinks pre-installed and configurable from the JIRA Admin UI

JIRA 4.3 EAP 5 includes Unified Application Links (UAL), so you no longer need to install it separately.

Application Links (AppLinks) is a plugin that allows you to link your JIRA, Confluence, FishEye, Crucible, Bamboo and Subversion applications. You can even choose to associate individual entities (i.e. JIRA projects, Confluence spaces, FishEye repositories, FishEye projects, Crucible projects, Bamboo projects) with each other. Applications Links is bundled with FishEye 2.4, Confluence 3.5, JIRA 4.3, and all later versions of those applications. In addition, Bamboo 3.1 is compatible with AppLinks. You can configure JIRA-to-Bamboo links via the JIRA administration screens.

Linking two applications allows you to share information and access one application's functions from within the other. For example, if you linked a JIRA server and a Confluence server, you could view JIRA issues in a Confluence page via the JIRA Issues Macro.

You also can associate entities of two linked applications. For example, you could associate a JIRA project with a Confluence space. This allows you to take advantage of additional integration features like link rendering, which lets you create links to issues or pages in the project or space using a simple textual reference, e.g. `[JRA-1234], [myConfluenceSpace:Test Page]`.

- In JIRA 4.3 EAP 5 there is now a '**Managed Unified Application Links**' option on the '**Manage Project**' screen, enabling you to easily configure application links for a given project.

For more about configuring Application Links, please see the Application Links Administrator's Guide.

^Top

Enhancements to Quick Search *(improved since EAP 3)*

When using Quick Search to find issues with a particular version, you can now use the wildcard symbol: "`*`". For example, "`ff:3.2*`" will match any issue whose Fix For Version is:

- 3.2
- 3.2-beta
- 3.2.1
- 3.2.x

You can also find issues reported by you, another user or with no reporter, using the prefix "`r:`" followed by a specific reporter term such as '`me`", a username or "`none`", such that:

- "`r:me`" — finds issues reported by you.
- "`r:samuel`" — finds issues reported by the user whose username is "samuel".
- "`r:none`" — finds issues with no reporter.

^Top

Faster Dashboards

We are extremely pleased to announce that you should see a noticeable reduction in the time it takes to load a Dashboard.

- On average, dashboard performance has improved by 29%
- For large dashboards, the performance improvement is 35%

^Top

REST API improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance. Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3 so far, click here.

^Top

JIRA 4.3 EAP 3 Release Notes

20 December 2010

JIRA 4.3 EAP 3 (a.k.a 4.3 milestone 3 or 'm3') is a public development release leading up to **JIRA 4.3**. An **Early Access Preview (EAP) release** is a a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 4.3 EAP 3 release. Development work on JIRA 4.3 has only commenced very recently, but we want your involvement from the earliest days. Please provide feedback here.

Identity management comes of age in JIRA 4.3, with complete LDAP integration. Additionally, you now have the ability to search an issue's change history. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins.

Dashboard Publish/Subscribe (i.e. Gadgets 2.0) is also included, so you can now add gadgets from another Atlassian application (Confluence, FishEye, Crucible, Bamboo) — or from another JIRA instance — to your JIRA Gadget Directory.

Note to developers: JIRA 4.3 EAP 3 includes Unified Application Links (UAL) *(new since EAP 2)*, Gadgets 2.0 *(new since EAP 2)*, and the Atlassian Plugin Framework version 2.6. Also please see the Plugin Developer Notes for JIRA 4.3, plus note that the REST API will continue to evolve through the 4.3 EAP releases — please watch the Developer blog for the latest news on the REST API.

Highlights of JIRA 4.3 EAP 3:

- Add another application's Gadgets to your JIRA Gadget Directory ***(new since EAP 2)***
- Search the Change History with JQL 'WAS' Function
- Full LDAP integration
- User Directory management via JIRA Admin UI
- Mail Server Configuration Improvements
- New Plugin Management System
- AppLinks pre-installed and configurable from the JIRA Admin UI ***(new since EAP 2)***
- Quick Search now has a Wild Card
- Faster Dashboards ***(new since EAP 2)***
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.3 EAP 3

Download EAP

Interested in a prototype plugin that simplifies JIRA project administration? http://atlss.in/ik76wI

Upgrading to JIRA 4.3 EAP 3

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.3 Upgrade Guide.

Do not use in production

- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.3 EAP 3

1

Add another application's Gadgets to your JIRA Gadget Directory *(new since EAP 2)*

JIRA 4.3 EAP 3 includes Atlassian Gadgets 2.0, allowing you to quickly add all gadgets from your Confluence, Bamboo, FishEye or Crucible instance — or from another JIRA instance — to your JIRA Gadget Directory, for easy addition to your JIRA dashboard:

In the JIRA Gadget Directory , you can now click 'Gadget Subscriptions'. There you can provide the URL for the other application (or other JIRA instance), and all the gadgets from that instance will be added to your JIRA Gadget Directory.

^Top

2

Search the Change History with JQL 'WAS' Function

We have released the first cut of Change History searching. There is much more functionality to follow, but for now you can search for changes to the Status field.

For example, the following will return all issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

Note that autocomplete has not yet been implemented, but full error reporting is supported.

^Top

3

Full LDAP integration

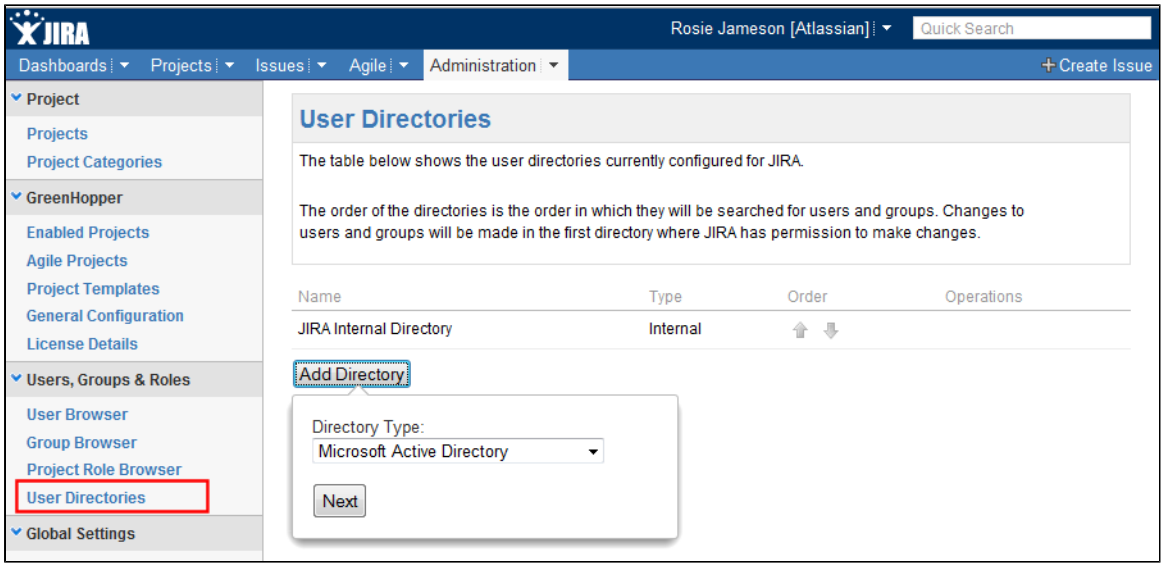
The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3. This provides a number of additional capabilities, mainly the ability to use an LDAP server for all user information. Key features are:

- Ability to connect to an LDAP server, including to Microsoft Active Directory, for user management. This includes:
  - Read-write access
  - Read-only access
  - Read-only access with local groups
  - Use LDAP for authentication only — *Previous functionality*
- Ability to connect to a Crowd server for user management — *Previous functionality*
- Ability to manage users fully within JIRA — *Previous functionality*
- Ability to use 2 or more LDAP and/ or Crowd servers simultaneously for user management.

^Top

User Directory management via JIRA Admin UI

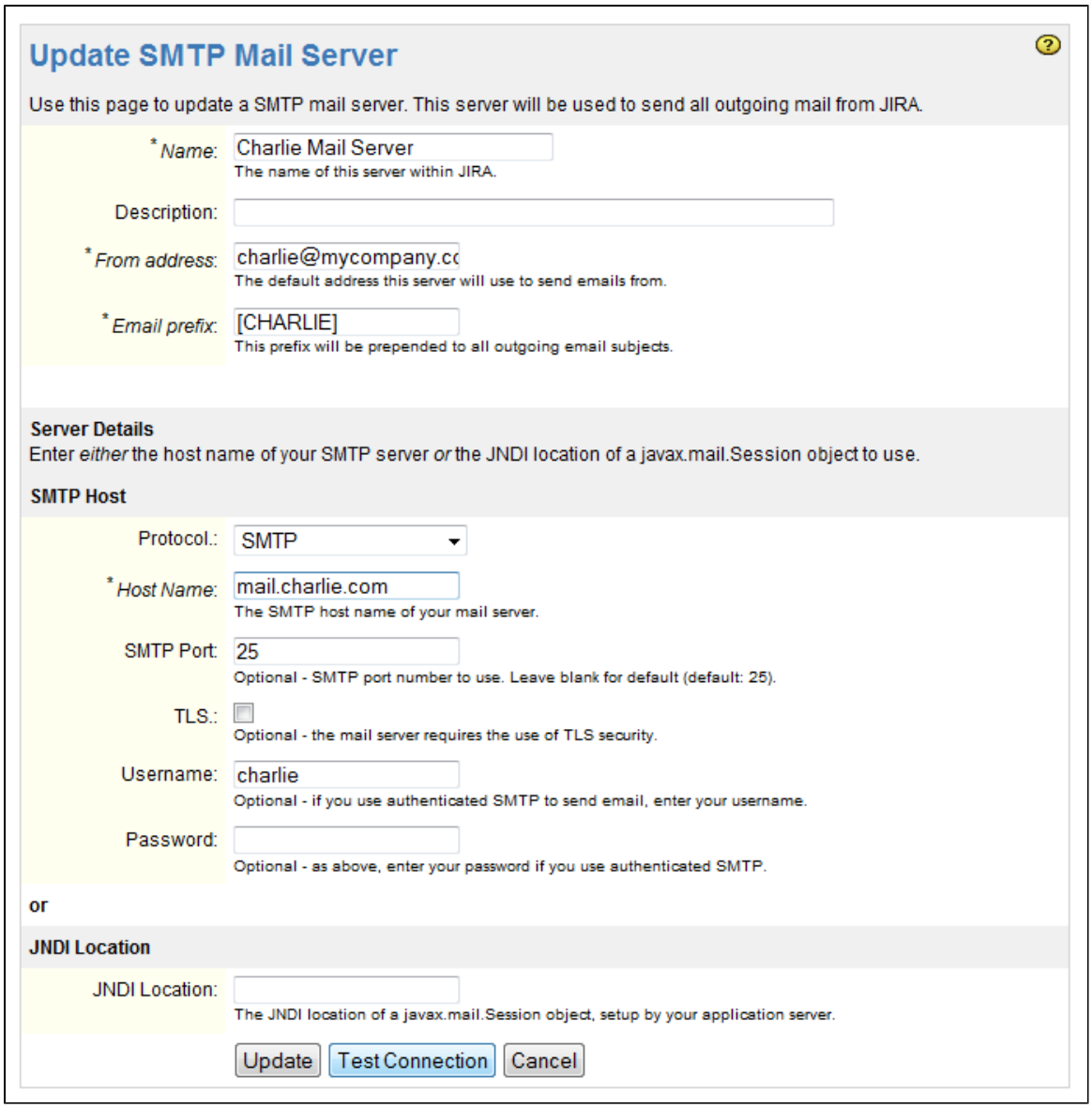
In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.



^Top

Mail Server Configuration Improvements

In JIRA 4.3, administrators can now test their mail server configuration with the "Test Connection" button.

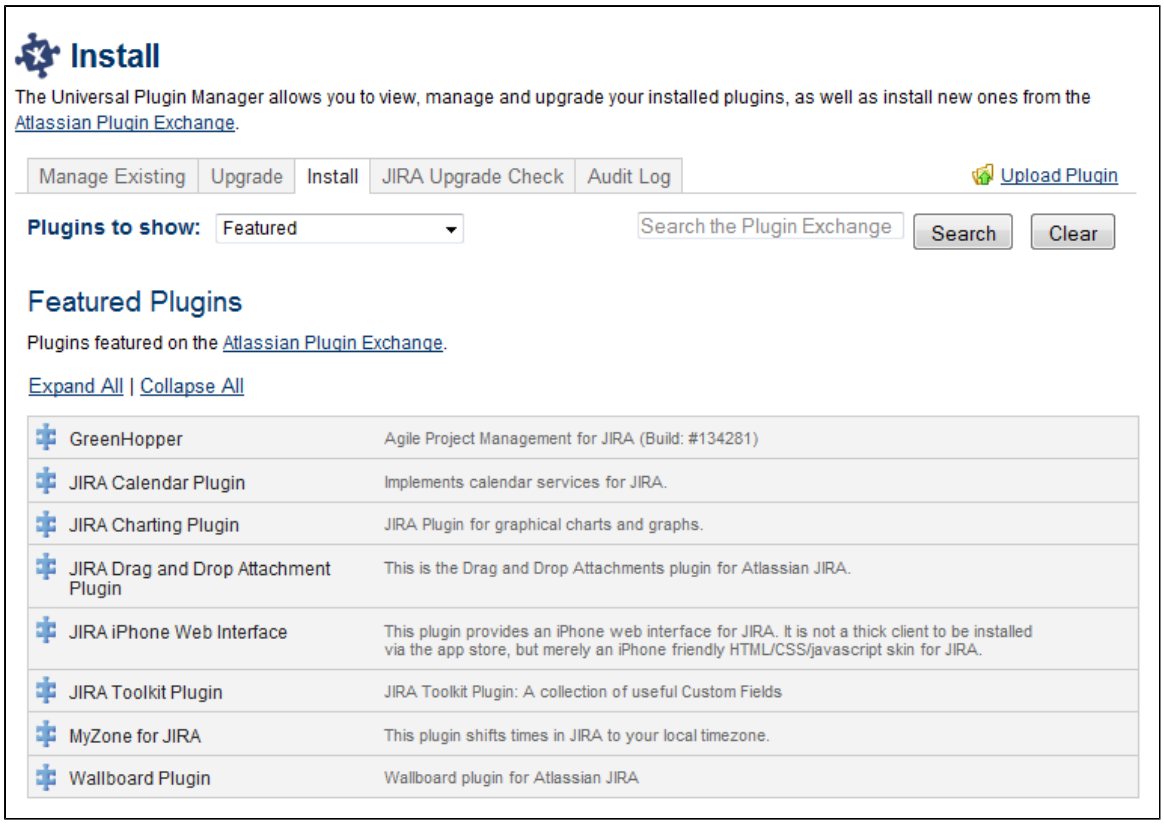


^Top

New Plugin Management System

JIRA 4.3 includes the Universal Plugin Manager (UPM), which provides you with a simple way of adding and managing plugins:

- Auto-discovery of available plugins (see **Featured Plugins** in the screenshot below)
- Point-and-click installation — no more downloading JAR files



^Top

AppLinks pre-installed and configurable from the JIRA Admin UI *(new since EAP 2)*

JIRA 4.3 EAP 3 includes Unified Application Links (UAL), so you no longer need to install it separately.

Application Links (AppLinks) is a plugin that allows you to link your JIRA, Confluence, FishEye, Crucible, Bamboo and Subversion applications. You can even choose to associate individual entities (i.e. JIRA projects, Confluence spaces, FishEye repositories, FishEye projects, Crucible projects, Bamboo projects) with each other.

Linking two applications allows you to share information and access one application's functions from within the other. For example, if you linked a JIRA server and a Confluence server, you could view JIRA issues in a Confluence page via the JIRA Issues Macro.

You also can associate entities of two linked applications. For example, you could associate a JIRA project with a Confluence space. This allows you to take advantage of additional integration features like link rendering, which lets you create links to issues or pages in the project or space using a simple textual reference, e.g. `[JRA-1234], [myConfluenceSpace:Test Page]`.

- In JIRA 4.3 EAP 3 there is now a '**Managed Unified Application Links**' option on the '**Manage Project**' screen, enabling you to easily configure application links for a given project.

For more about configuring Application Links, please see the Application Links Administrator's Guide.

^Top

Quick Search now has a Wild Card

When using Quick Search to find issues with a particular version, you can now use the wildcard symbol: `"*"`. For example, `"ff:3.2*"` will match any issue whose Fix For Version is:

- 3.2
- 3.2-beta
- 3.2.1
- 3.2.x

^Top

Faster Dashboards *(new since EAP 2)*

We are extremely pleased to announce that you should see a noticeable reduction in the time it takes to load a Dashboard.

- On average, dashboard performance has improved by 29%
- For large dashboards, the performance improvement is 35%

^Top

REST API improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance. Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3 so far, click here.

^Top

JIRA 4.3 EAP 2 Release Notes

7 December 2010

JIRA 4.3 EAP 2 (a.k.a 4.3 milestone 2 or 'm2') is a public development release leading up to **JIRA 4.3**. An **Early Access Preview (EAP) release** is a a snapshot of our work in progress, primarily focused on allowing JIRA users to see the new features in advance and provide us with some useful feedback. It also gives plugin developers an opportunity to test and fix their plugins in advance of an official release. For all production use and testing of JIRA, please use the latest official release.

The Atlassian team is proud to bring you the JIRA 4.3 EAP 2 release. Development work on JIRA 4.3 has only commenced very recently, but we want your involvement from the earliest days. Please provide feedback here.

Identity management comes of age in JIRA 4.3, with complete LDAP integration. Additionally, you now have the ability to search an issue's change history. JIRA 4.3 also includes the Universal Plugin Manager for easier management of plugins.

Note to developers: JIRA 4.3 EAP 2 includes the Atlassian Plugin Framework version 2.6. Also please see the Plugin Developer Notes for JIRA 4.3, plus note that the REST API will continue to evolve through the 4.3 EAP releases — please watch the Developer blog for the latest news on the REST API.

Highlights of JIRA 4.3 EAP 2:

- Search the Change History with JQL 'WAS' Function ***(new since EAP 1)***
- Full LDAP integration ***(Crowd 2.1 connectivity is new since EAP 1)***
- User Directory management via JIRA Admin UI
- Mail Server Configuration Improvements
- New Plugin Management System ***(new since EAP 1)***

- Quick Search now has a Wild Card ***(new since EAP 1)***
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.3 EAP 2
Download EAP

Upgrading to JIRA 4.3 EAP 2

JIRA EAP releases are available here. When upgrading, please follow the JIRA 4.3 Upgrade Guide.

Do not use in production

- **EAP releases are not safe** — EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No upgrade path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

Highlights of JIRA 4.3 EAP 2

Search the Change History with JQL 'WAS' Function *(new since EAP 1)*

In EAP 2 we are releasing the first cut of Change History searching. There is much more functionality to follow, but for now you can search for changes to the Status field.

For example, the following will return all issues that currently have, or previously had, a status of 'In Progress':

```
status WAS "In Progress"
```

Note that autocomplete has not yet been implemented, but full error reporting is supported.

^Top

Full LDAP integration *(Crowd 2.1 connectivity is new since EAP 1)*

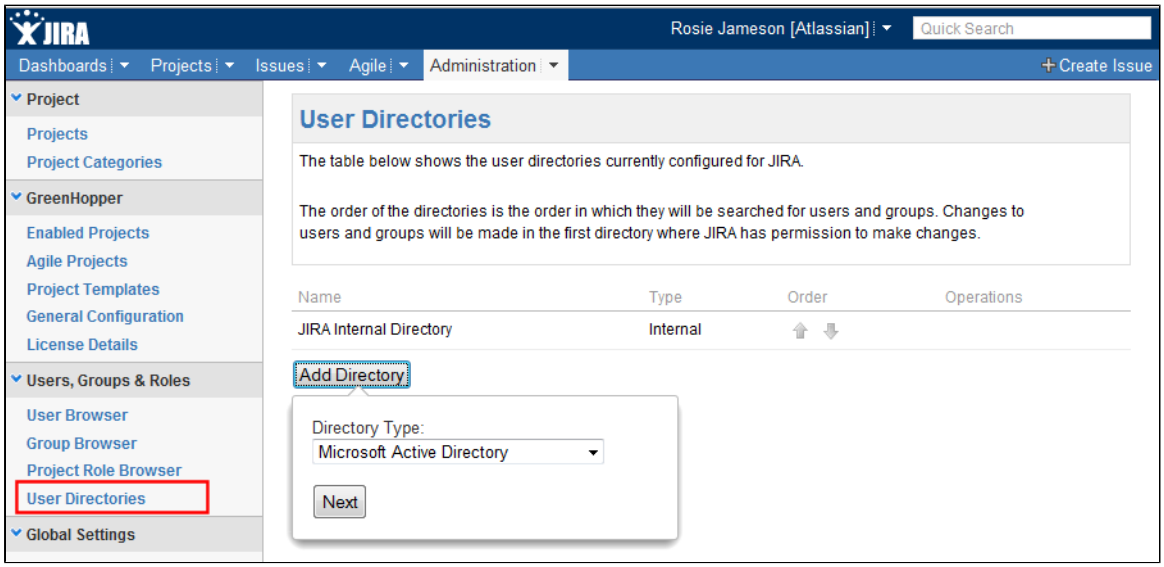
The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3. This provides a number of additional capabilities, mainly the ability to use an LDAP server for all user information. Key features are:

- Ability to connect to an LDAP server, including to Microsoft Active Directory, for user management. This includes:
  - Read-write access
  - Read-only access
  - Read-only access with local groups
  - Use LDAP for authentication only — *Previous functionality*
- Ability to connect to a Crowd server for user management — *Previous functionality*
- Ability to manage users fully within JIRA — *Previous functionality*
- Ability to use 2 or more LDAP and/ or Crowd servers simultaneously for user management.

^Top

User Directory management via JIRA Admin UI

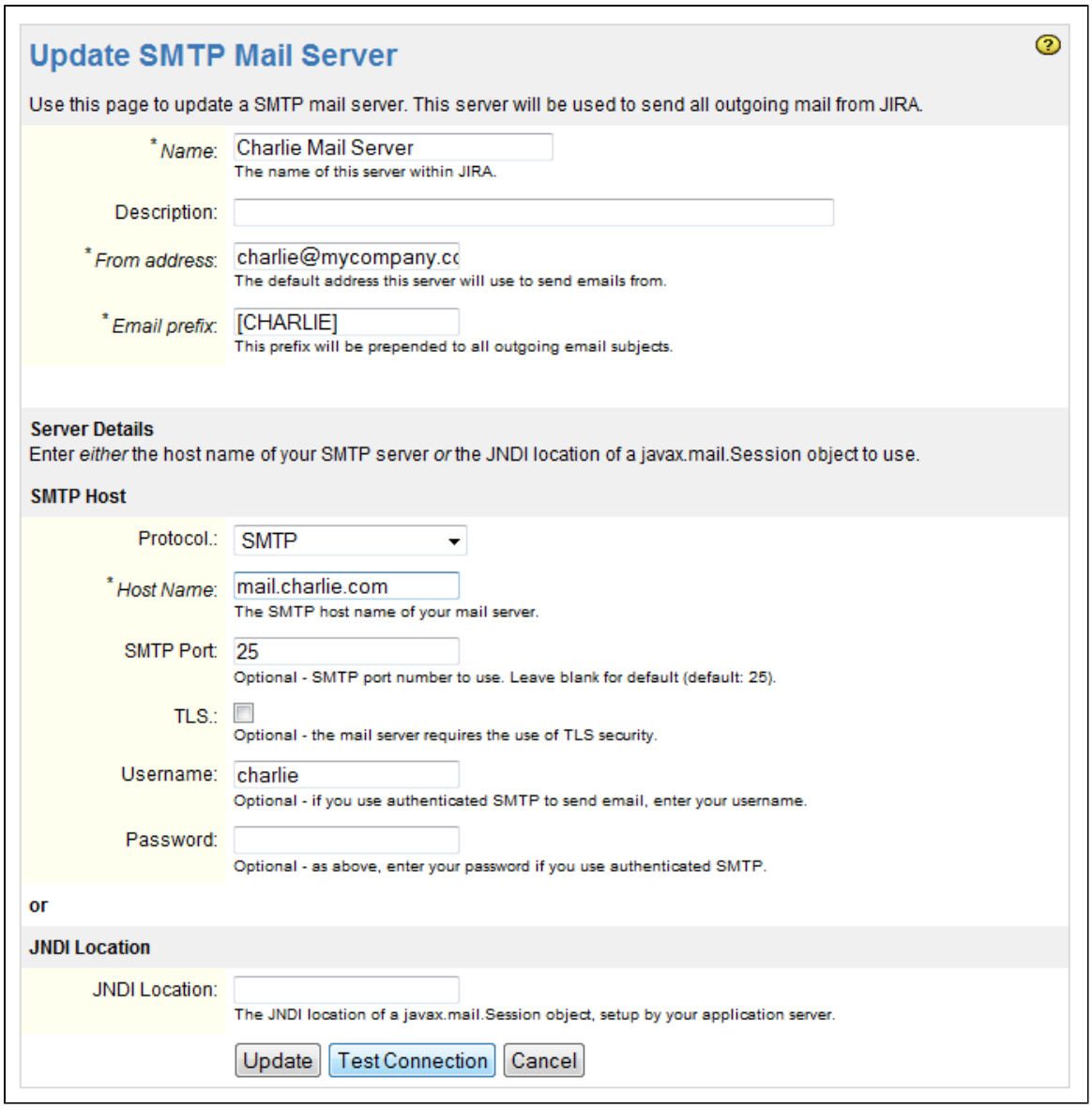
In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.



^Top

Mail Server Configuration Improvements

In JIRA 4.3, administrators can now test their mail server configuration with the "Test Connection" button.

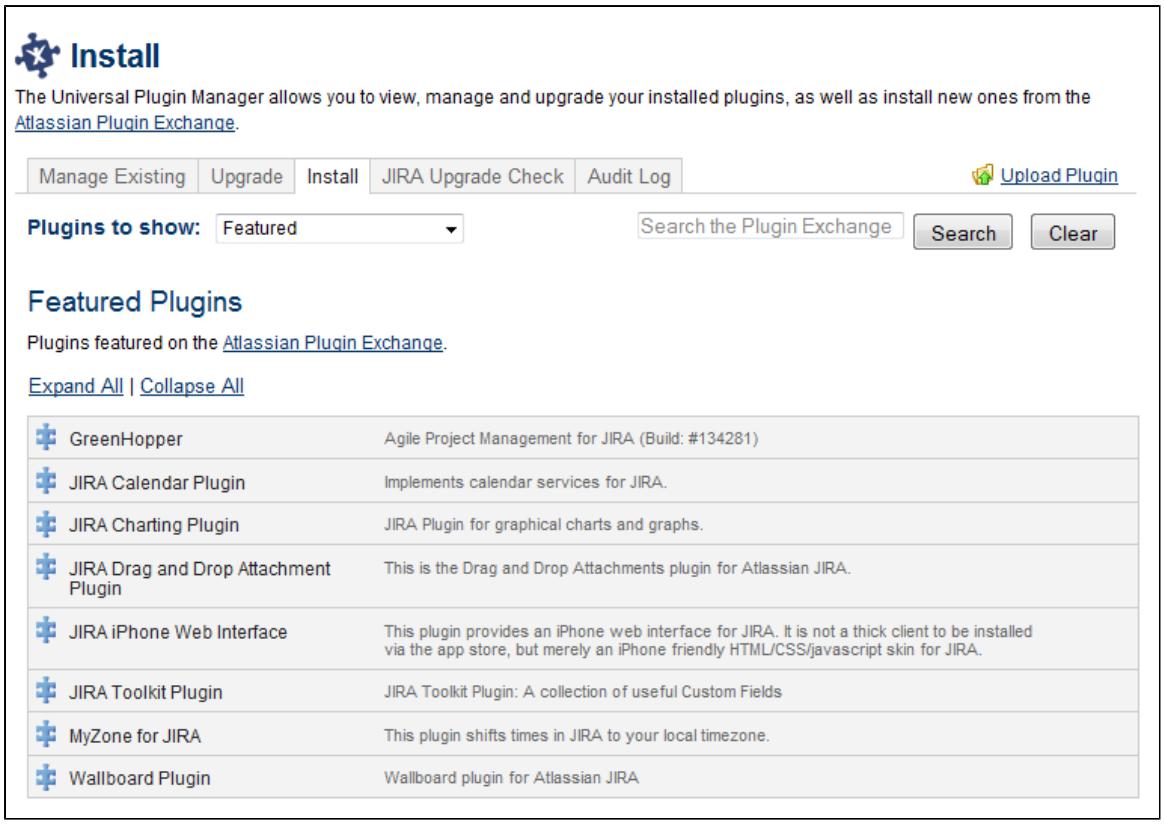


^Top

New Plugin Management System *(new since EAP 1)*

JIRA 4.3 includes the Universal Plugin Manager (UPM), which provides you with a simple way of adding and managing plugins:

- Auto-discovery of available plugins (see **Featured Plugins** in the screenshot below)
- Point-and-click installation — no more downloading JAR files and restarting JIRA



^Top

Quick Search now has a Wild Card *(new since EAP 1)*

When using Quick Search to find issues with a particular version, you can now use the wildcard symbol: **`"". For example, "ff:3.2"`** will match any issue whose Fix For Version is:

- 3.2
- 3.2-beta
- 3.2.1
- 3.2.x

^Top

REST API improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance, and to create links between issues.

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3 so far, click here.

^Top

JIRA 4.3 EAP 1 Release Notes

18 November 2010

JIRA 4.3 EAP 1 (a.k.a 4.3 milestone 1 or 'm1') is a public development release leading up to **JIRA 4.3**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

The Atlassian team is proud to bring you the JIRA 4.3 EAP 1 release. Development work on JIRA 4.3 has only commenced very recently, but we want your involvement from the earliest days. Please provide feedback here.

Note to developers: JIRA 4.3 EAP 1 includes Atlassian Plugin Framework version 2.6. Also please see the Plugin Developer Notes for JIRA 4.3, plus note that the REST API will continue to evolve through the 4.3 EAP releases — please watch the JIRA blog for the latest news on the REST API.

Highlights of JIRA 4.3 EAP 1:

- Full LDAP integration
- User Directory management via JIRA Admin UI
- Mail Server Configuration Improvements
- REST API improvements
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.3 EAP 1
Download EAP

Upgrading to JIRA 4.3 EAP 1

JIRA EAP releases are available here. Please follow the JIRA 4.3 Upgrade Guide.

Highlights of JIRA 4.3 EAP 1

Full LDAP integration

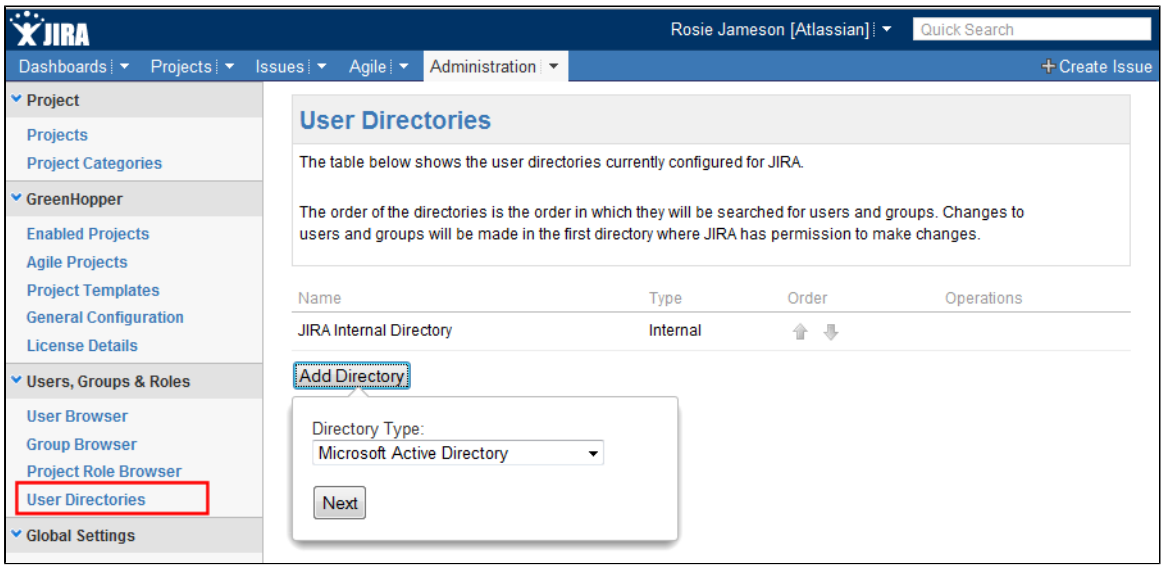
The way users and groups are stored and accessed in JIRA has been totally rewritten in Release 4.3. This provides a number of additional capabilities, mainly the ability to use an LDAP server for all user information. Key features are:

- Ability to connect to an LDAP server, including to Microsoft Active Directory, for user management. This includes:
  - Read-write access
  - Read-only access
  - Read-only access with local groups
  - Use LDAP for authentication only — *Previous functionality*

^Top

User Directory management via JIRA Admin UI

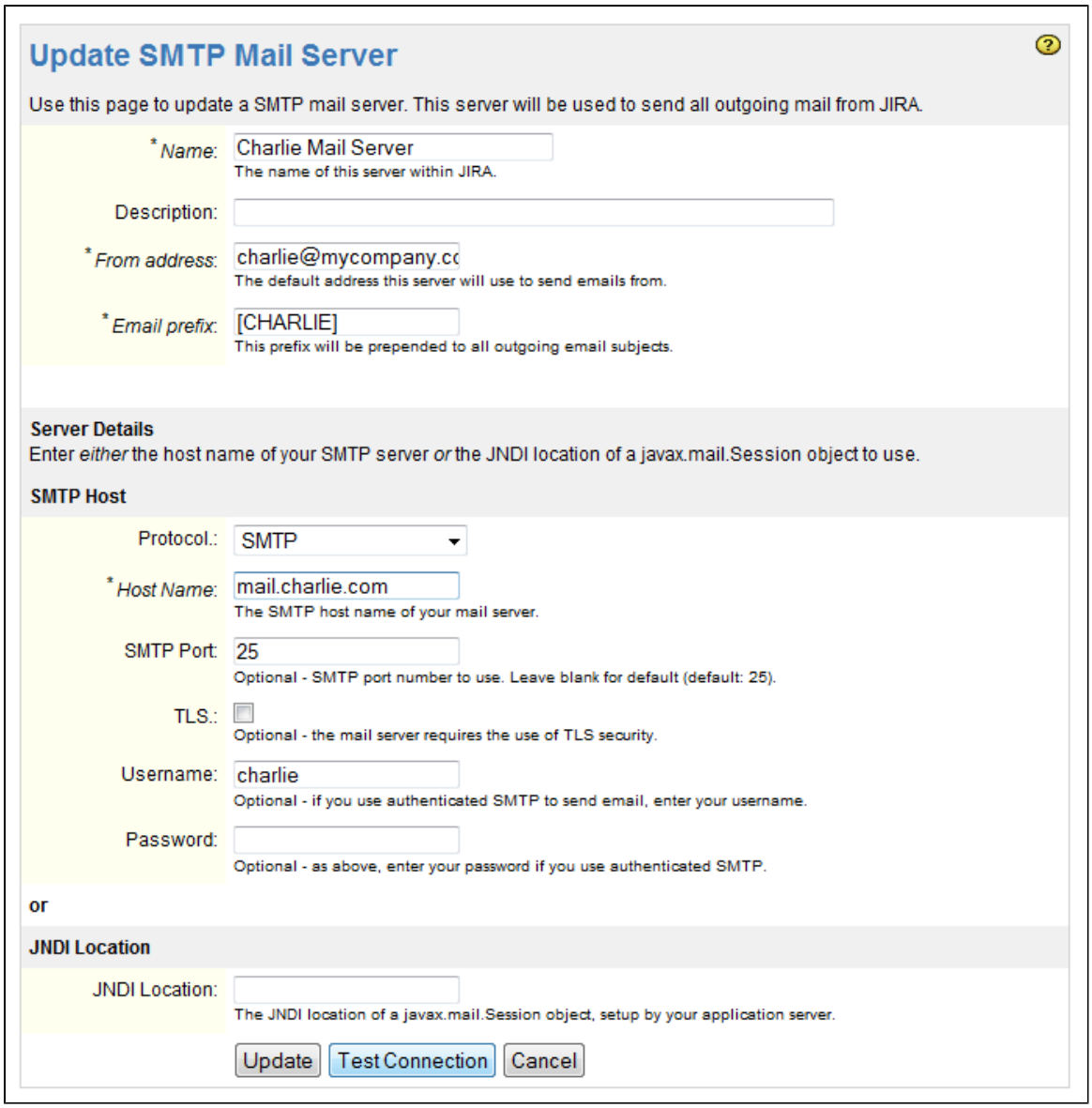
In JIRA 4.3, administrators can now manage external User Directories directly from the JIRA Administration screens. Gone are the days of manually editing XML files.



^Top

Mail Server Configuration Improvements

In JIRA 4.3, administrators can now test their mail server configuration with the "Test Connection" button.



^Top

REST API improvements

We have improved discoverability of issues and projects through the REST API: there is now a `search` resource, which can be used to search for issues using JIRA Query Language (JQL). It is also possible to obtain a list of projects in a JIRA instance. Have a look at the reference documentation to see if the REST API offers the features that you want (let us know if not).

Clients that log in through the REST API can now determine when the login has been denied due to a CAPTCHA requirement: the HTTP header `X-Authentication-Denied-Reason` has all the necessary information.

^Top

Other Enhancements and Fixes

For the list of issues resolved in JIRA 4.3 so far, click here.

^Top

JIRA 4.2 EAP 4 Release Notes

4 August 2010

JIRA 4.2 EAP 4 (a.k.a 4.2 milestone 9) is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

GreenHopper Support

Please be aware that GreenHopper 5.0.1 is not compatible with the JIRA 4.2 EAP 4 release.

The Atlassian team is proud to bring you the JIRA 4.2 EAP 4 release. We are getting closer to the final release of JIRA 4.2, but we want your feedback.

Time-tracking is now much more flexible: you can edit the Original Estimate, and set the Remaining Estimate to zero on resolving an issue. You can also log work via workflow 'transition' screens — which are now provided as in-place dialogs rather than separate screens. For the mouse-averse, the new 'Operations Dialog' provides access to all menu options via the keyboard. The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users — and everyone else who likes to label their JIRA issues. Dashboard Publish/Subscribe with Confluence is also included.

Note to developers: JIRA 4.2 includes Atlassian Plugin Framework version 2.5.

Note to partners and translators: The preliminary translation diff files will be made available when we release JIRA 4.2 beta in the coming weeks.

Highlights of JIRA 4.2 :

- Dialogs for Common Actions and Workflow Operations
- Dialogs in the Issue Navigator
- 'Operations Dialog' Replaces the Mouse
- Improvements to the 'View Issue' Screen
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- New JQL Functions
- 'Filter' Gadget creation via the Issue Navigator
- More Keyboard Shortcuts
- User Avatars and User Hover
- Labels Now Included
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.2 EAP 4
Download EAP

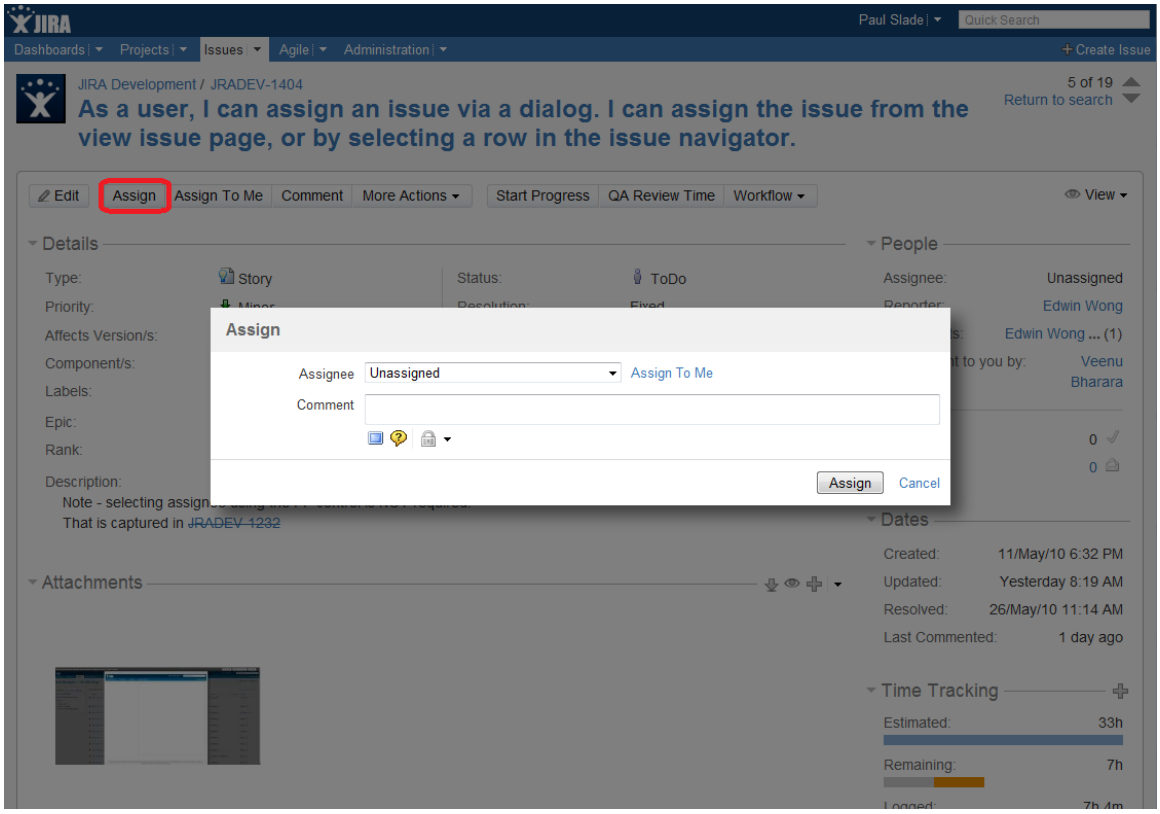
Upgrading to JIRA 4.2 EAP 4

JIRA EAP releases are available here. To upgrade from JIRA 4.1.x to this release, please follow the normal JIRA Upgrade Guide.

Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

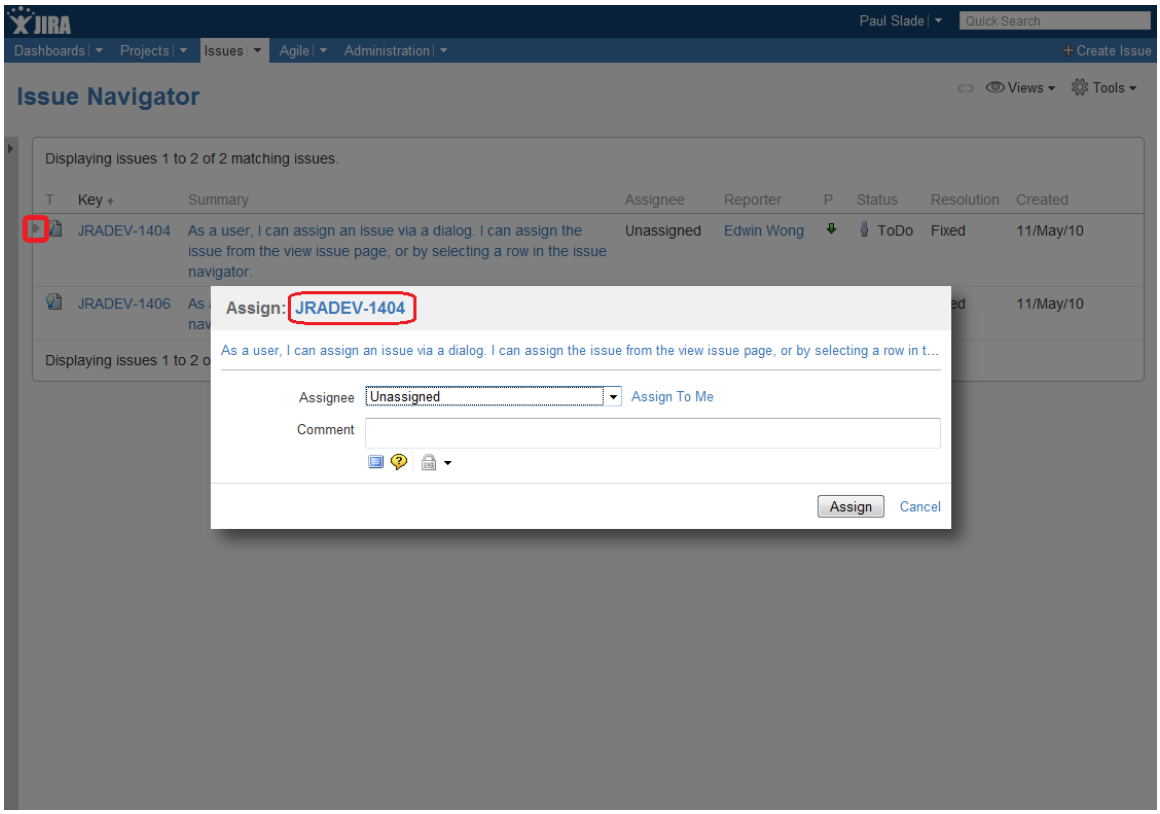
For faster edits to an issue, pop-up dialogs have replaced screen changes:



^Top

Dialogs in the Issue Navigator

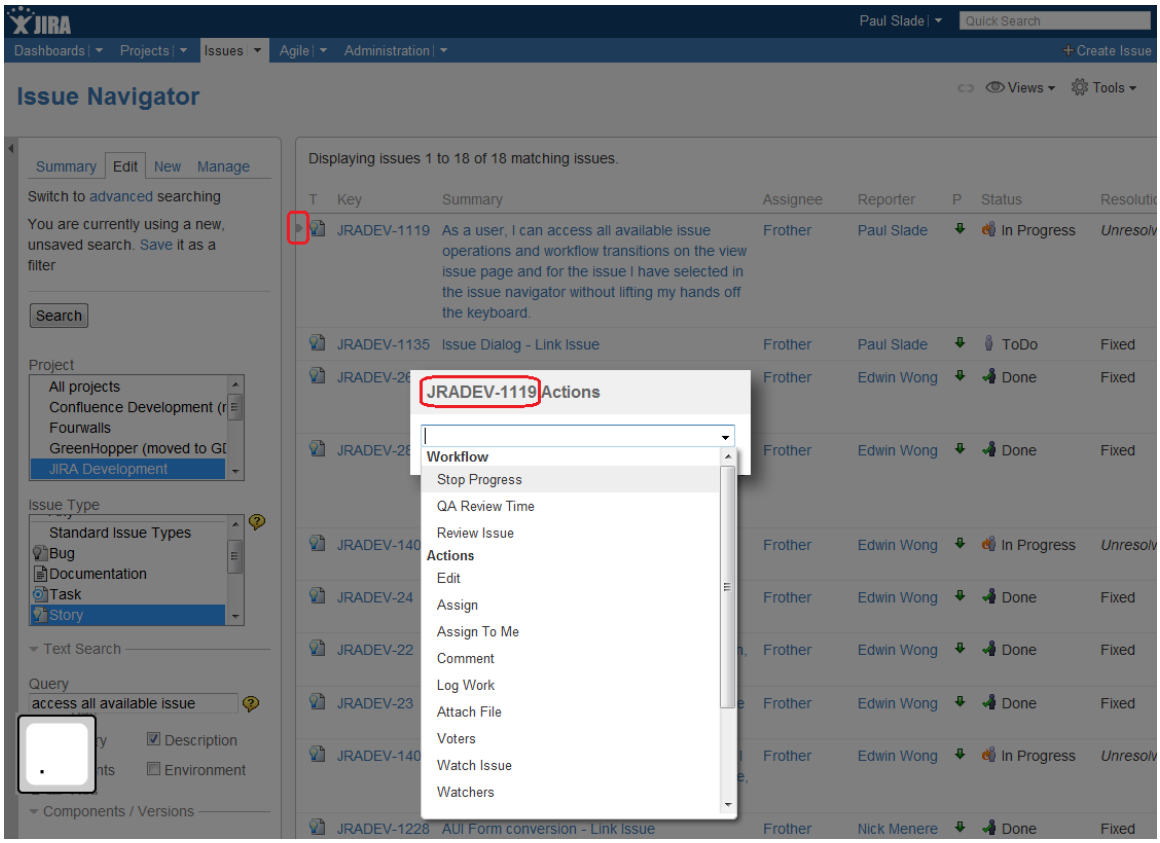
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



^Top

'Operations Dialog' Replaces the Mouse

The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.

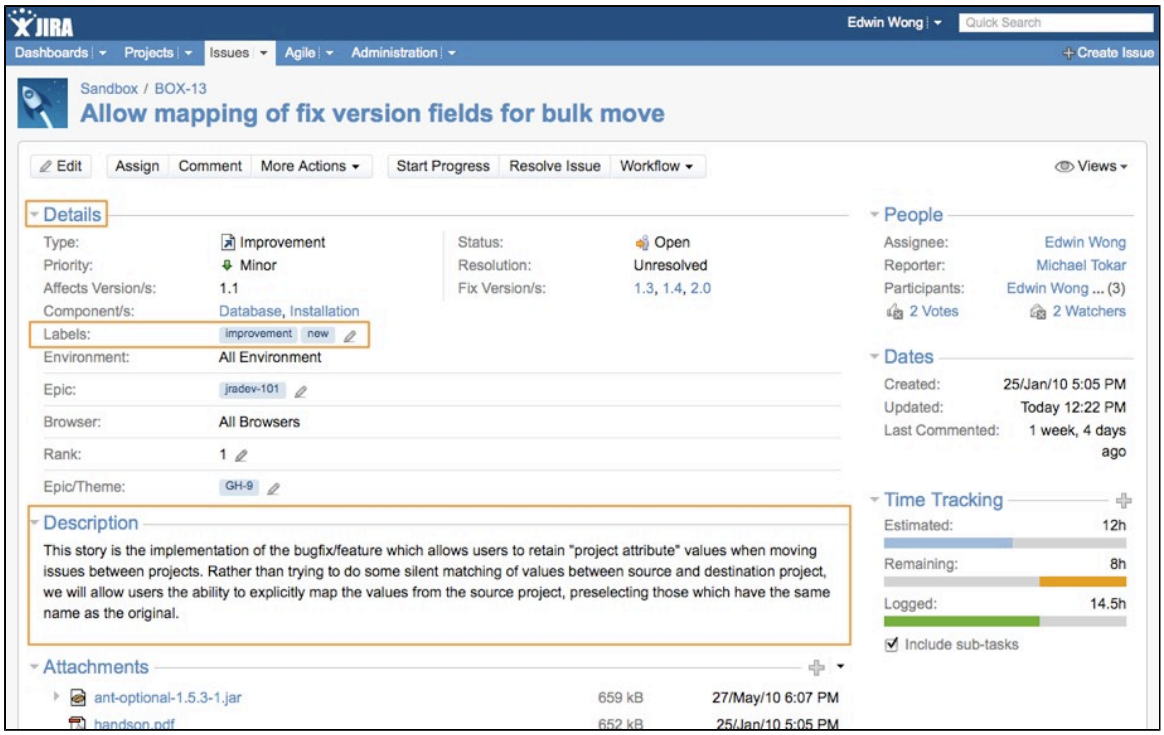


^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the view issue screen even more. Some improvements include:

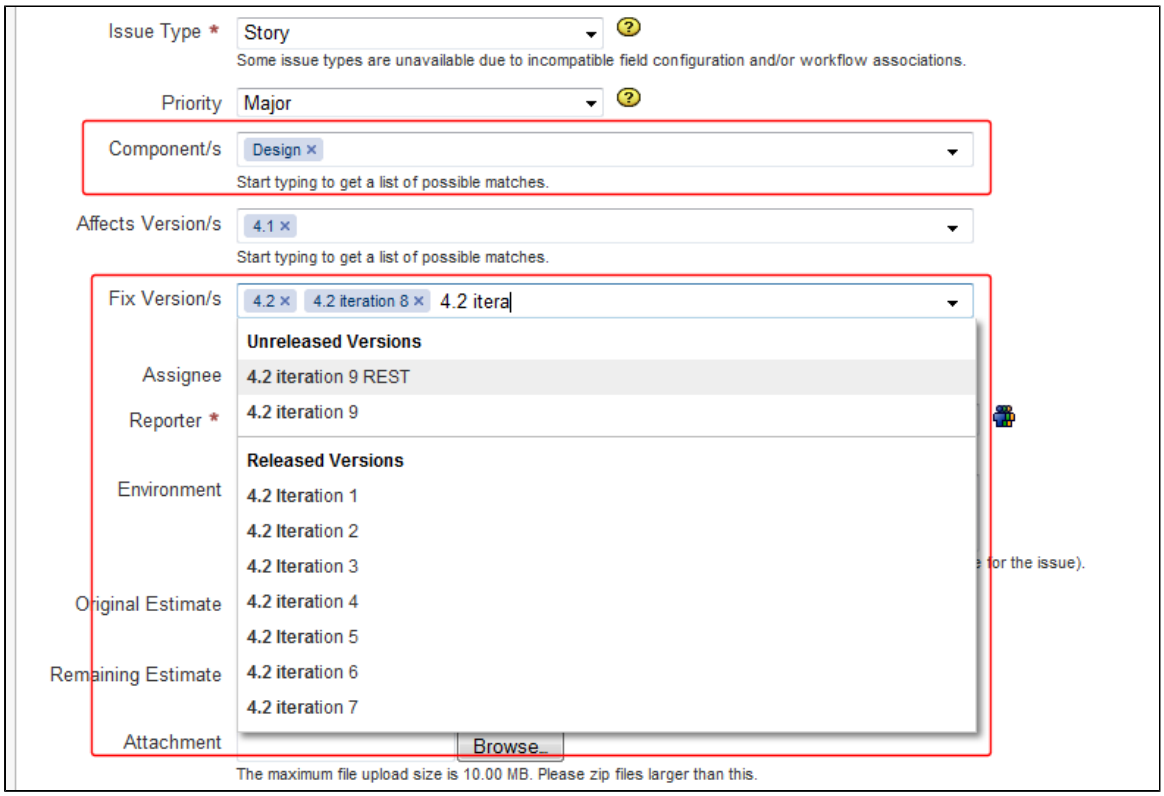
- A tighter spacing between lines to reduce space between elements
- Stronger section headings
- The description field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the commenting section.

We've also updated the look and feel of the Components and Affects/Fix Versions lists in Edit mode. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

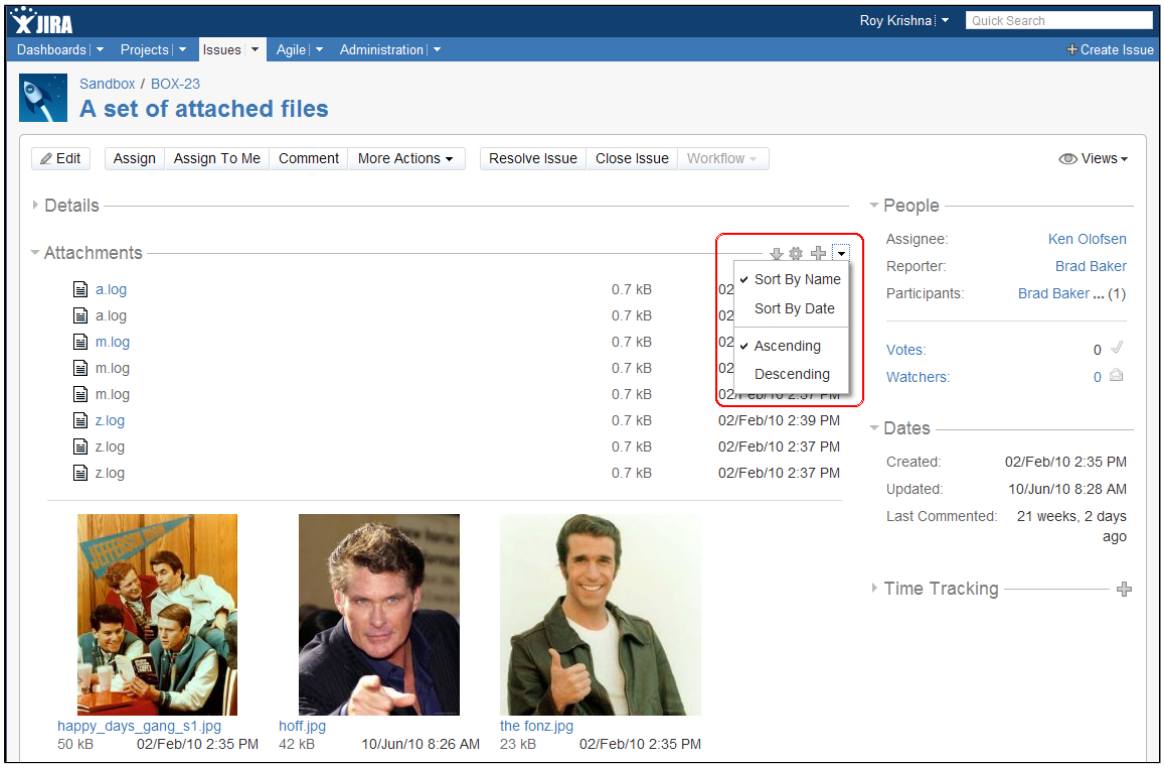
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

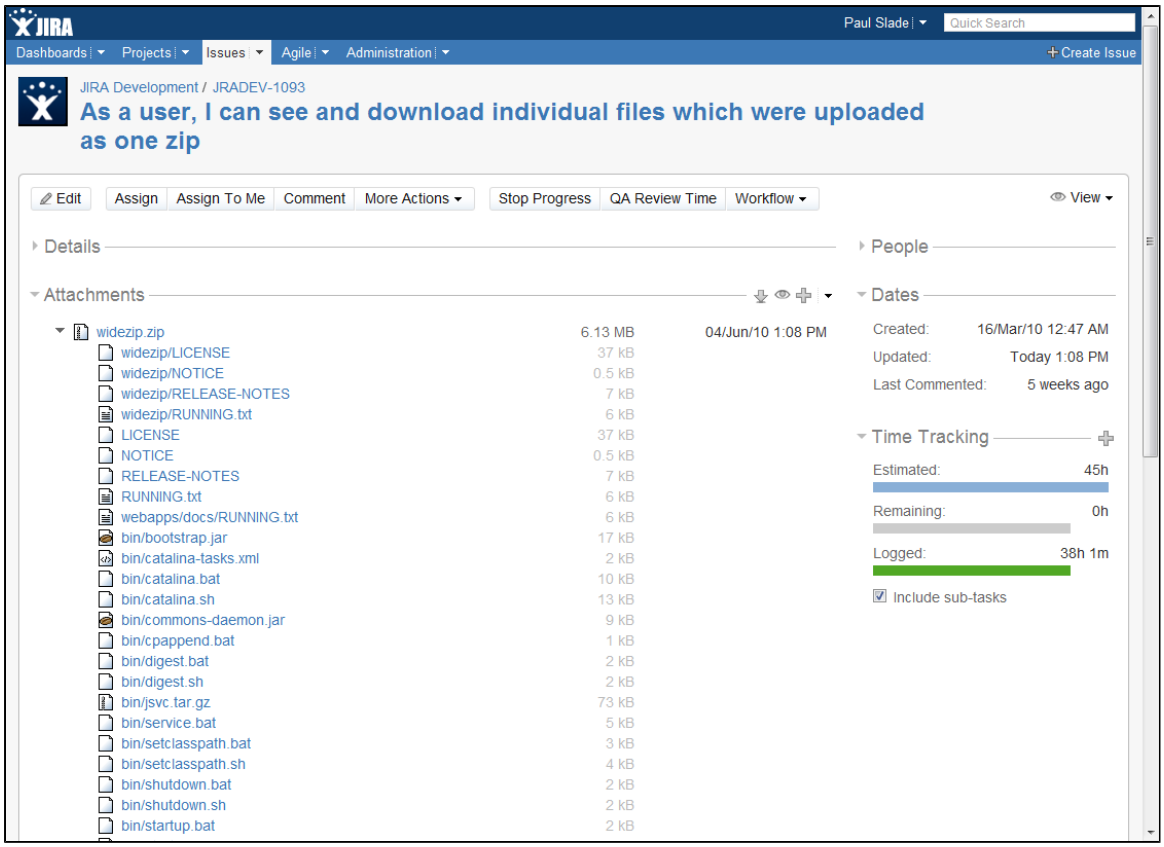
Attachments can now be sorted by date or by name, in ascending or descending order.



^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



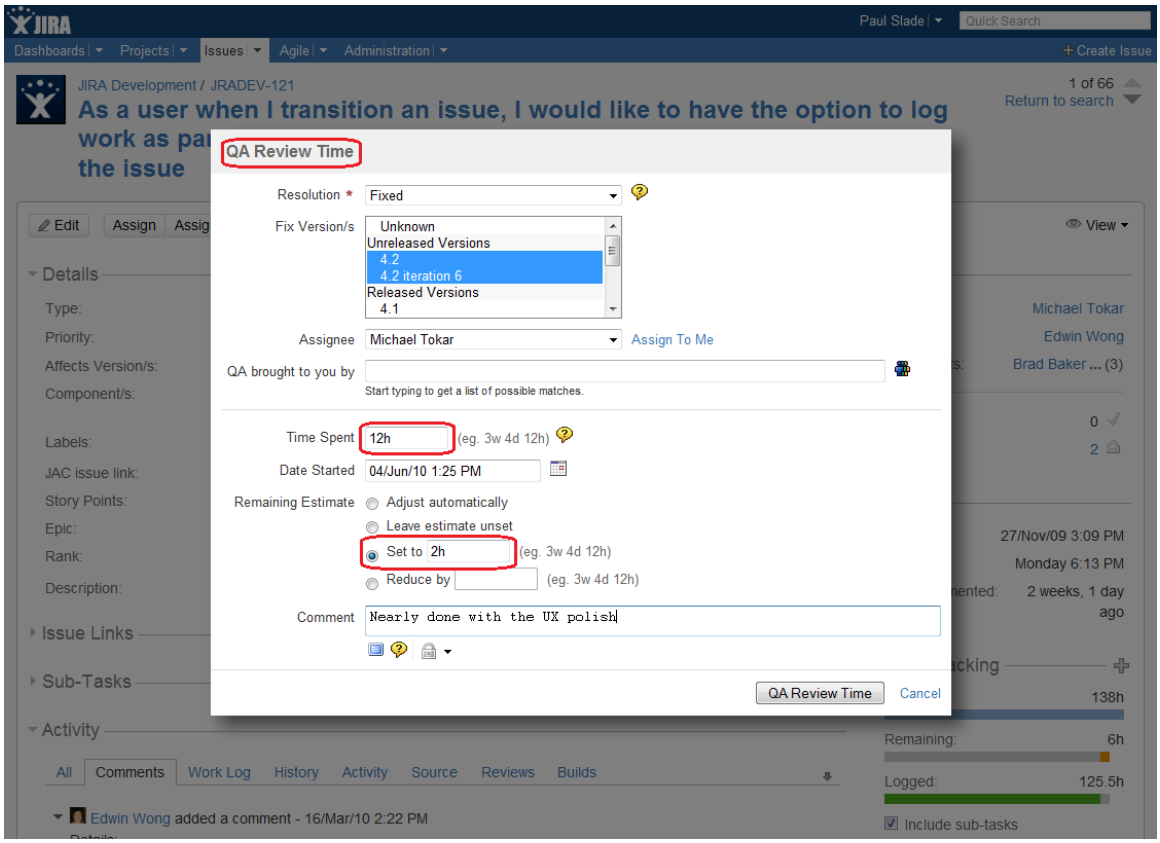
^Top

'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).



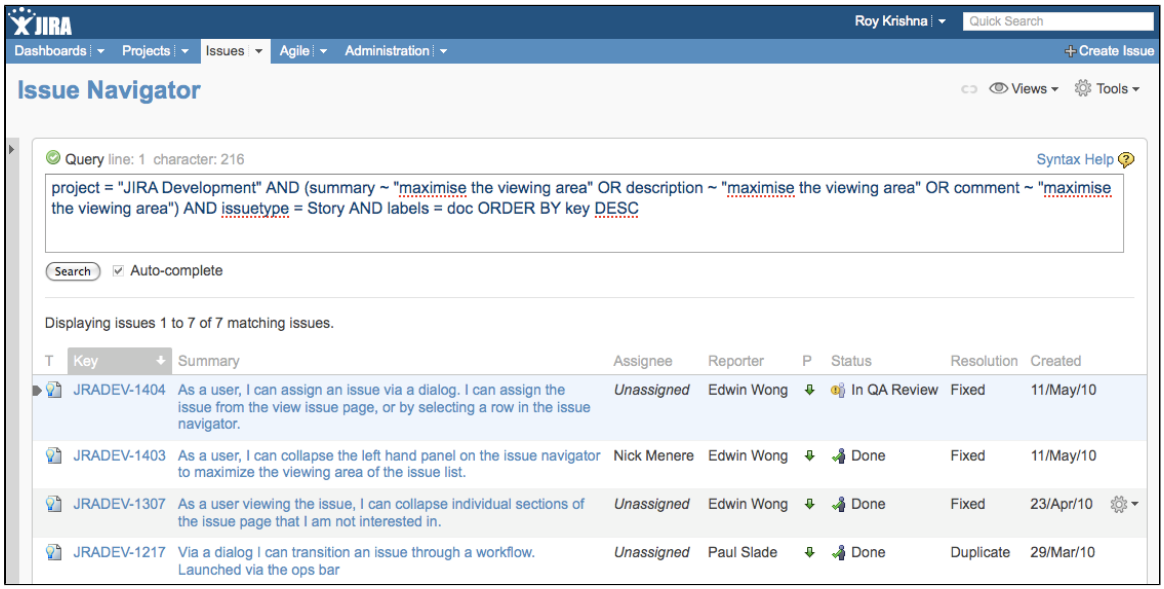
^Top

8

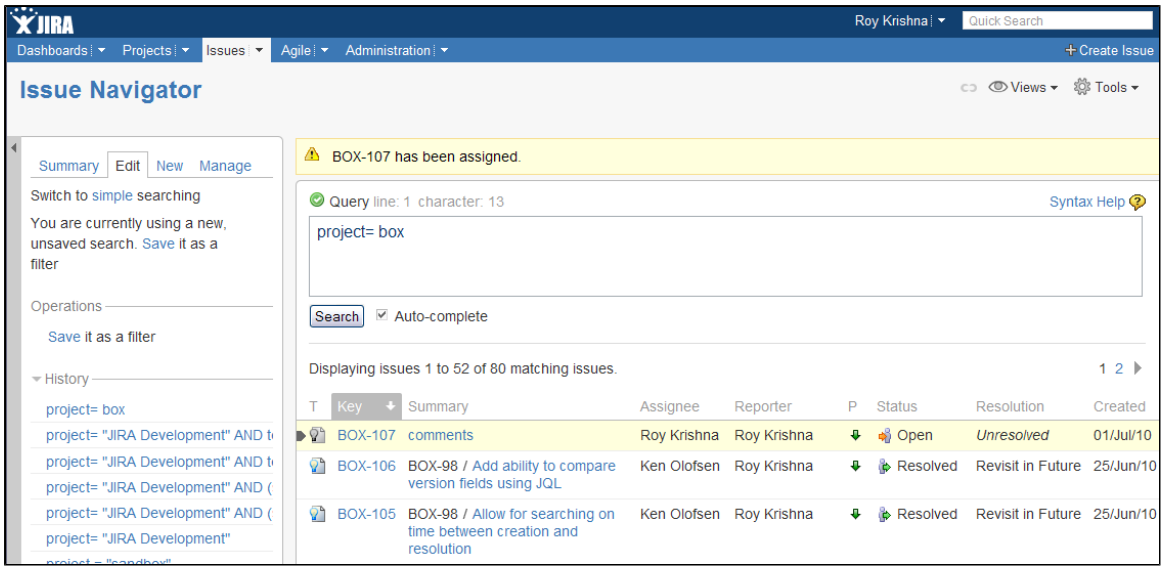
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



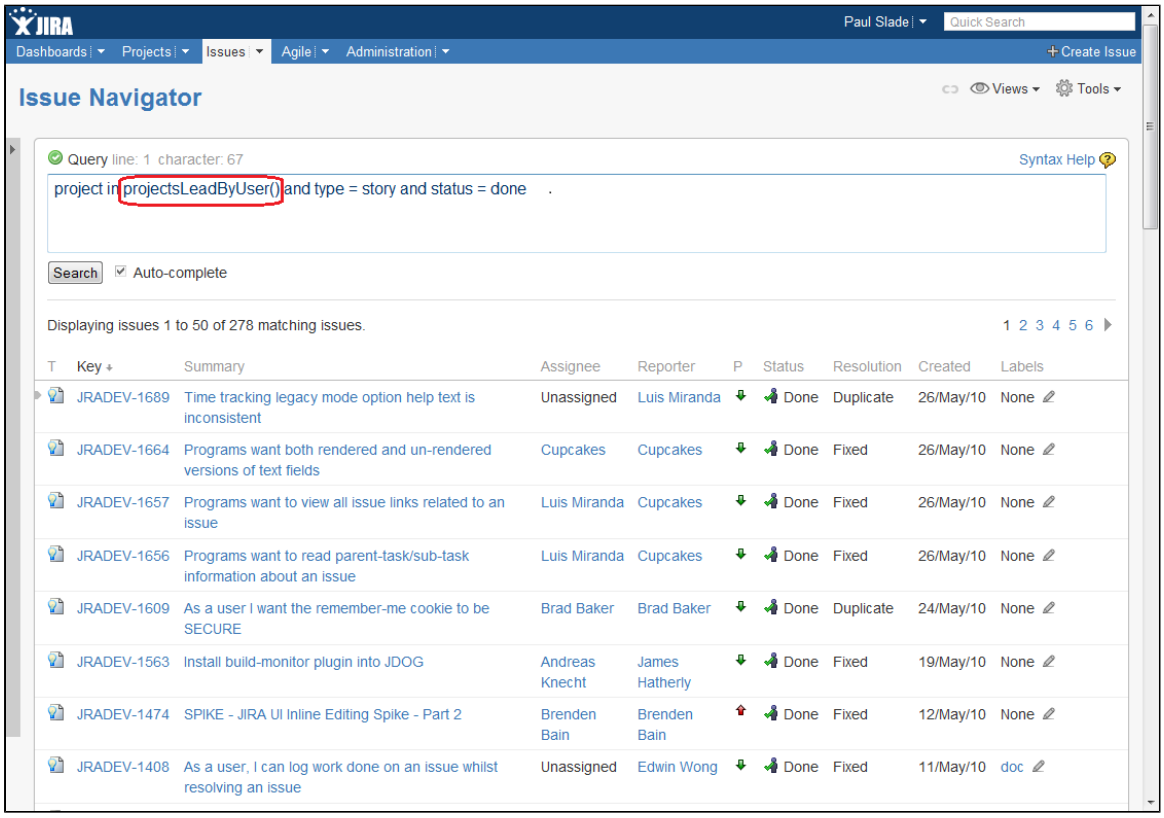
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

New JQL Functions

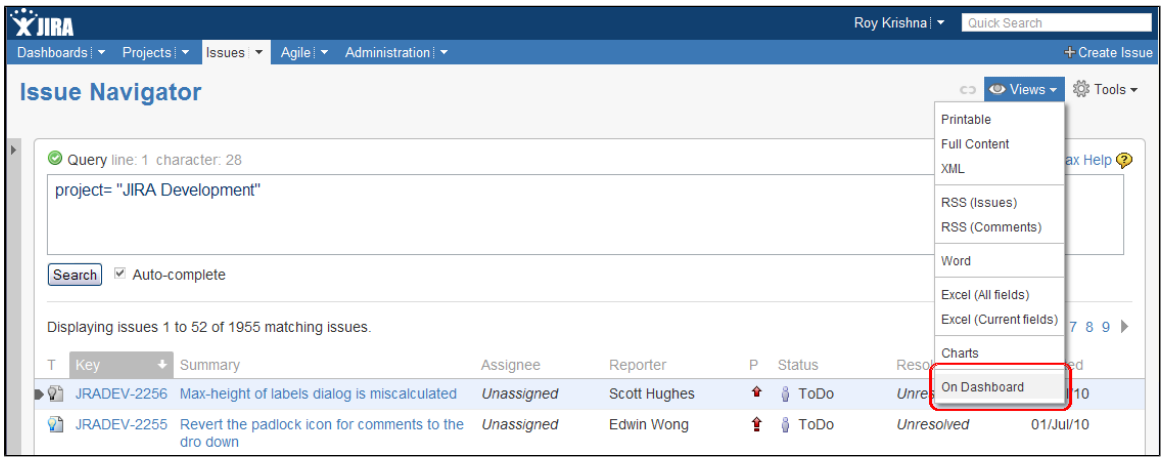
Additional JQL functions for projectLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



^Top

'Filter' Gadget creation via the Issue Navigator

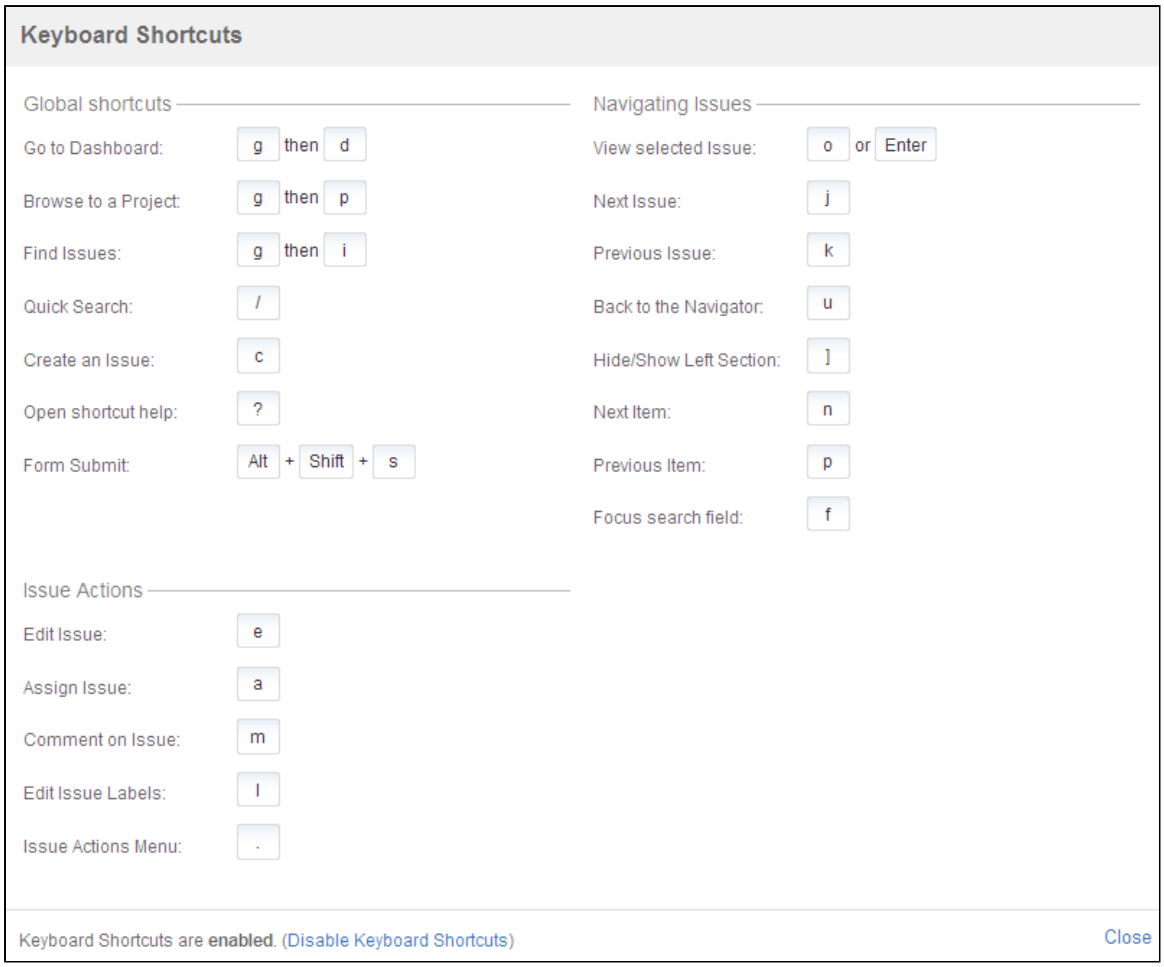
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



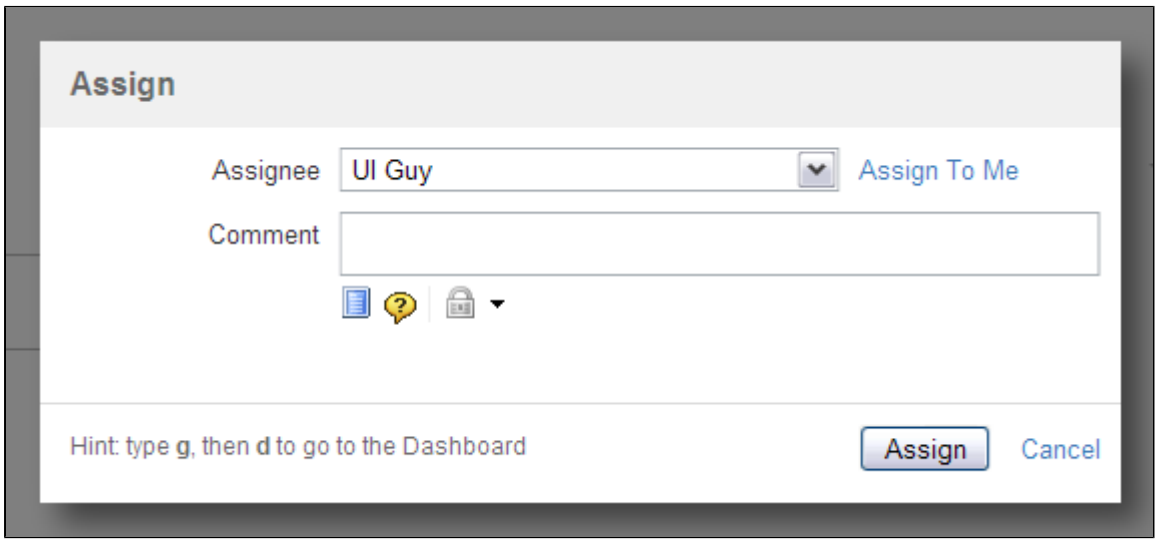
^Top

More Keyboard Shortcuts

JIRA users can now do more without a mouse --- perfect for those power users.



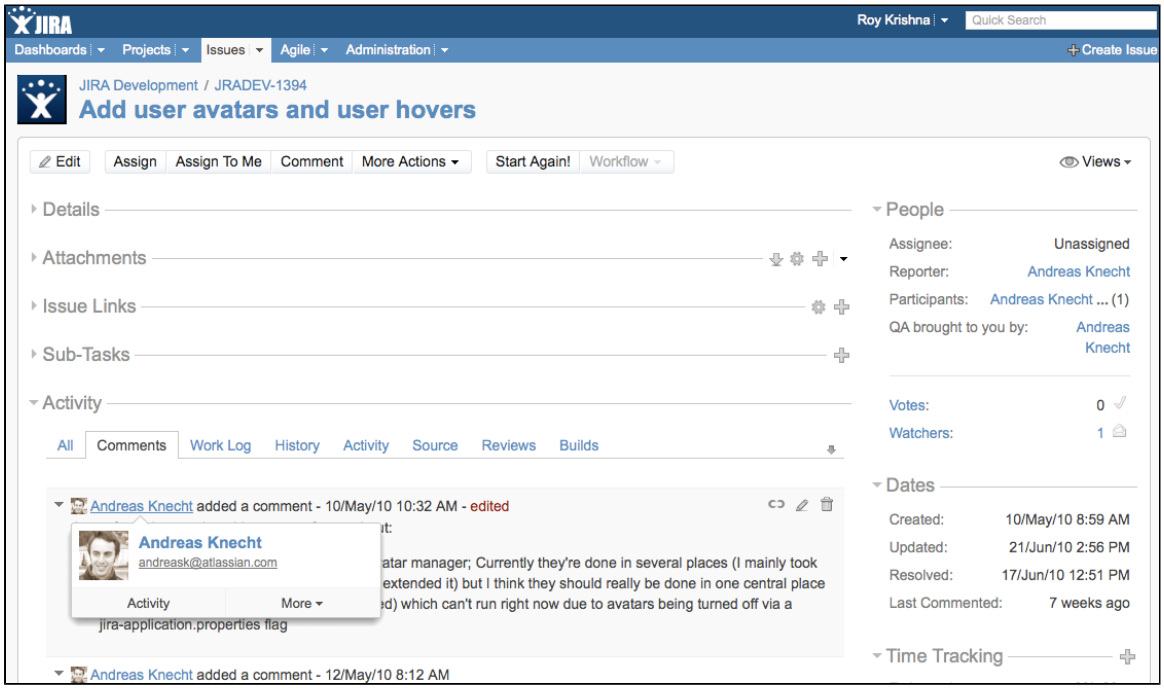
Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly.



^Top

User Avatars and User Hover

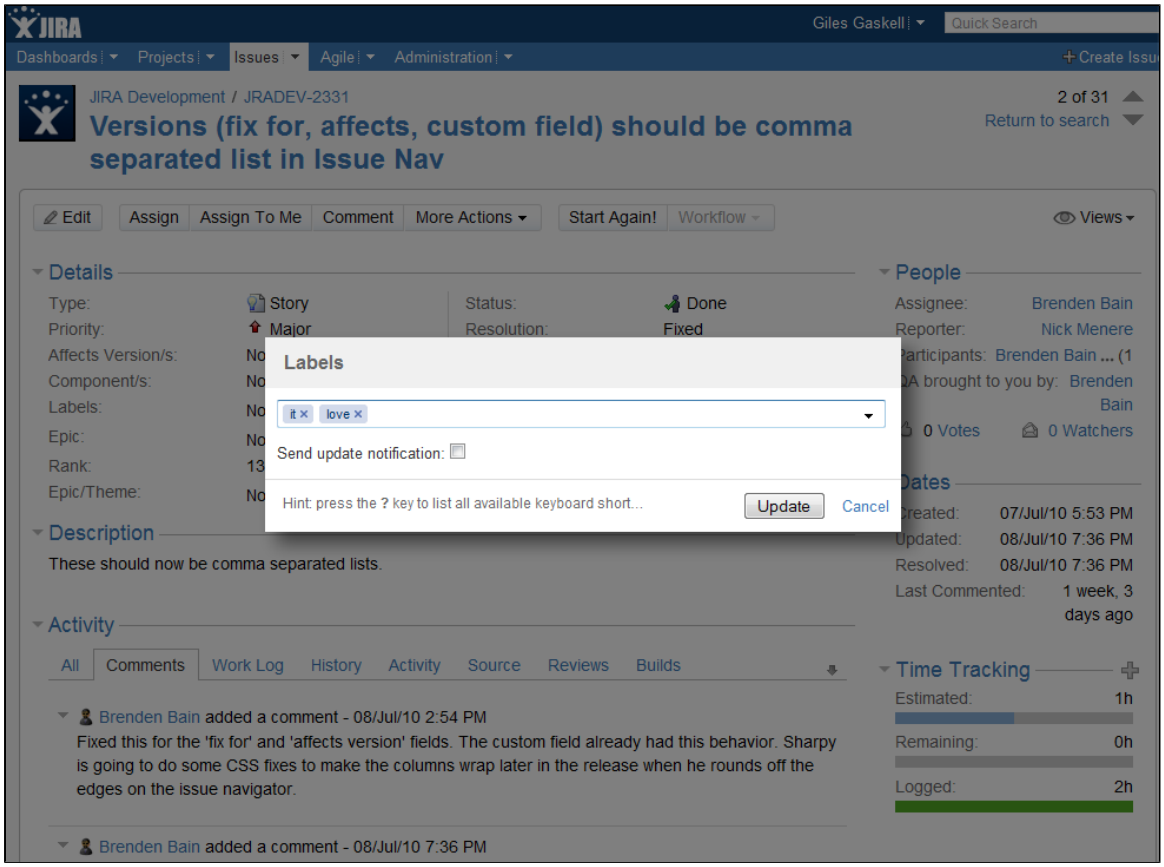
JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.



^Top

Labels Now Included

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.



^Top

Other Enhancements and Fixes

Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.2 EAP 3 Release Notes

19 July 2010

JIRA 4.2 EAP 3 (a.k.a 4.2 milestone 8) is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

GreenHopper Support

Please be aware that GreenHopper 5.0.1 is not compatible with the JIRA 4.2 EAP 3 release.

The Atlassian team is proud to bring you the JIRA 4.2 EAP 3 release. We are getting closer to the final release of JIRA 4.2, but we want your feedback.

Time-tracking is now much more flexible: you can edit the Original Estimate, and set the Remaining Estimate to zero on resolving an issue. You can also log work via workflow 'transition' screens — which are now provided as in-place dialogs rather than separate screens. For the mouse-averse, the new 'Operations Dialog' provides access to all menu options via the keyboard. The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users — and everyone else who likes to label their JIRA issues. Dashboard Publish/Subscribe with Confluence is also included.

Note to developers: JIRA 4.2 includes Atlassian Plugin Framework version 2.5.

Note to partners and translators: The preliminary translation diff files will be made available when we release JIRA 4.2 beta in the coming weeks.

Highlights of JIRA 4.2 :

- Dialogs for Common Actions and Workflow Operations
- Dialogs in the Issue Navigator
- 'Operations Dialog' Replaces the Mouse
- Improvements to the 'View Issue' Screen
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- New JQL Functions

- 'Filter' Gadget creation via the Issue Navigator
- More Keyboard Shortcuts
- User Avatars and User Hover
- Labels Now Included
- Other Enhancements and Fixes

What's New in EAP 3

- For EAP 3, we've mainly focused on improving the view issue screen. Some highlights include:
    1. Reduced line height of text in the entire page.
    2. Stronger and colored section headings.
    3. Description field separated into its own separate section.
    4. Division lines for custom fields
    5. Labels as highlighted bubbles.
    6. Votes and Watchers fields on the same line.
- When creating or editing an issue, the versions fields now have the same look and feel as the component field.
- We've also reduced the size of labels in dialog boxes.

Thank you for your interest in JIRA 4.2 EAP 3

Download EAP

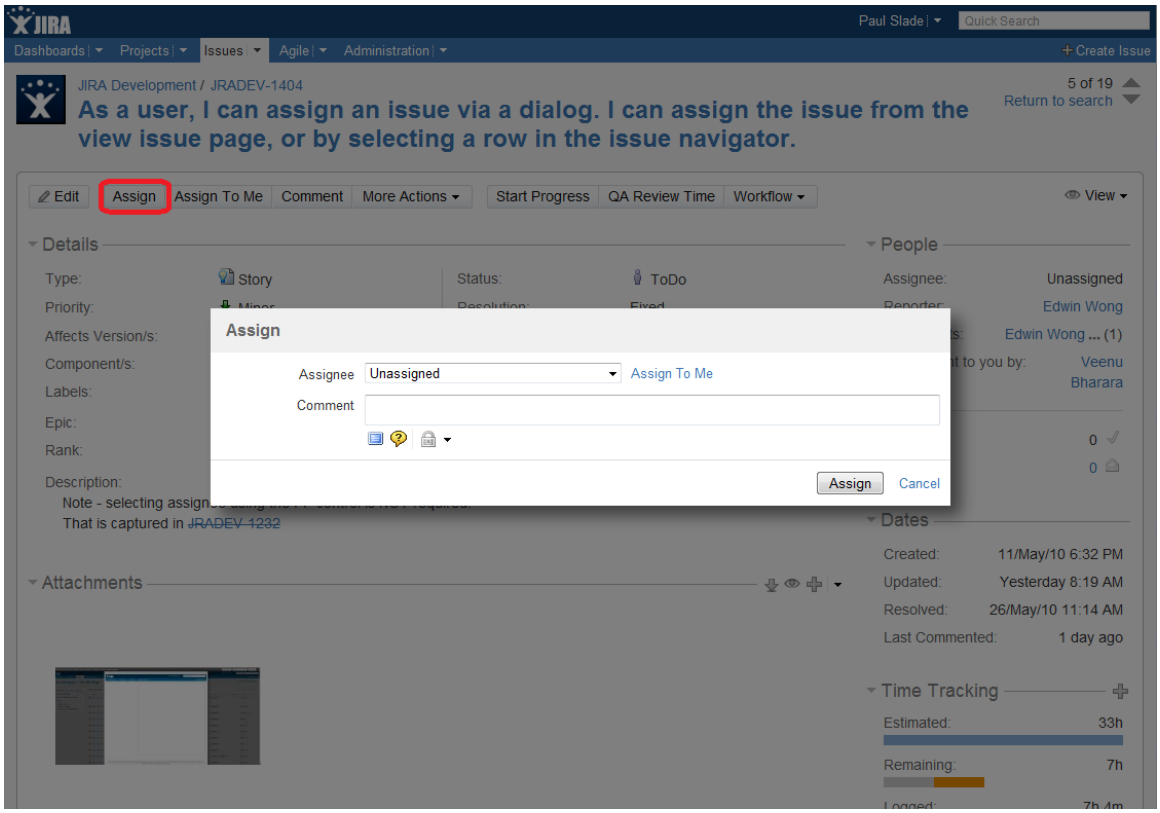
Upgrading to JIRA 4.2 EAP 3

JIRA EAP releases are available here. To upgrade from JIRA 4.1.x to this release, please follow the normal JIRA Upgrade Guide.

Highlights of JIRA 4.2

Dialogs for Common Actions and Workflow Operations

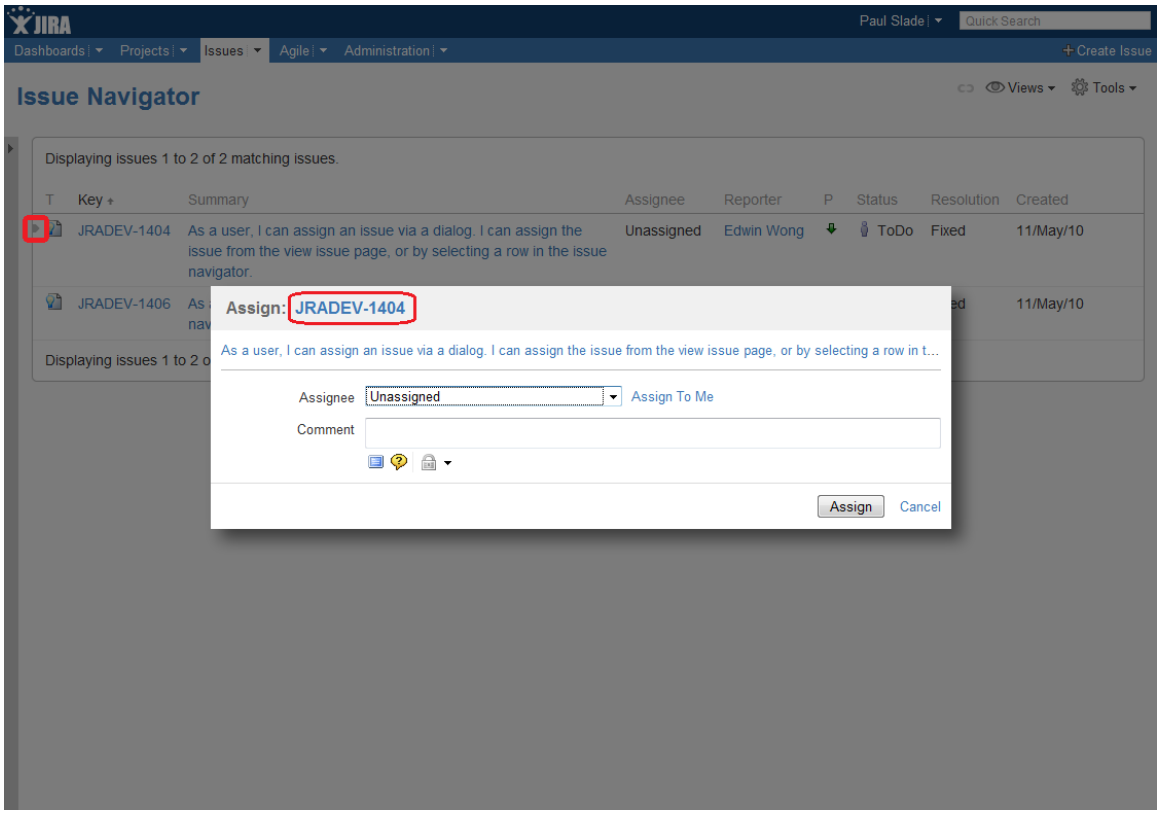
For faster edits to an issue, pop-up dialogs have replaced screen changes:



^Top

Dialogs in the Issue Navigator

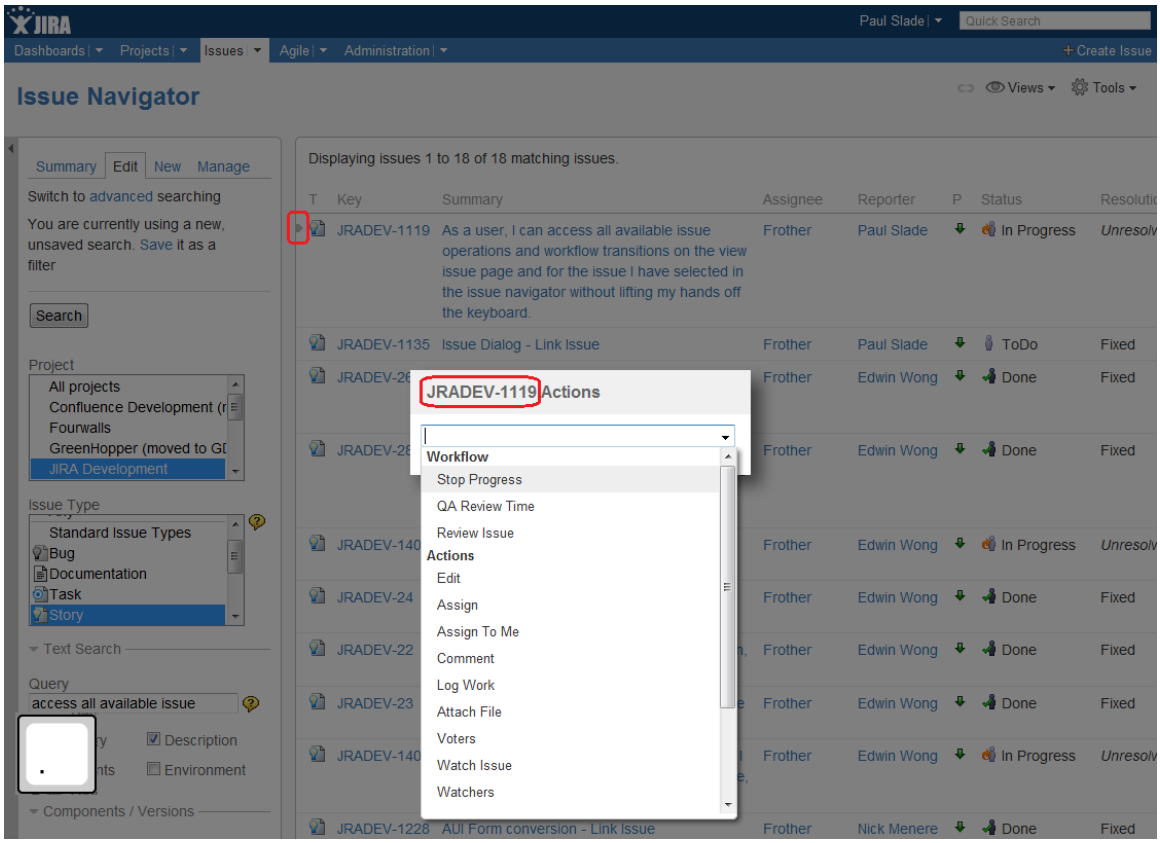
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



^Top

'Operations Dialog' Replaces the Mouse

The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.

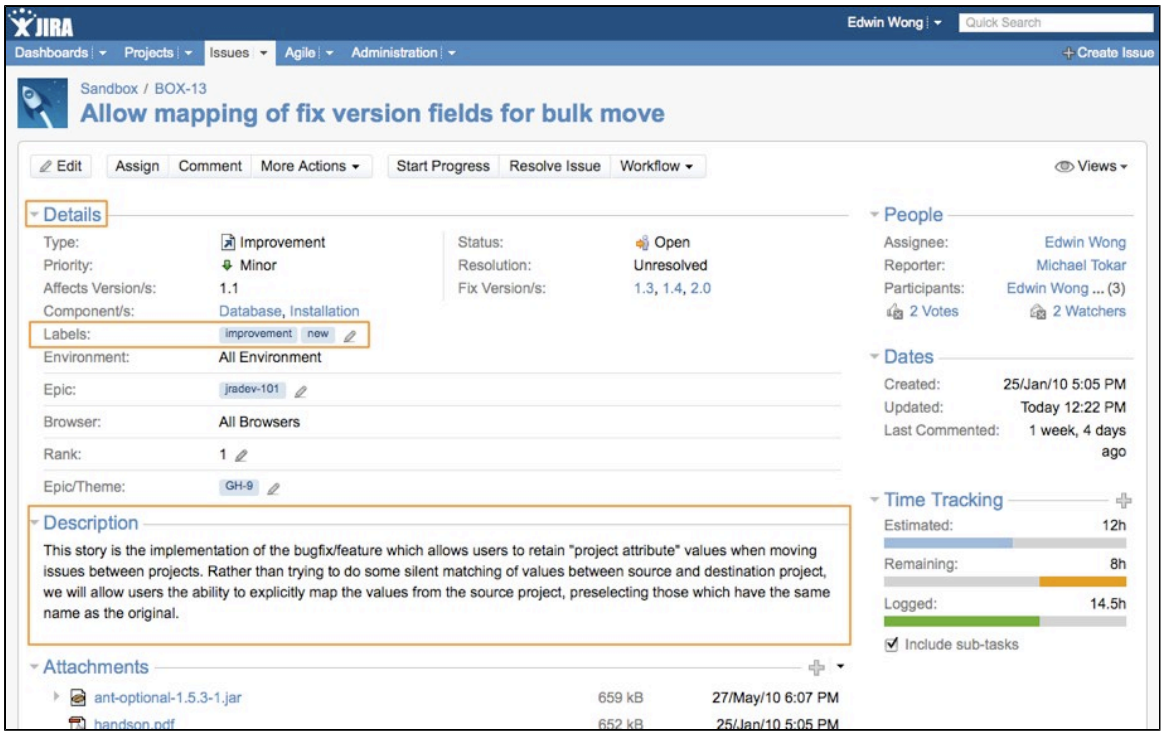


^Top

Improvements to the 'View Issue' Screen

We have improved the look and feel of the view issue screen even more. Some improvements include:

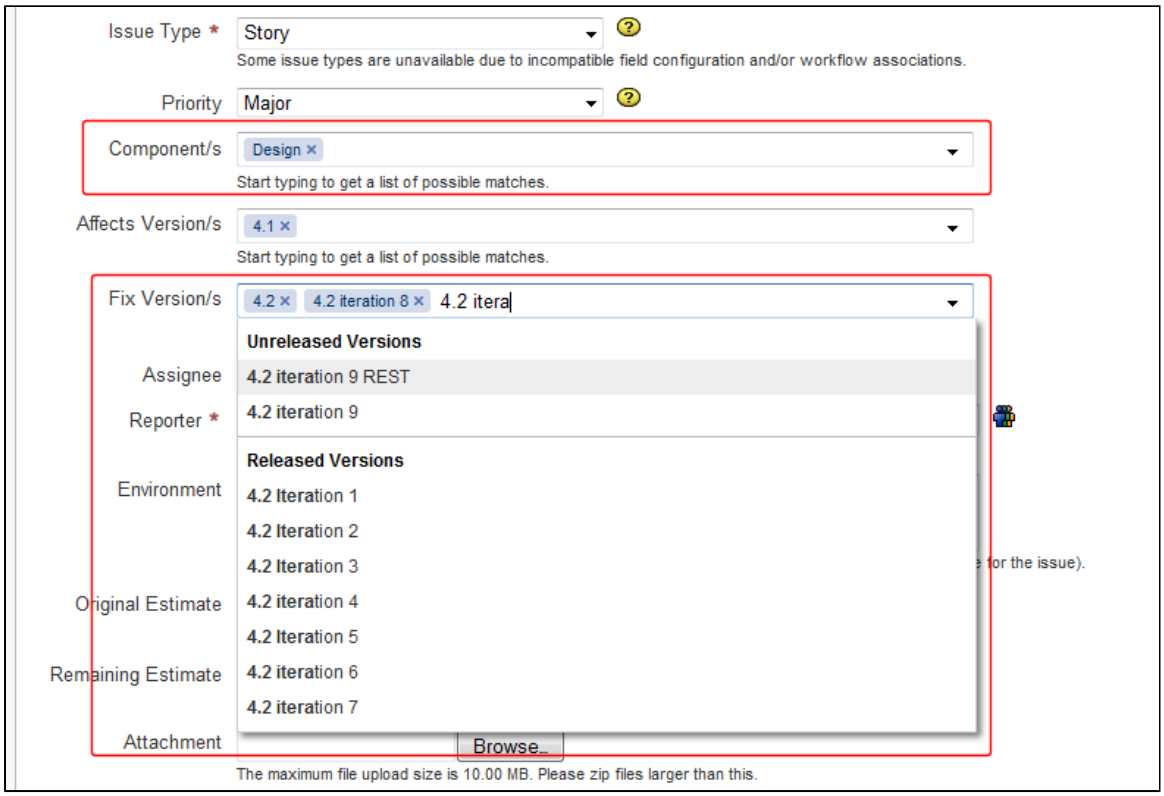
- A tighter spacing between lines to reduce space between elements
- Stronger section headings
- The description field separated into its own section
- Horizontal division lines between each custom field
- Votes and Watchers fields on the same line
- Labels represented as bubbles



Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues. After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the commenting section.

We've also updated the look and feel of the Components and Affects/Fix Versions lists in Edit mode. Upon typing into one of these fields, a dropdown menu appears with a list of options matching the first few characters you typed.

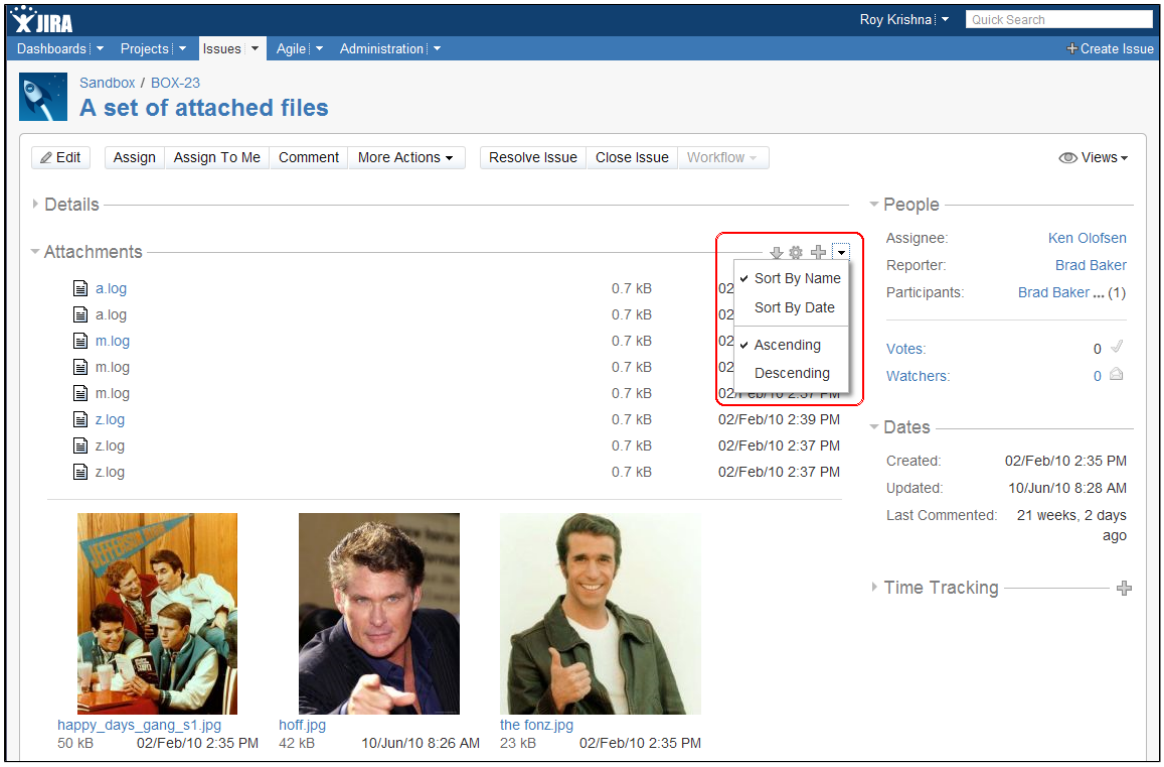
For convenience, the version lists are divided into **Released** and **Unreleased** categories.



^Top

Attachment Sorting by Date or Name

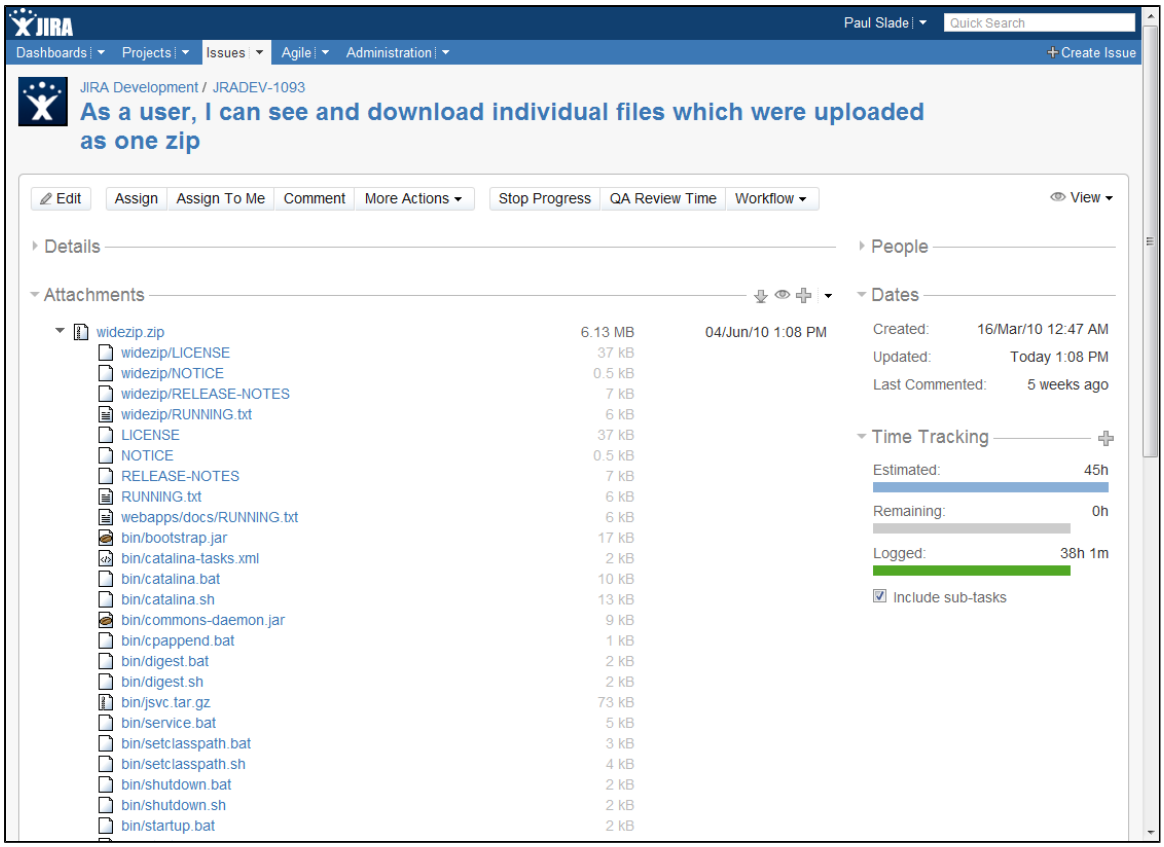
Attachments can now be sorted by date or by name, in ascending or descending order.



^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



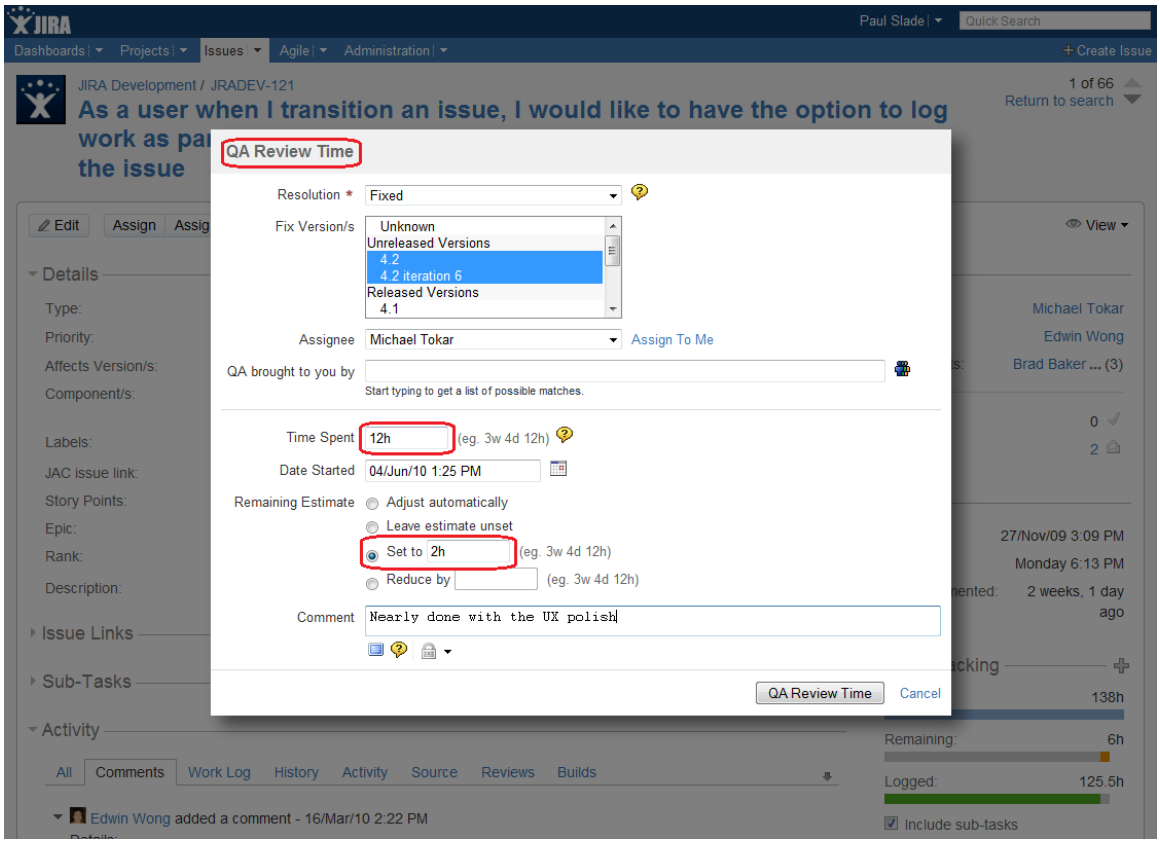
^Top

'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).



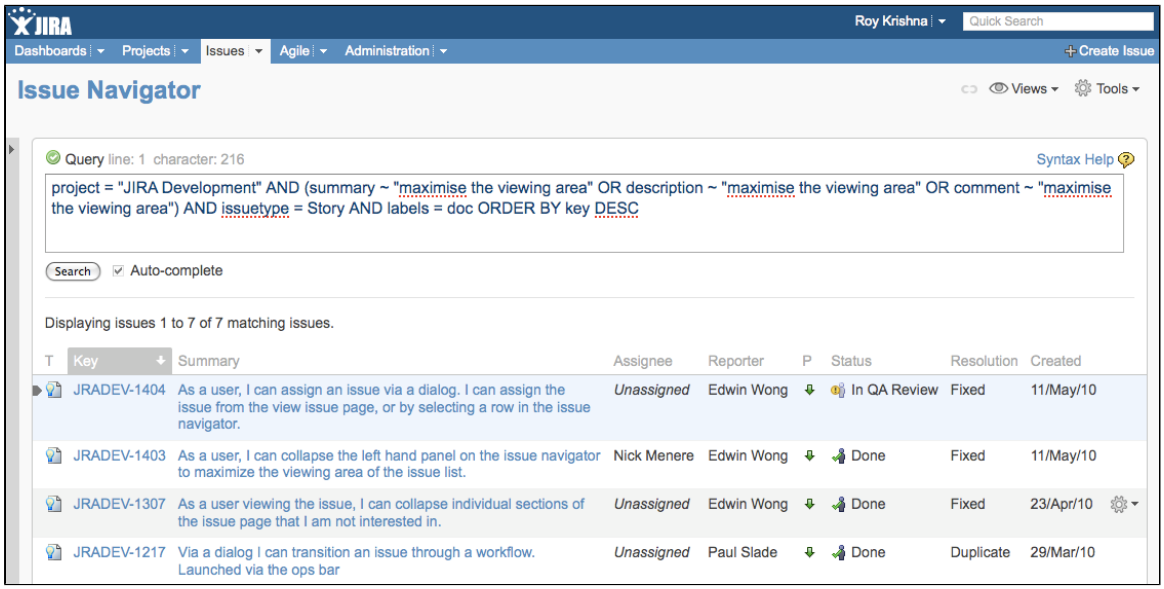
^Top

8

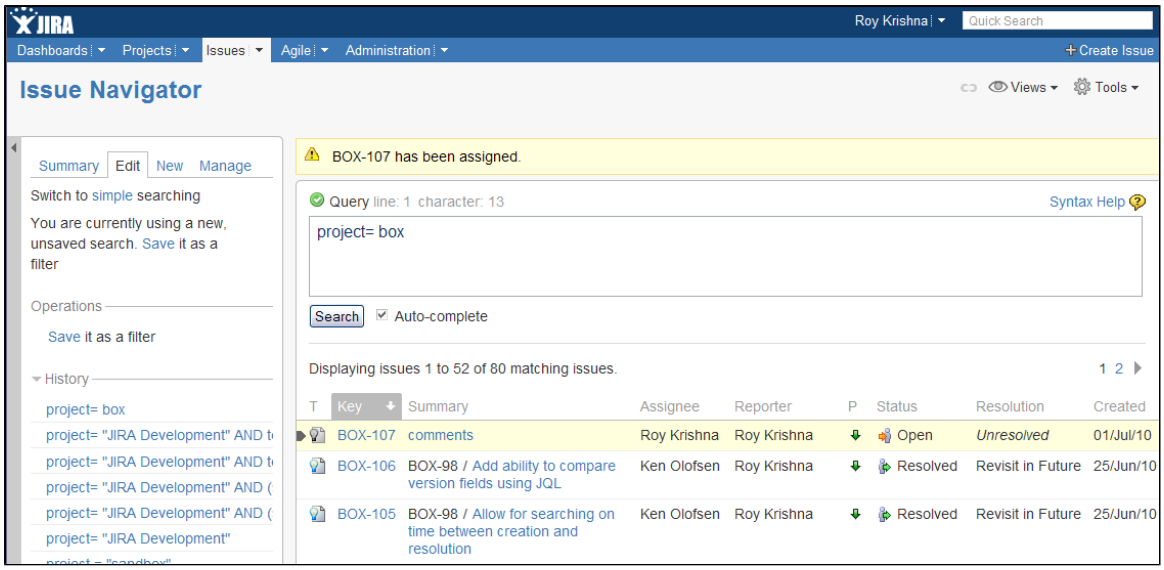
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



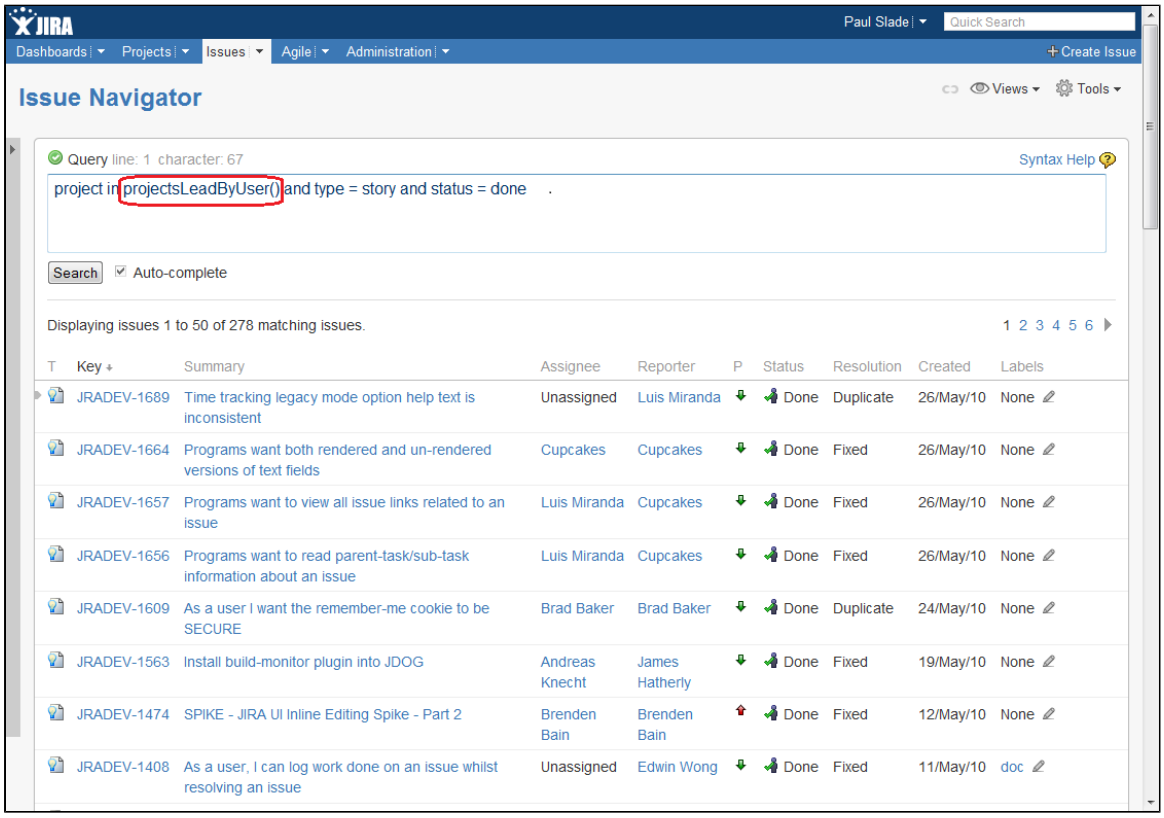
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

New JQL Functions

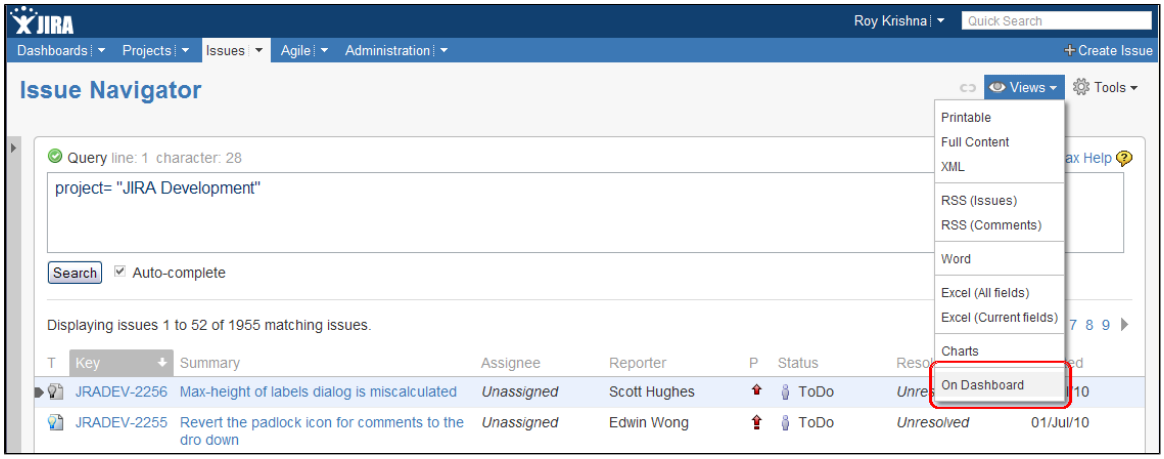
Additional JQL functions for projectLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



^Top

'Filter' Gadget creation via the Issue Navigator

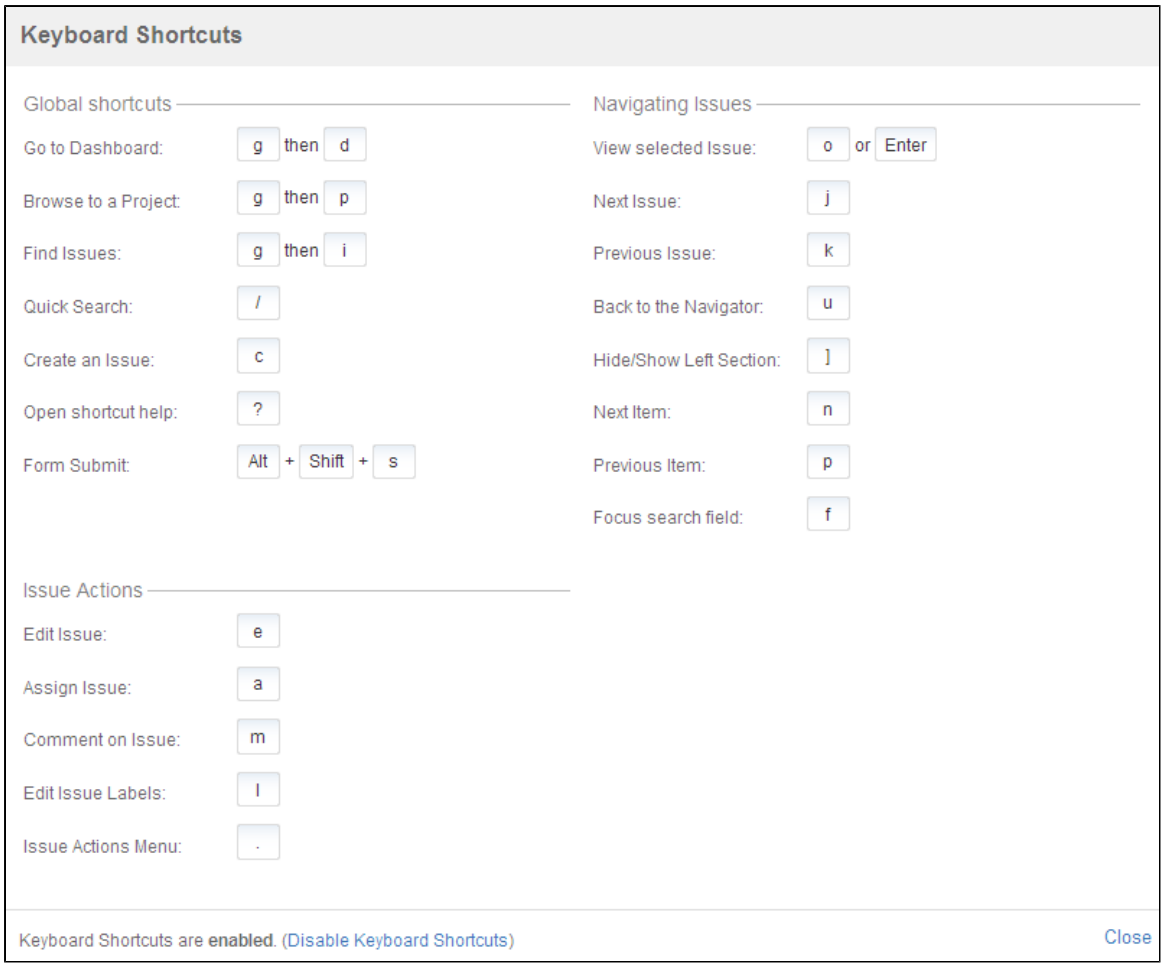
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



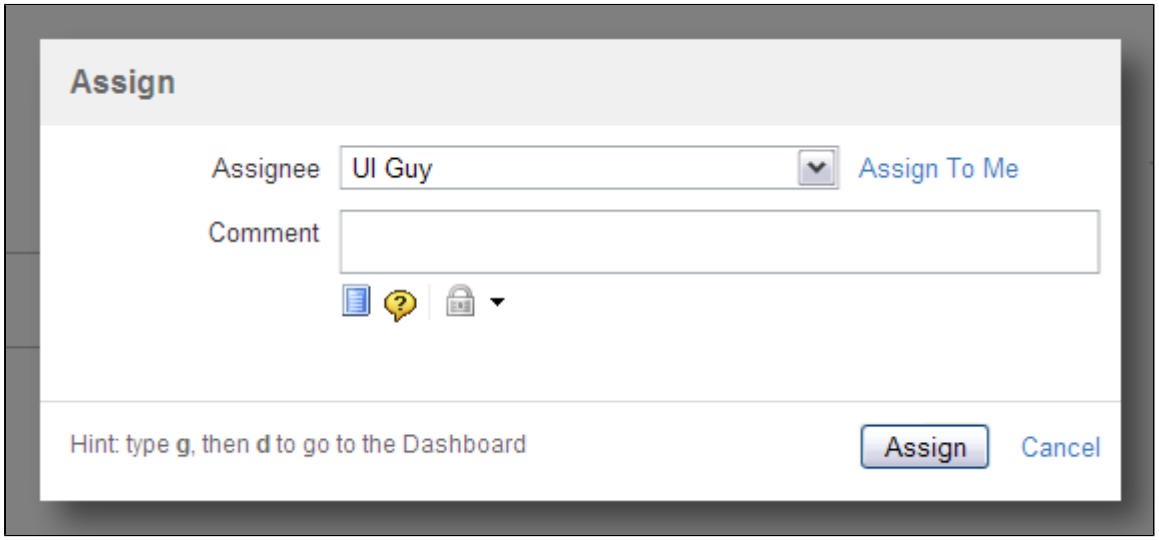
^Top

More Keyboard Shortcuts

JIRA users can now do more without a mouse --- perfect for those power users.



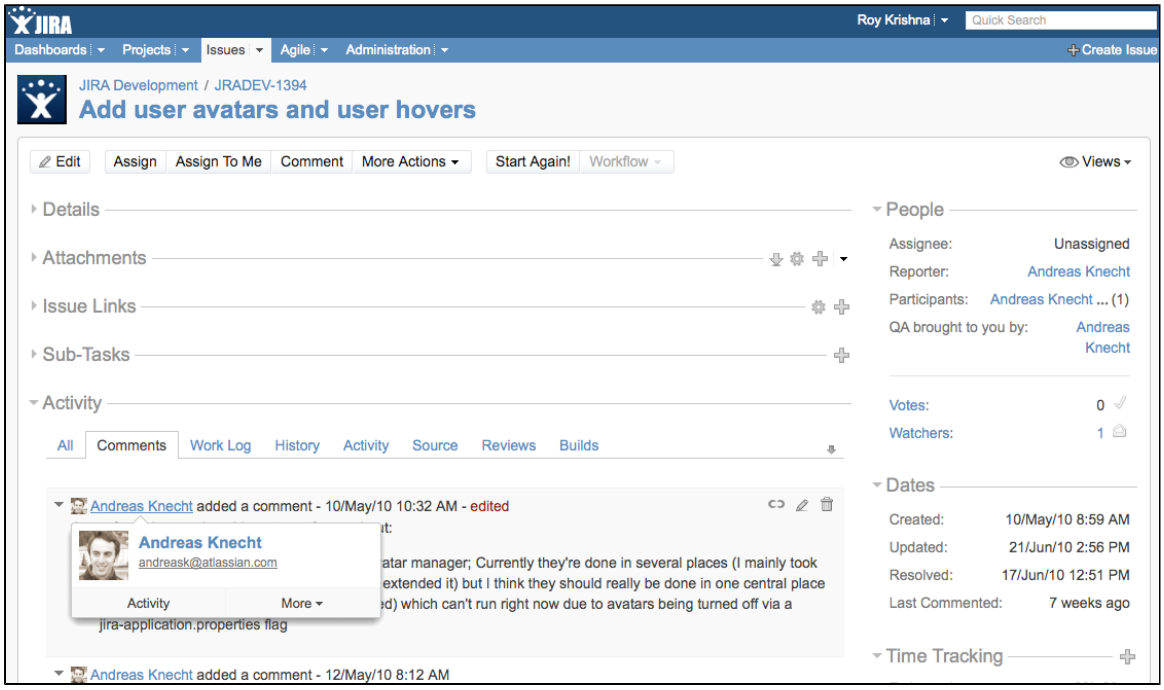
Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly.



^Top

User Avatars and User Hover

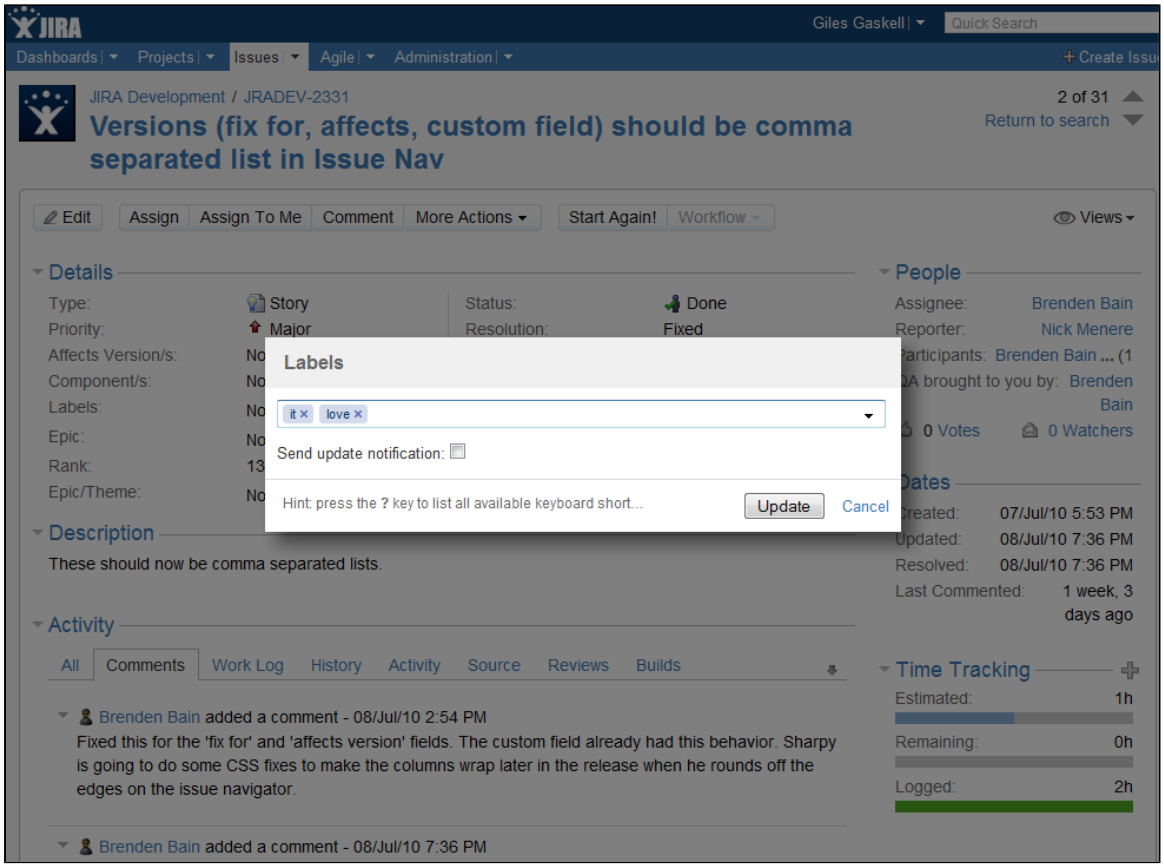
JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.



^Top

Labels Now Included

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.



^Top

Other Enhancements and Fixes

Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.2 EAP 2 Release Notes

6 July 2010

JIRA 4.2 EAP 2 (a.k.a 4.2 milestone 7) is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

GreenHopper Support

Please be aware that GreenHopper 5.0.1 is not compatible with the JIRA 4.2 EAP 2 release.

The Atlassian team is proud to bring you the JIRA 4.2 EAP 2 release. We still have a little more work to do, but overall, this is how 4.2 will look when it gets rolled out later this year. And we want your feedback.

Time-tracking is now much more flexible: you can edit the Original Estimate, and set the Remaining Estimate to zero on resolving an issue. You can also log work via workflow 'transition' screens — which are now provided as in-place dialogs rather than separate screens. For the mouse-averse, the new 'Operations Dialog' provides access to all menu options via the keyboard. The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users — and everyone else who likes to label their JIRA issues. Dashboard Publish/Subscribe with Confluence is also included.

Note to developers: JIRA 4.2 includes Atlassian Plugin Framework version 2.5.

Note to partners and translators: The preliminary translation diff files will be made available when we release JIRA 4.2 beta in the coming weeks.

Highlights of JIRA 4.2 EAP 2:

- Dialogs for Common Actions and Workflow Operations
- Dialogs in the Issue Navigator
- 'Operations Dialog' Replaces the Mouse
- Improvements to the 'View Issue' Screen
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- New JQL Functions

- 'Filter' Gadget creation via the Issue Navigator
- More Keyboard Shortcuts
- User Avatars and User Hover
- Labels Now Included
- Other Enhancements and Fixes

What's New in JIRA 4.2 EAP 2

1. Components Look and Feel update.
2. Colour highlight for keyboard navigation (blue).
3. Feedback on changes made via dialogs in the issue navigator as confirmation.
4. Ability to add work log descriptions in wiki-markup (no screenshot).

Thank you for your interest in JIRA 4.2 EAP 2

Download EAP

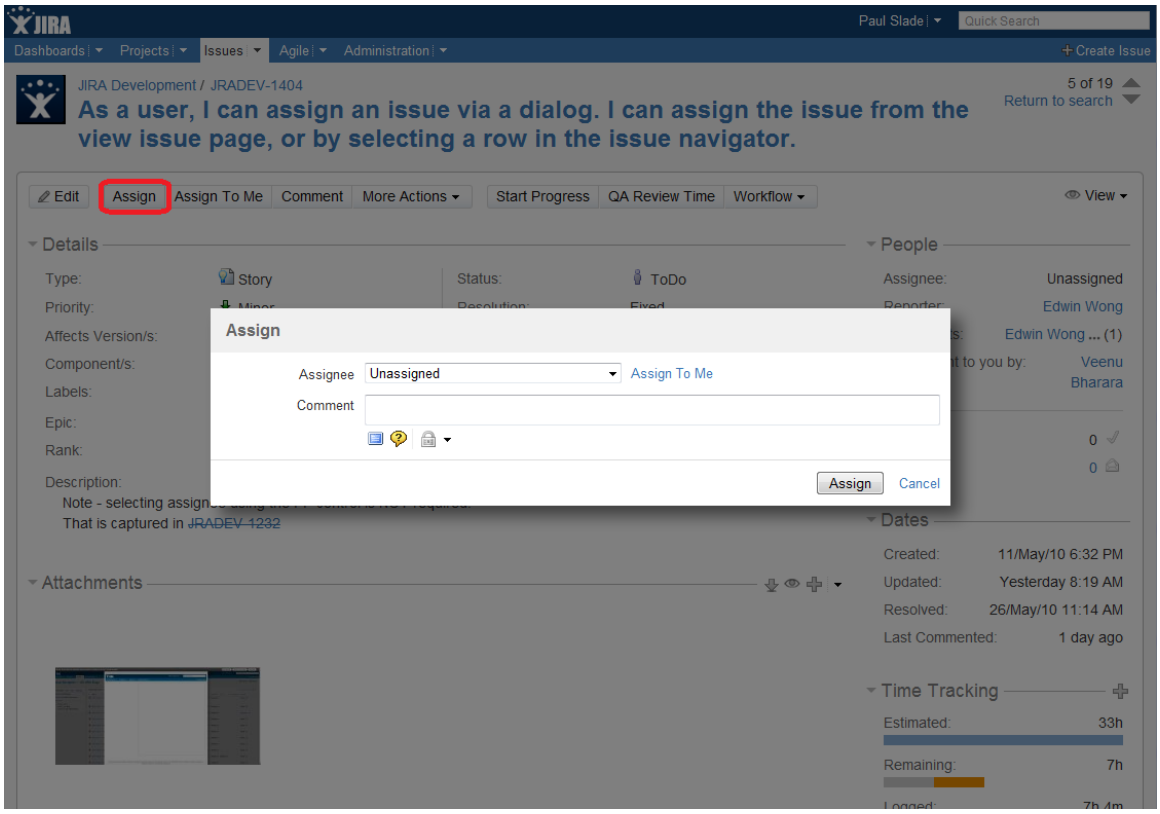
Upgrading to JIRA 4.2 EAP 2

JIRA EAP releases are available here. To upgrade from JIRA 4.1.x to this release, please follow the normal JIRA Upgrade Guide.

Highlights of JIRA 4.2 EAP 2

Dialogs for Common Actions and Workflow Operations

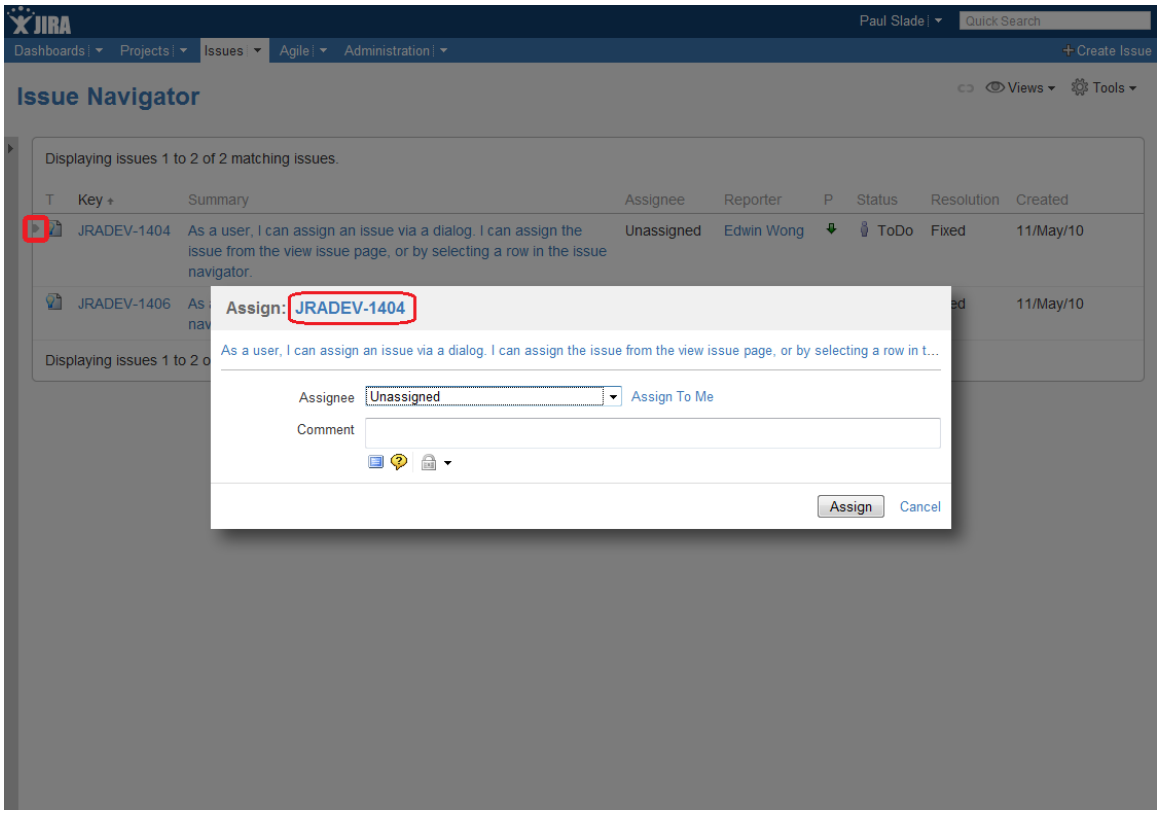
For faster edits to an issue, pop-up dialogs have replaced screen changes:



^Top

Dialogs in the Issue Navigator

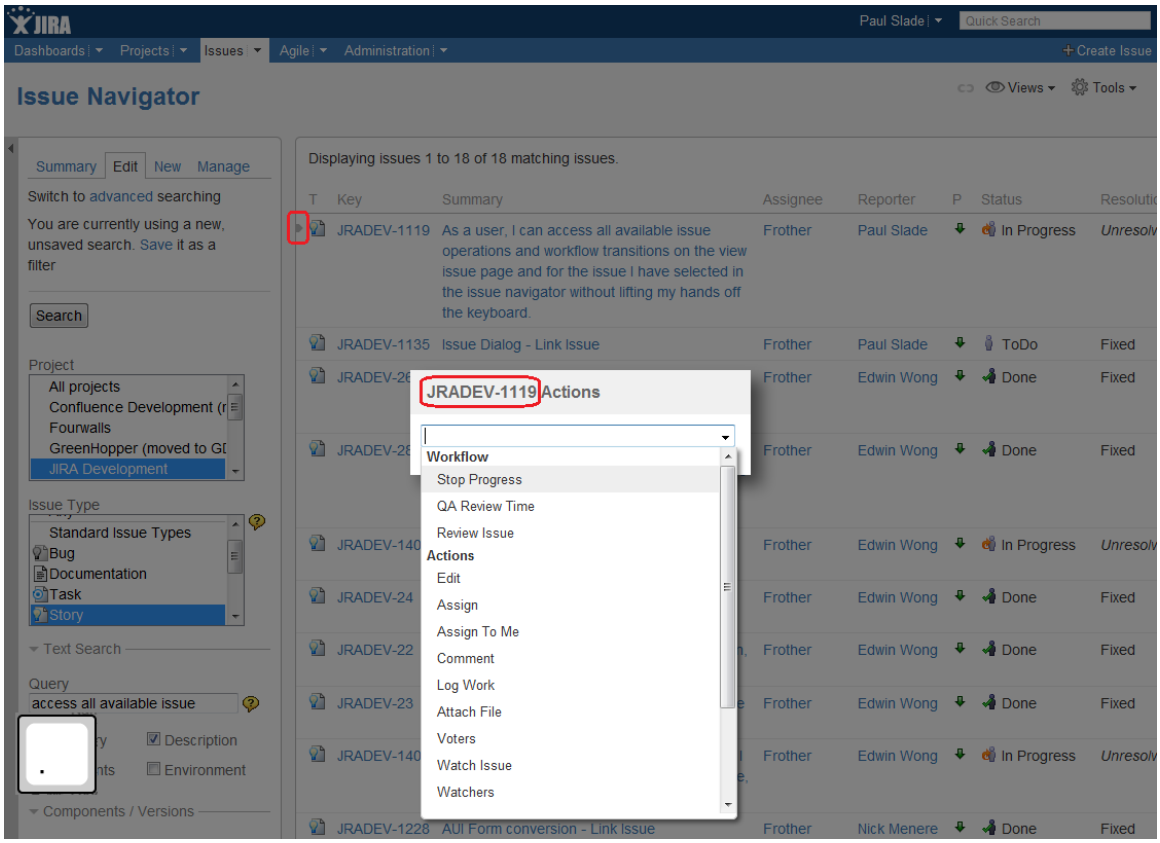
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



^Top

'Operations Dialog' Replaces the Mouse

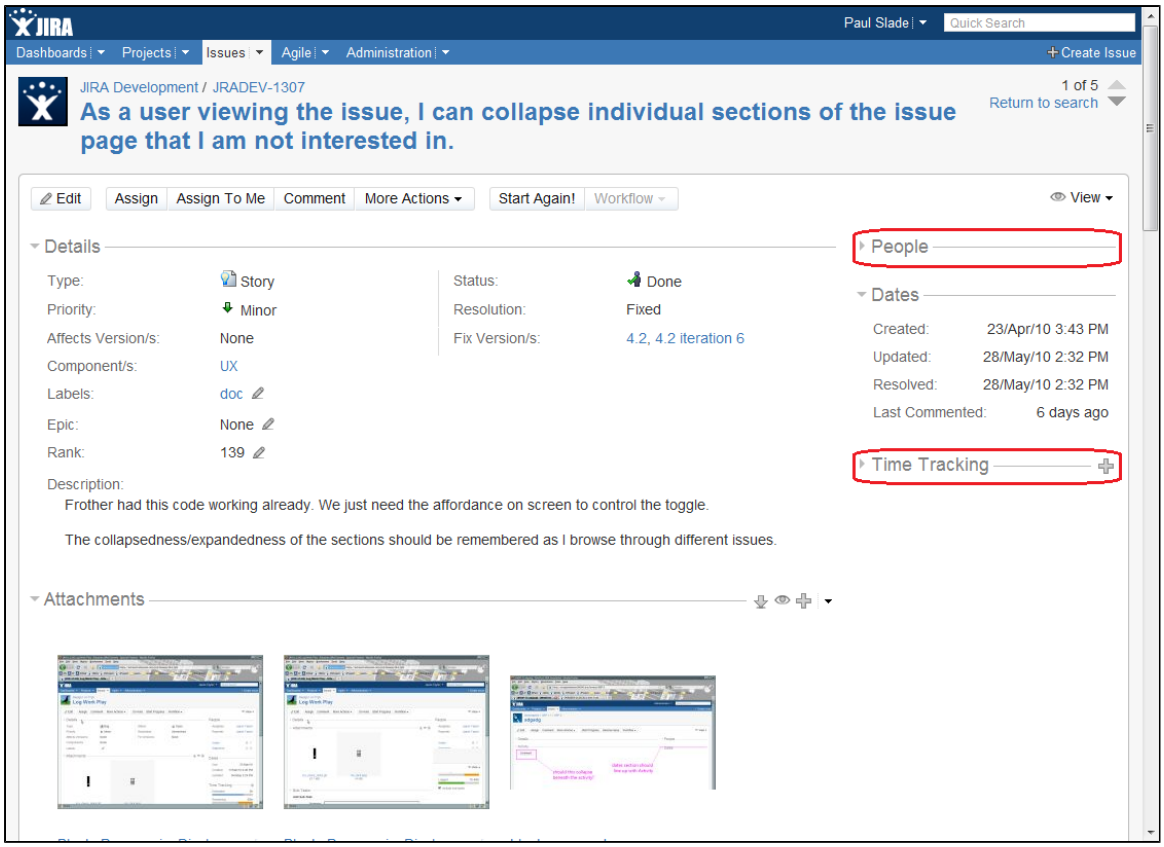
The new 'Operations Dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The Operations Dialog works from the Issue Navigator and also when viewing an individual issue.



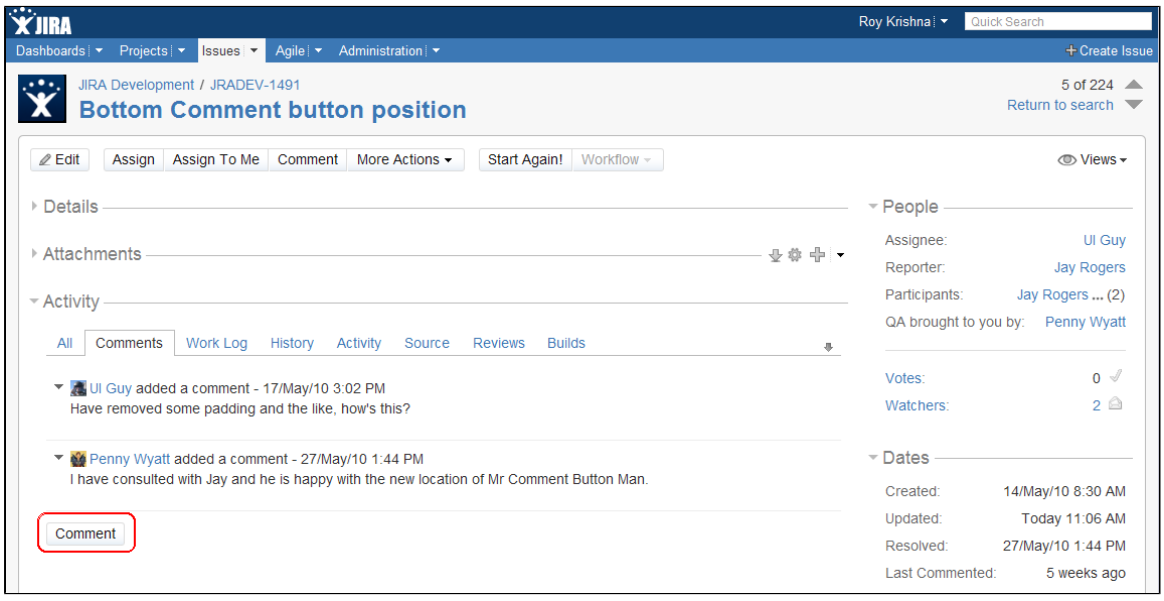
^Top

Improvements to the 'View Issue' Screen

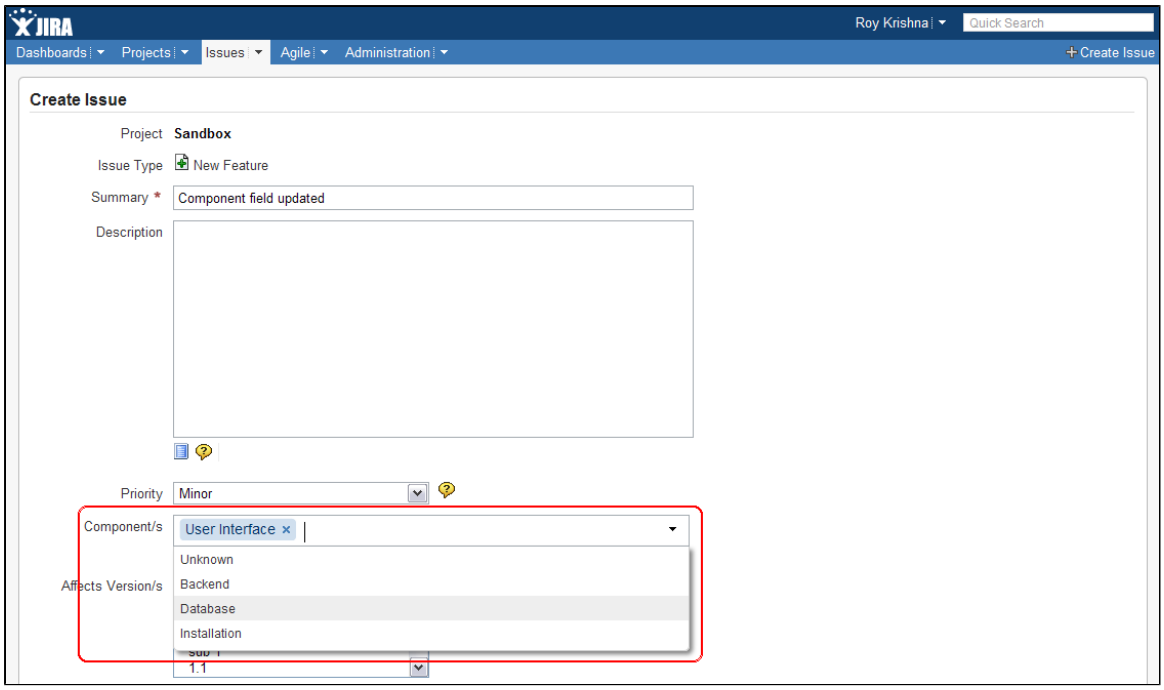
Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues.



After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the commenting section.



We've also managed to update the look and feel of the component list in Edit mode.

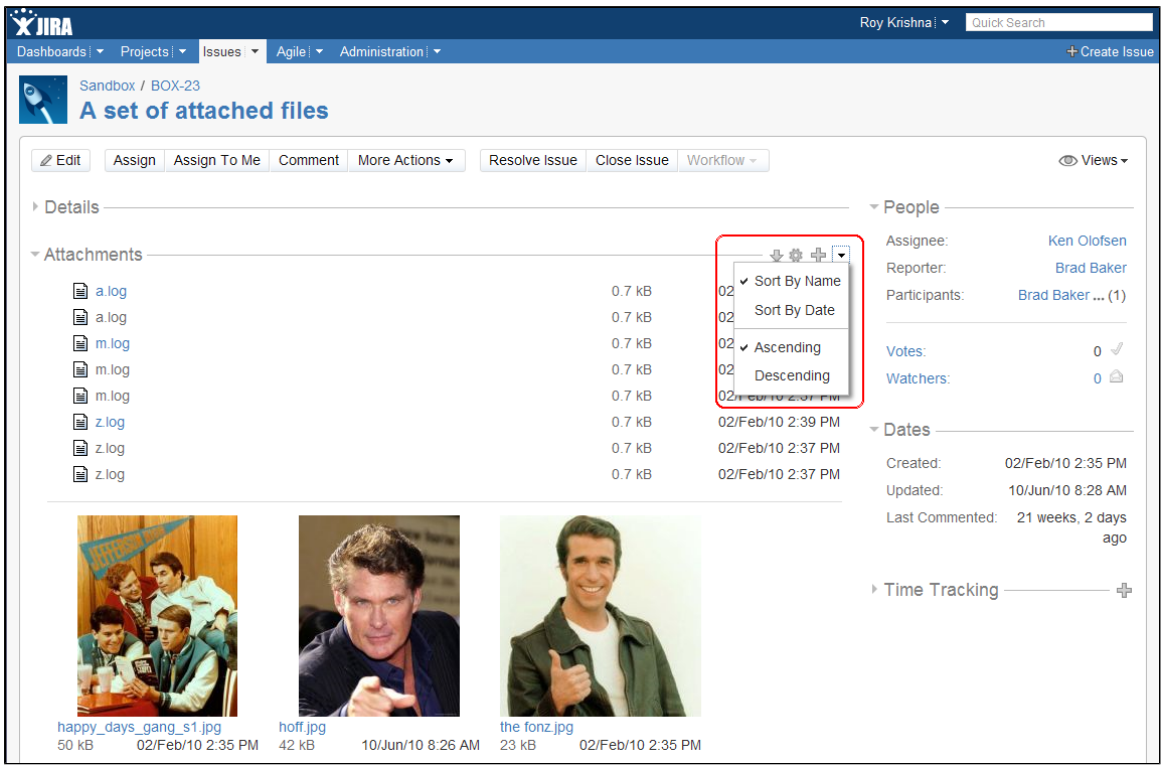


^Top

5

Attachment Sorting by Date or Name

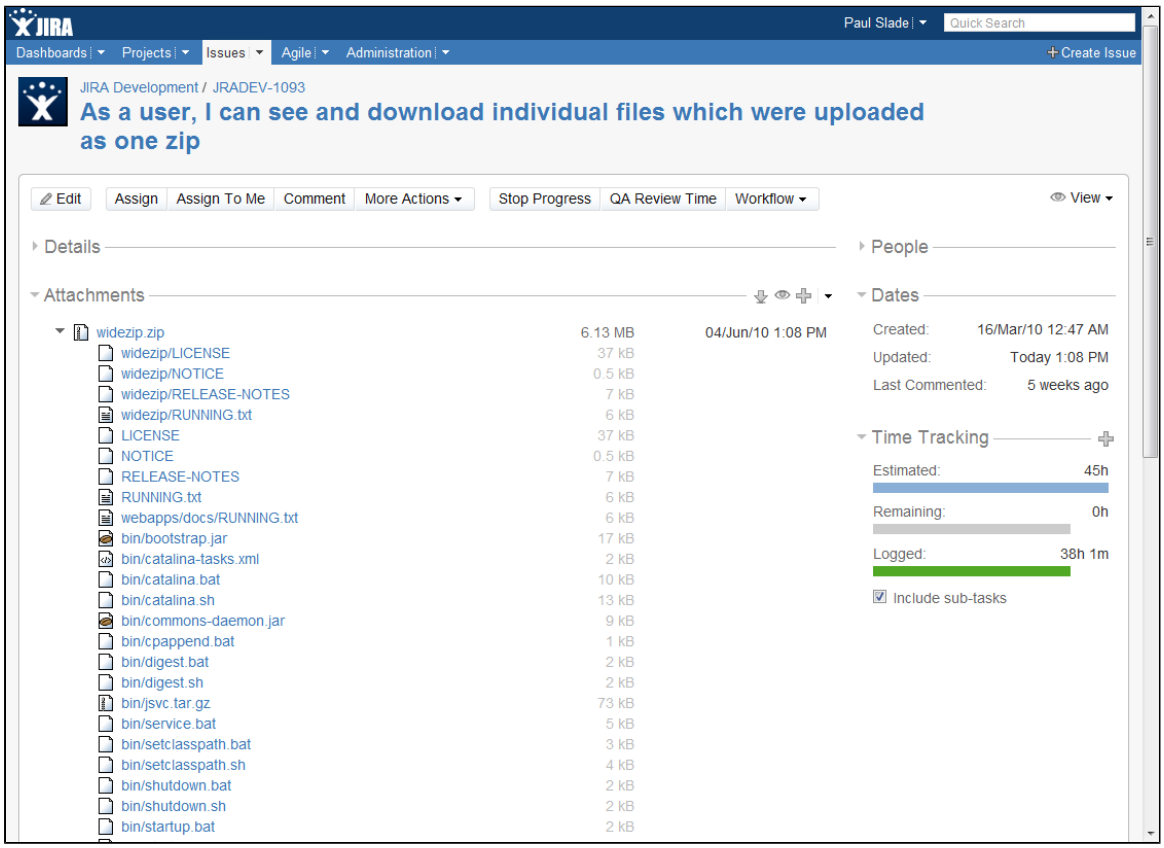
Attachments can now be sorted by date or by name, in ascending or descending order.



^Top

Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.



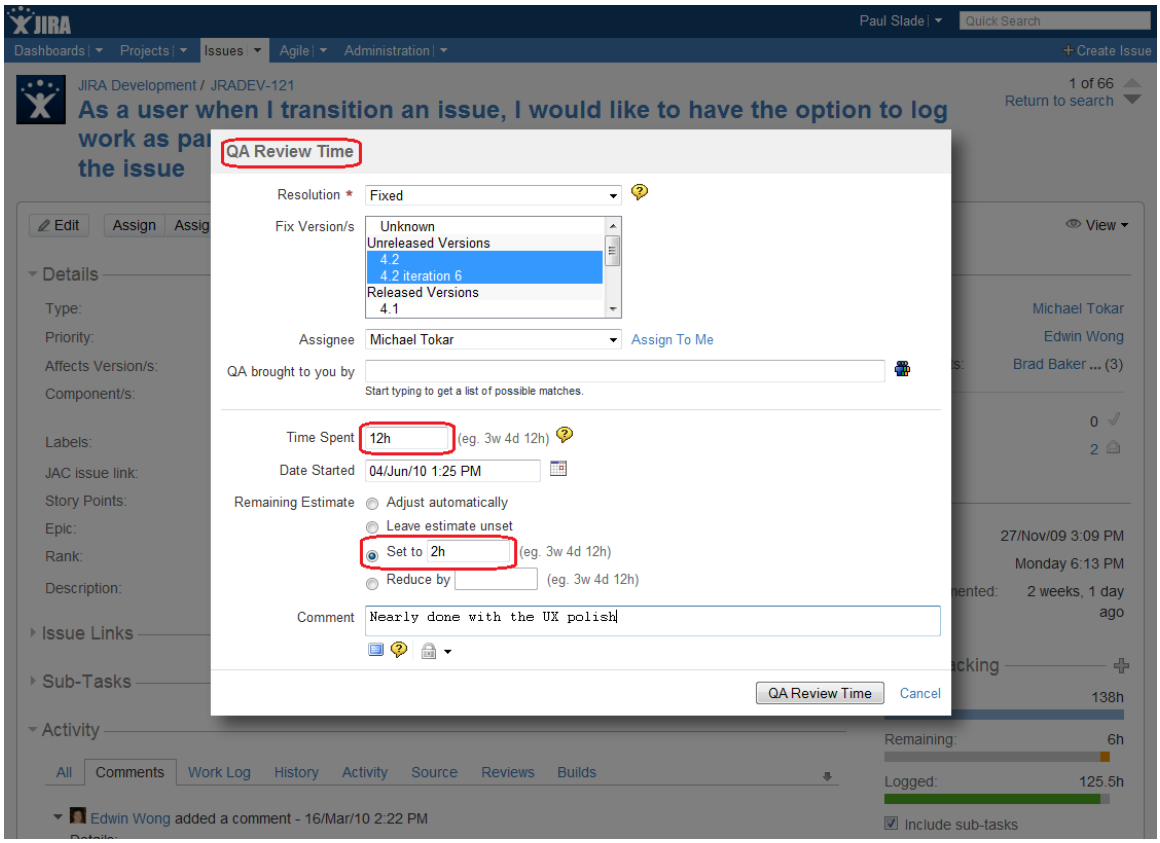
^Top

'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the Remaining Estimate to 0 upon resolving an issue.

You now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

We've also included the ability to add work log descriptions in wiki-markup (not shown here).



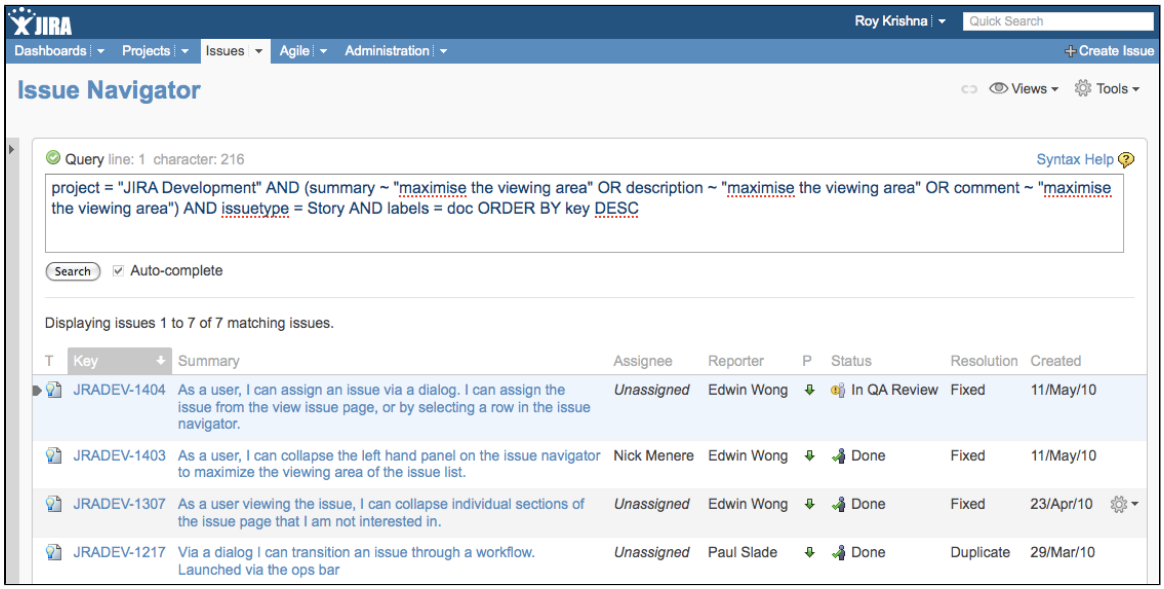
^Top

8

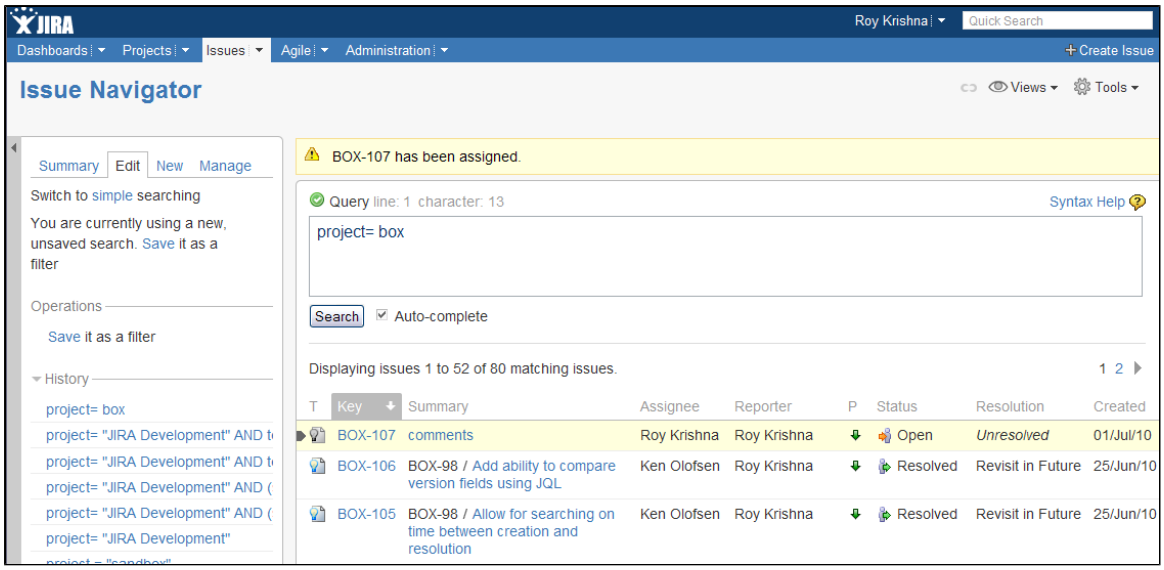
Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue.

We've also added: an issue marker and colour highlight for keyboard navigation (blue); colour highlight for mouse selection (grey); toggle to collapse the search form.



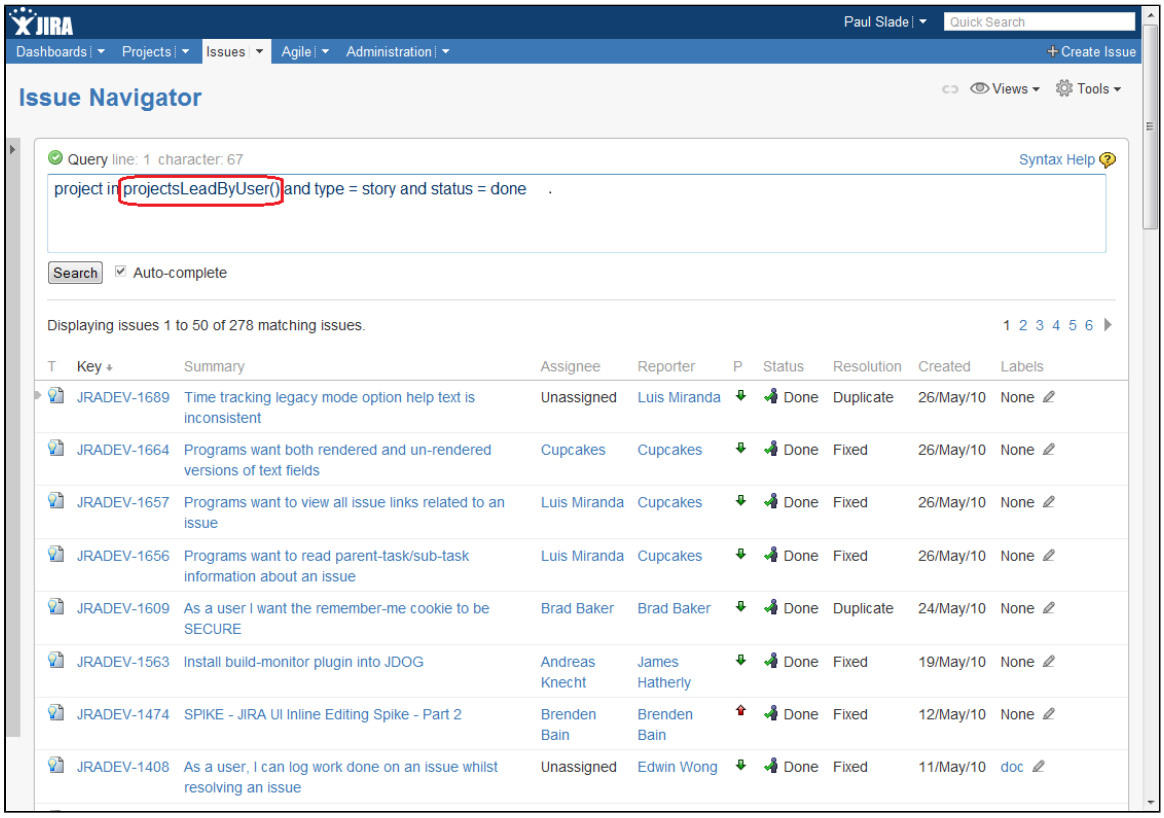
When a change is made via a dialog, JIRA will give you feedback on that change as confirmation.



^Top

New JQL Functions

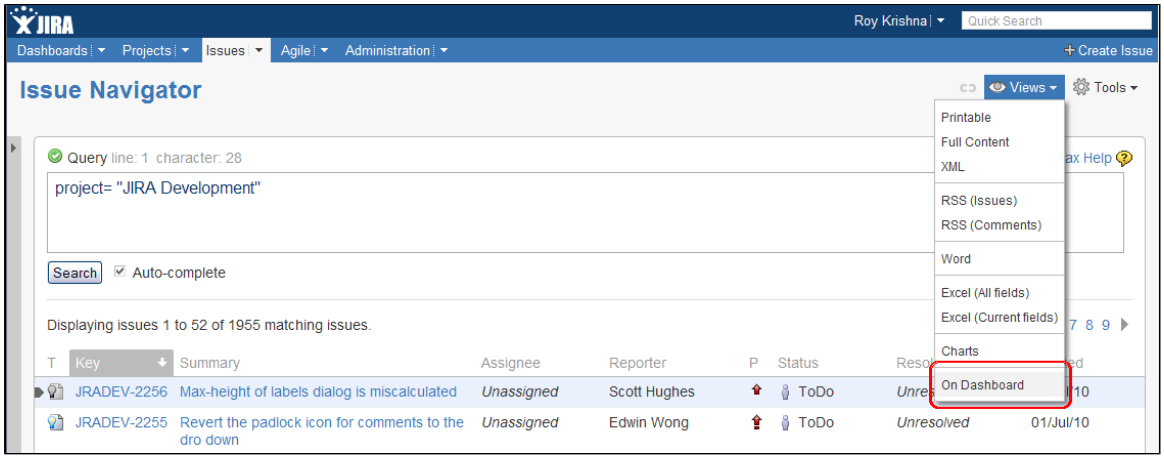
Additional JQL functions for projectLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



^Top

'Filter' Gadget creation via the Issue Navigator

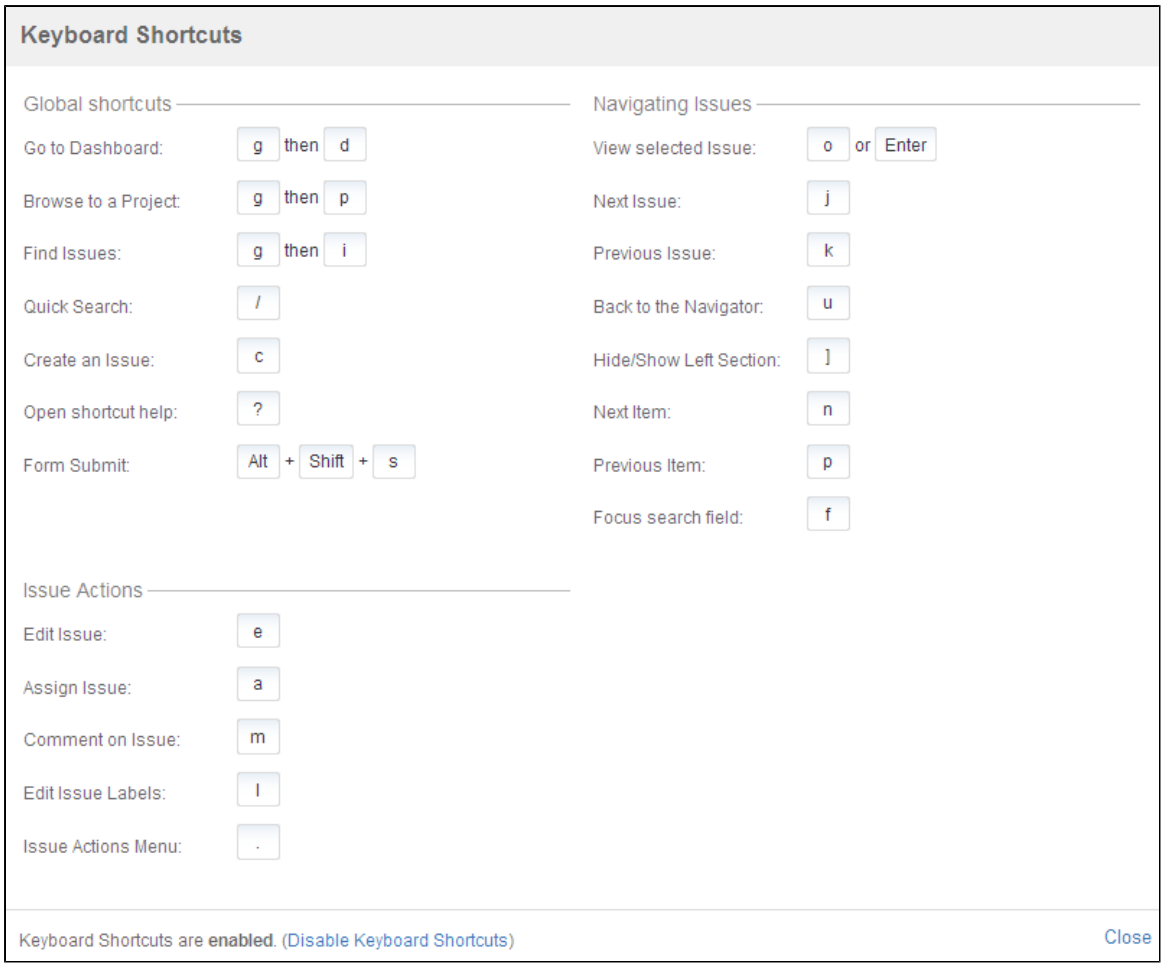
When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'Views' menu.



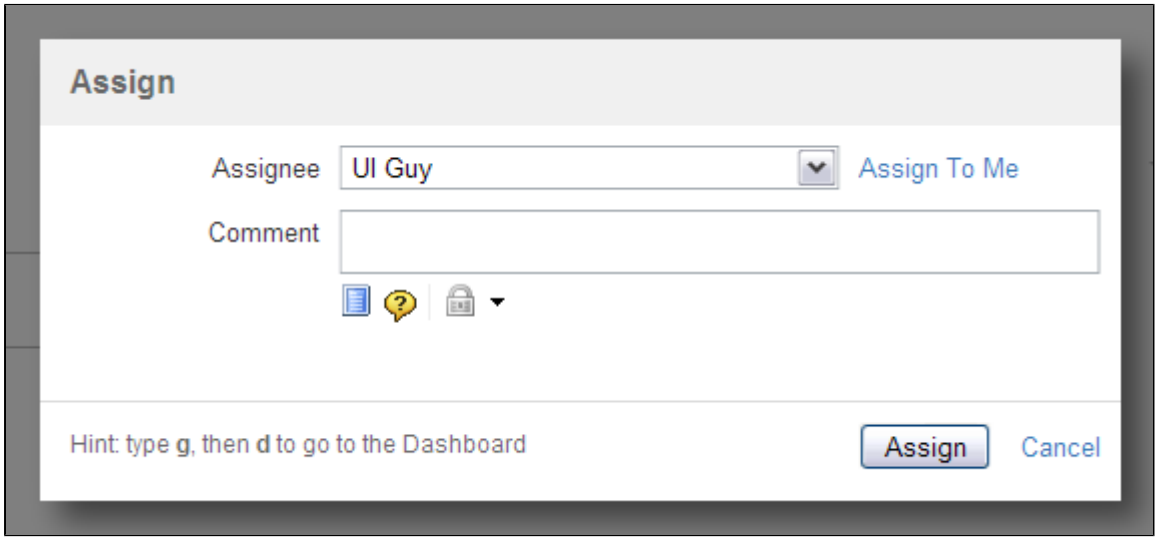
^Top

More Keyboard Shortcuts

JIRA users can now do more without a mouse --- perfect for those power users.



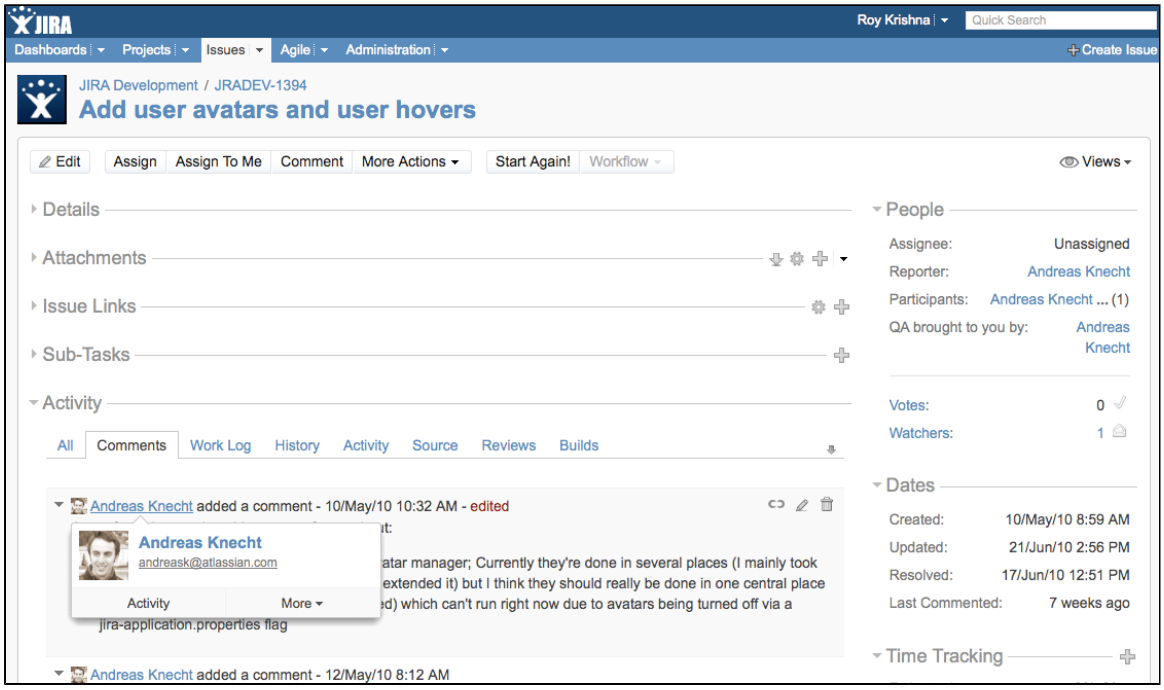
Hints also appear at the bottom of a dialog, helping you to learn keyboard shortcuts on the fly.



^Top

User Avatars and User Hover

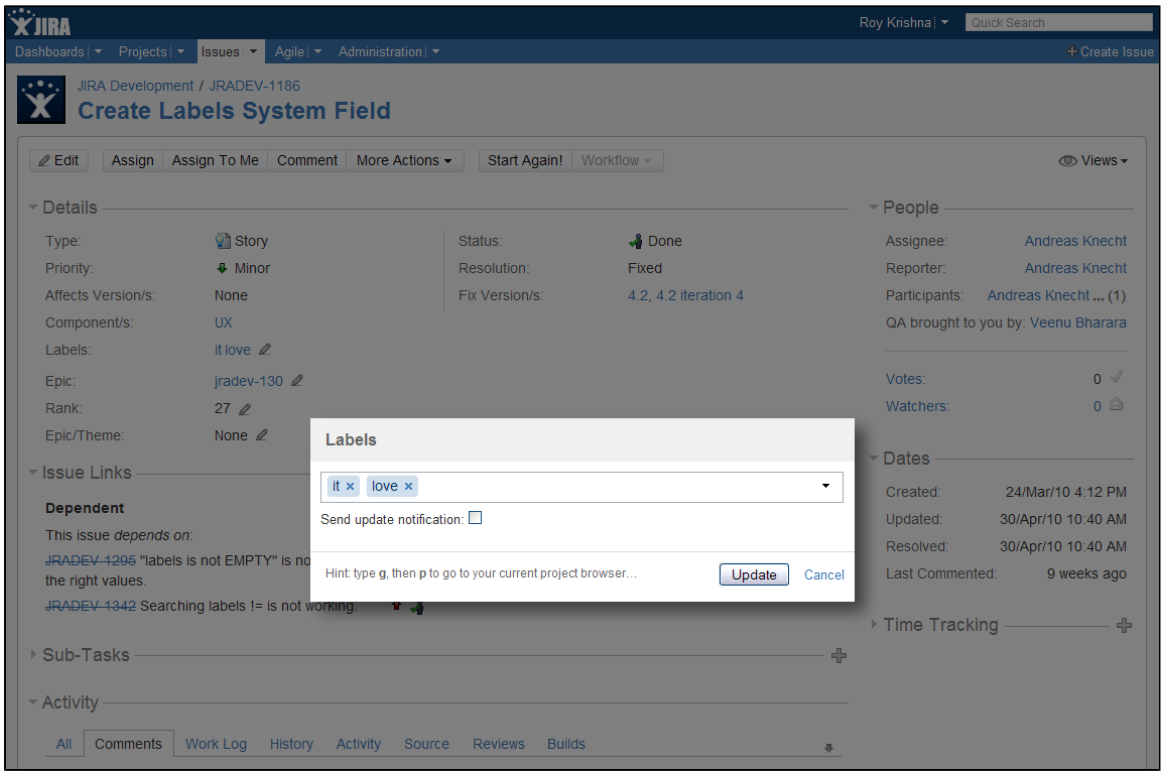
JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.



^Top

Labels Now Included

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI, and issue notifications are now optional when updating labels for an issue.



^Top

Other Enhancements and Fixes

Click here for the full list of issues resolved in 4.2.

^Top

JIRA 4.2 EAP 1 Release Notes

8 June 2010

JIRA 4.2 EAP 1 (a.k.a 4.2 milestone 6) is a public development release leading up to **JIRA 4.2**. For all production use and testing of JIRA, please use the latest official release.

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

GreenHopper Support

Please be aware that GreenHopper 5.0 is not compatible with the JIRA 4.2 EAP releases.

The Atlassian team is proud to bring you the JIRA 4.2 EAP 1 release. We still have a little more work to do, but overall, this is how 4.2 will look when it gets rolled out later this year. And we want your feedback.

Time-tracking is now much more flexible: you can edit the Original Estimate, and set the Remaining Estimate to zero on resolving an issue. You can also log work via workflow 'transition' screens — which are now provided as in-place dialogs rather than separate screens. For the mouse-averse, the new 'dot dialog' provides access to all menu options via the keyboard. The 'Labels' plugin is now a permanent part of JIRA, simplifying system management for GreenHopper users — and everyone else who likes to label their JIRA issues. Dashboard Publish/Subscribe with Confluence is also included.

Note to developers: JIRA 4.2 includes Atlassian Plugin Framework version 2.5.

Note to partners and translators: The preliminary translation diff files will be made available when we release JIRA 4.2 beta in the coming weeks.

Highlights of JIRA 4.2 EAP:

- Dialogs for Common Actions and Workflow Operations
- Dialogs in the Issue Navigator
- 'Dot Dialog' Replaces the Mouse
- Improvements to the 'View Issue' Screen
- Attachment Sorting by Date or Name
- Viewable Files in Zipped Attachments
- 'Log Work' Improvements
- Improvements to the Issue Navigator
- Separate Searching in Multiple Tabs
- New JQL Functions
- User Avatars and User Hover
- 'Filter' Gadget creation via the Issue Navigator
- Labels Now Included
- Other Enhancements and Fixes

Thank you for your interest in JIRA 4.2 EAP 1
Download EAP

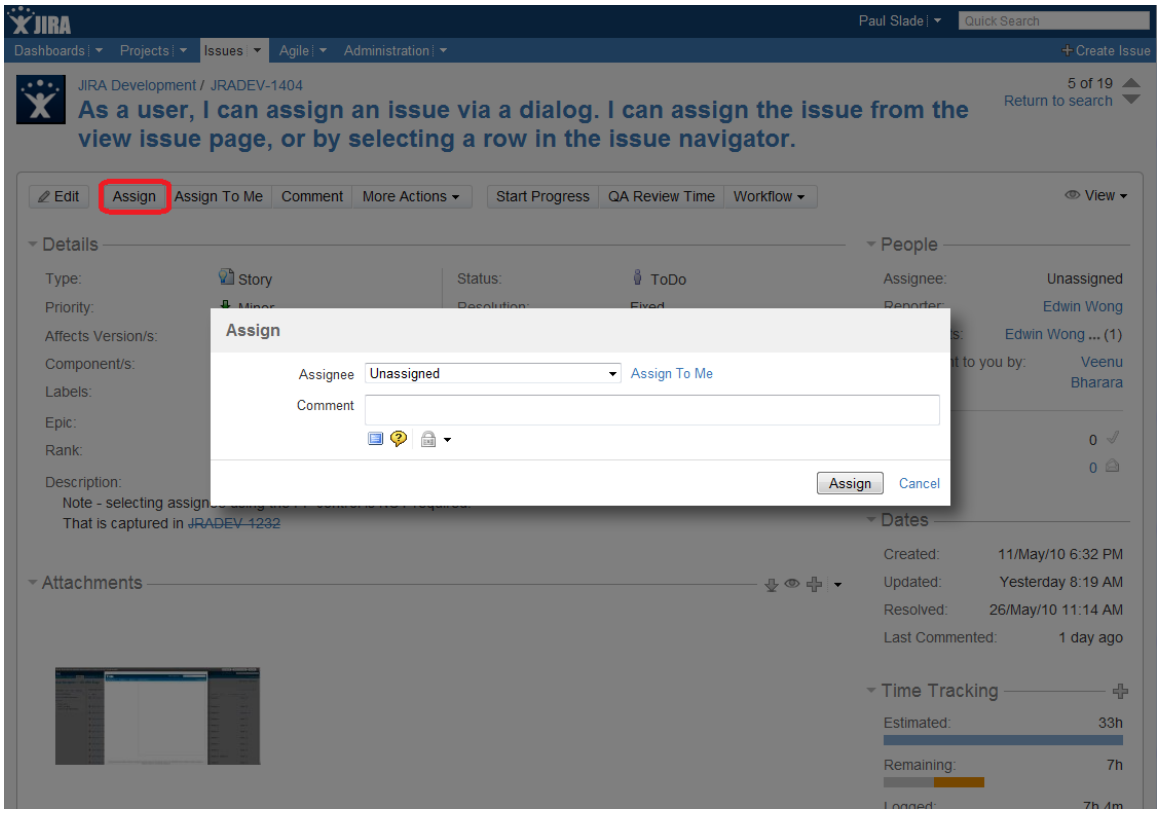
> **Upgrading to JIRA 4.2 EAP 1**
>
> JIRA EAP releases are available here. To upgrade from JIRA 4.1.x to this release, please follow the normal JIRA Upgrade Guide.

Highlights of JIRA 4.2 EAP 1

1

Dialogs for Common Actions and Workflow Operations

For faster edits to an issue, pop-up dialogs have replaced screen changes:

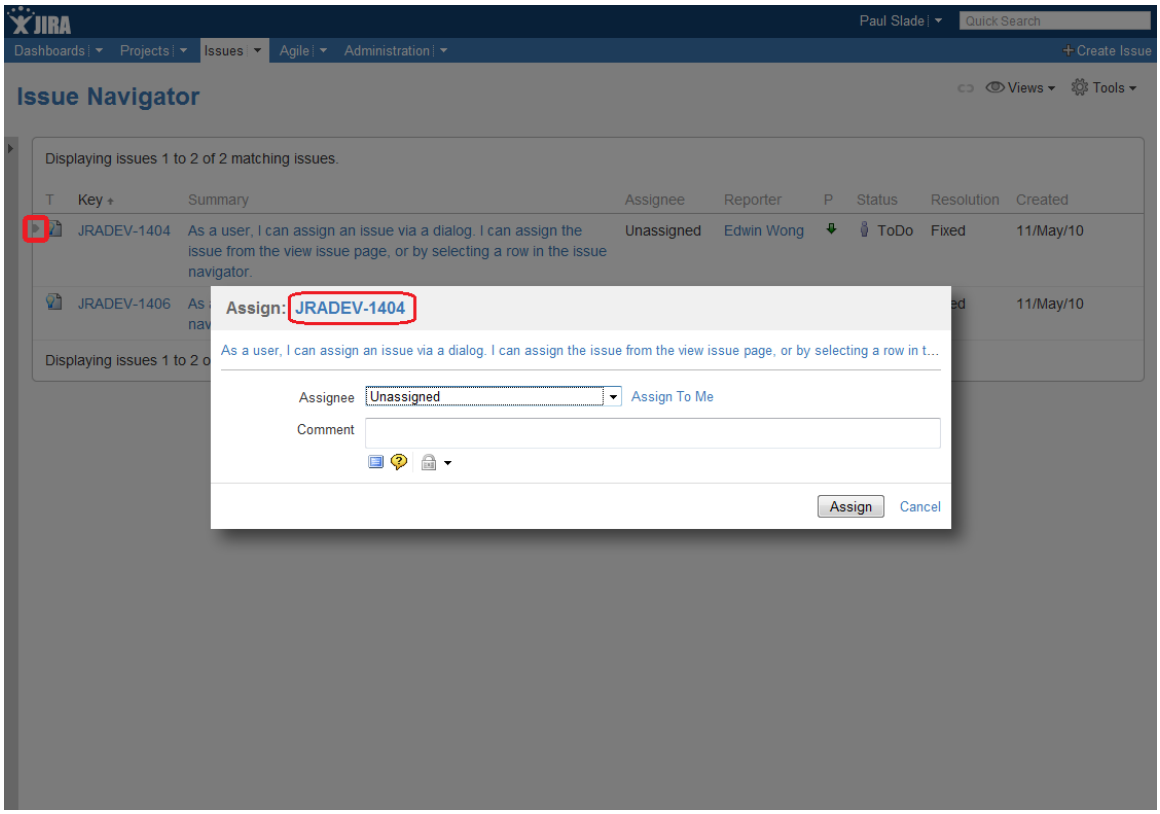


^Top

2

Dialogs in the Issue Navigator

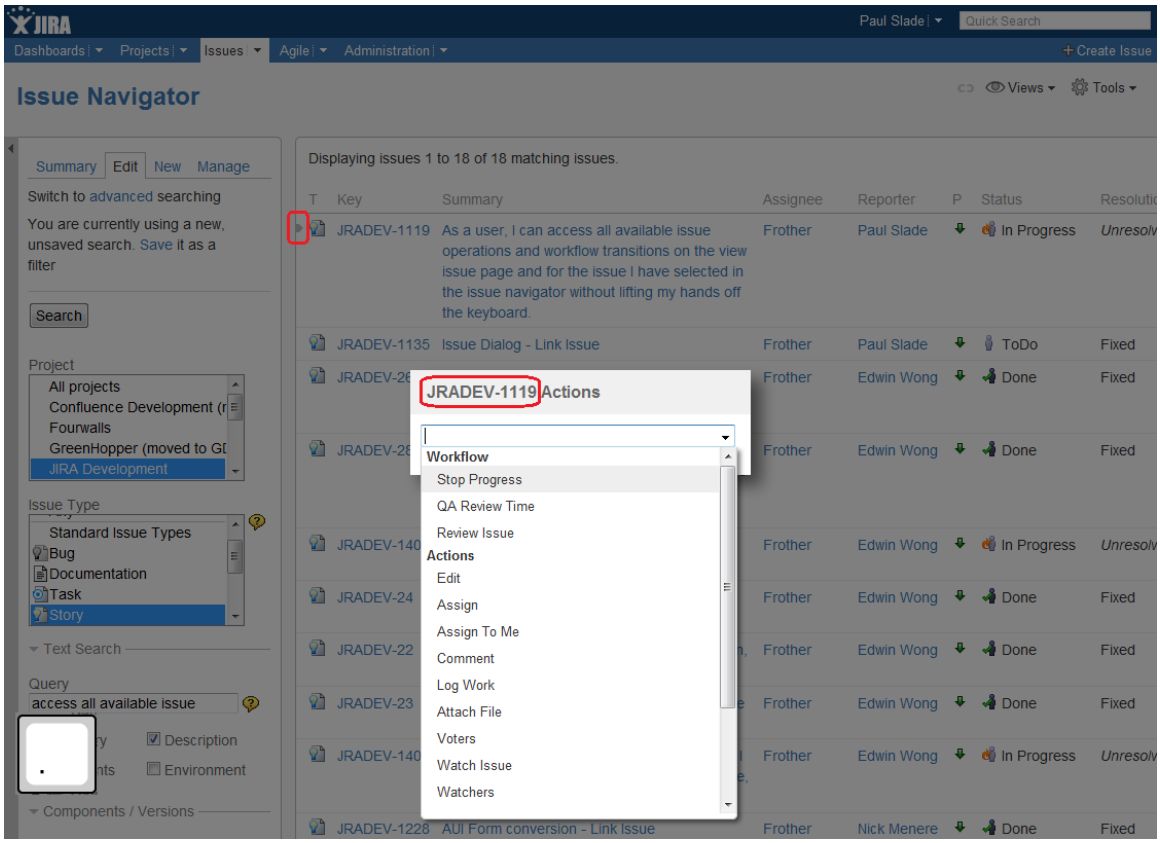
You can now perform an action on an issue, via a dialog, directly from the Issue Navigator — without opening the issue first. The issue marker shows which issue you have selected (via your keyboard) in the result list.



^Top

'Dot Dialog' Replaces the Mouse

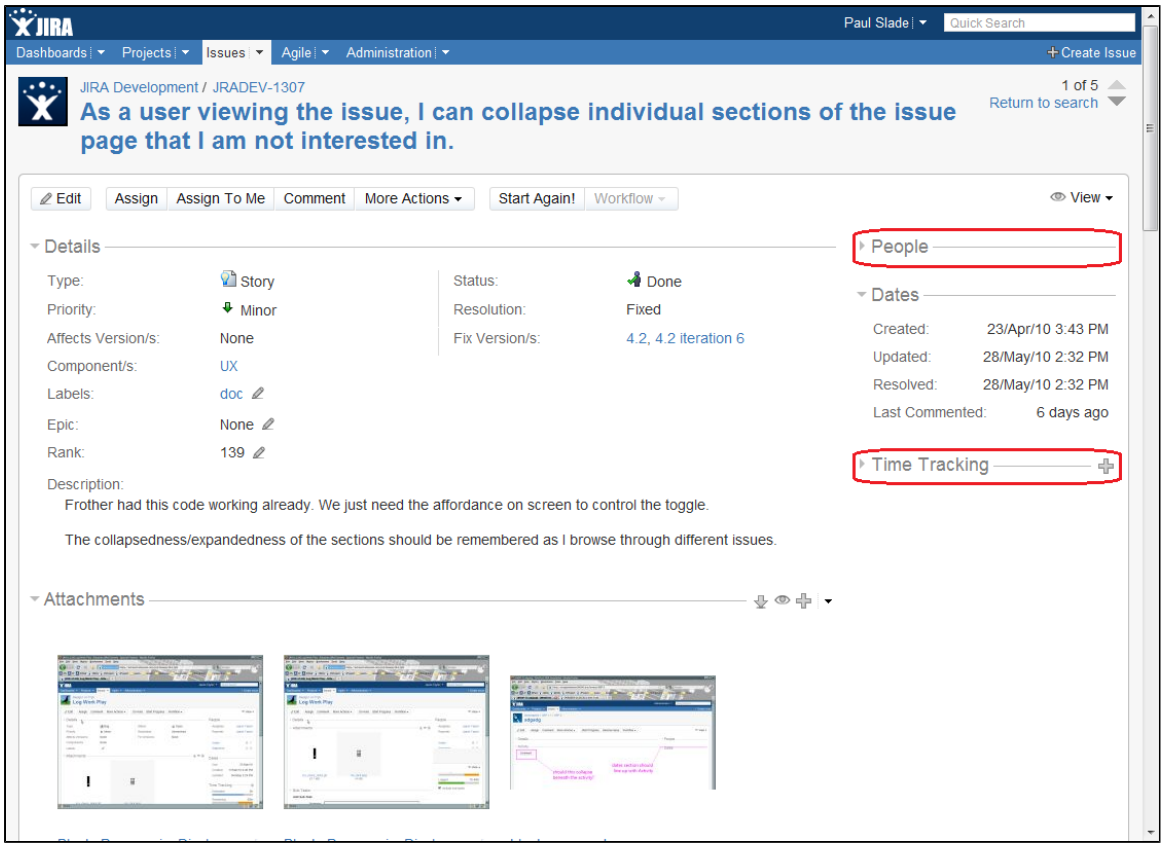
The new 'dot dialog' lets you perform actions via the keyboard (instead of the mouse), using the full-stop ('dot') key to access the 'Actions' and 'Workflow' menus. The dot dialog works from the Issue Navigator and also when viewing an individual issue.



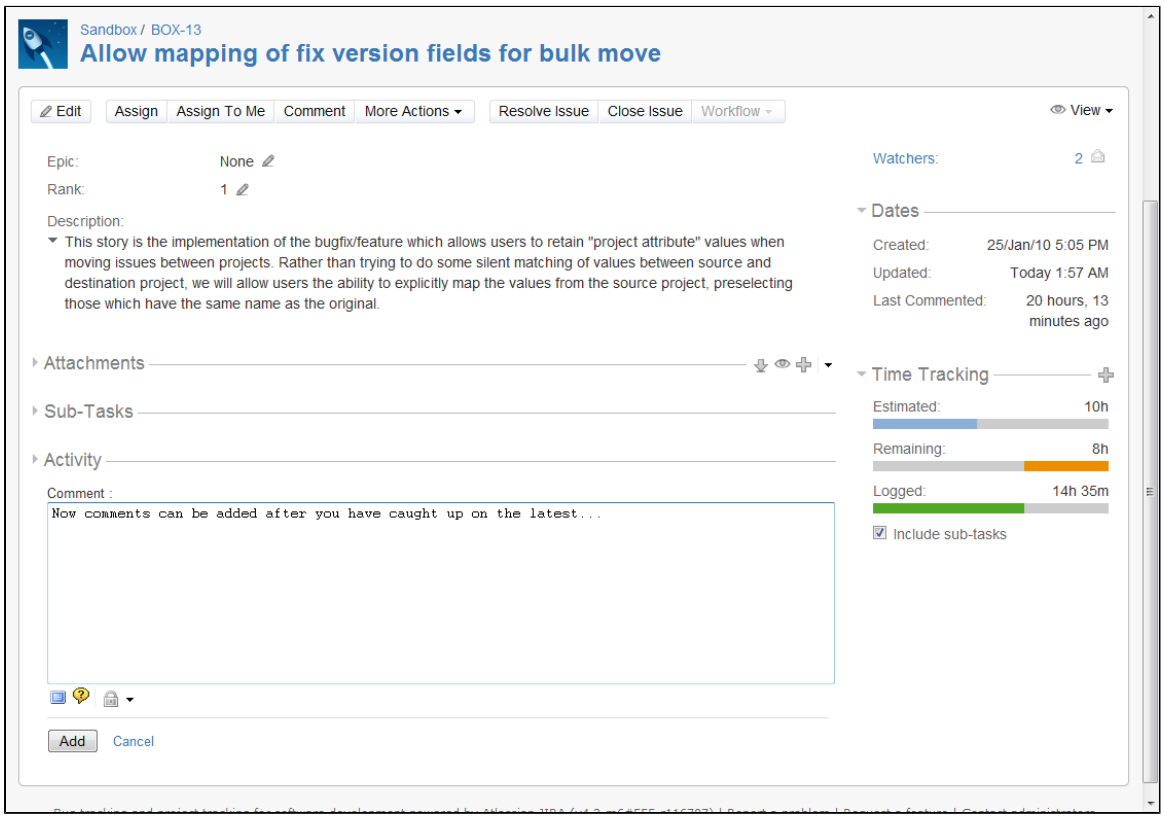
^Top

Improvements to the 'View Issue' Screen

Individual blocks can now be collapsed to streamline your issue view. The view state (i.e. which blocks are expanded/collapsed) will also be remembered across issues.

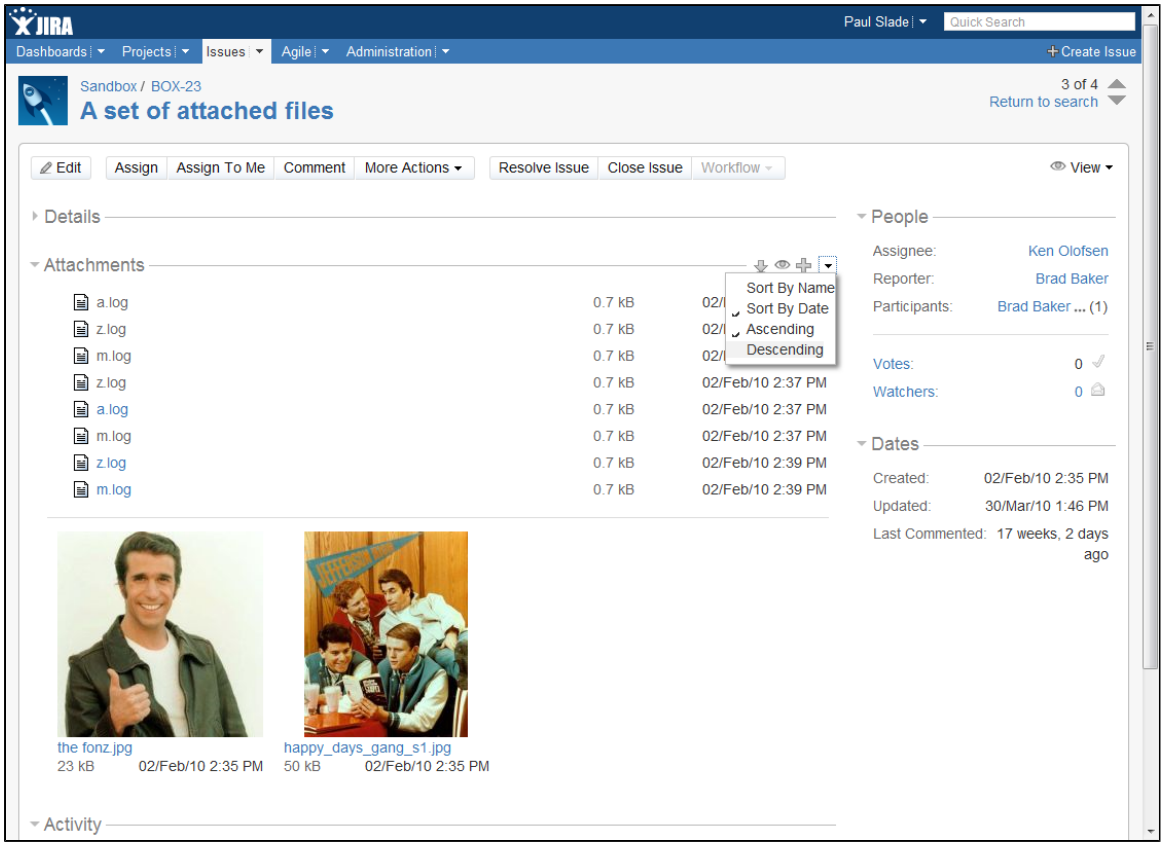


After much feedback on the 4.1 updates to the 'View Issue' screen, you can now also add a comment at the bottom of the commenting section.



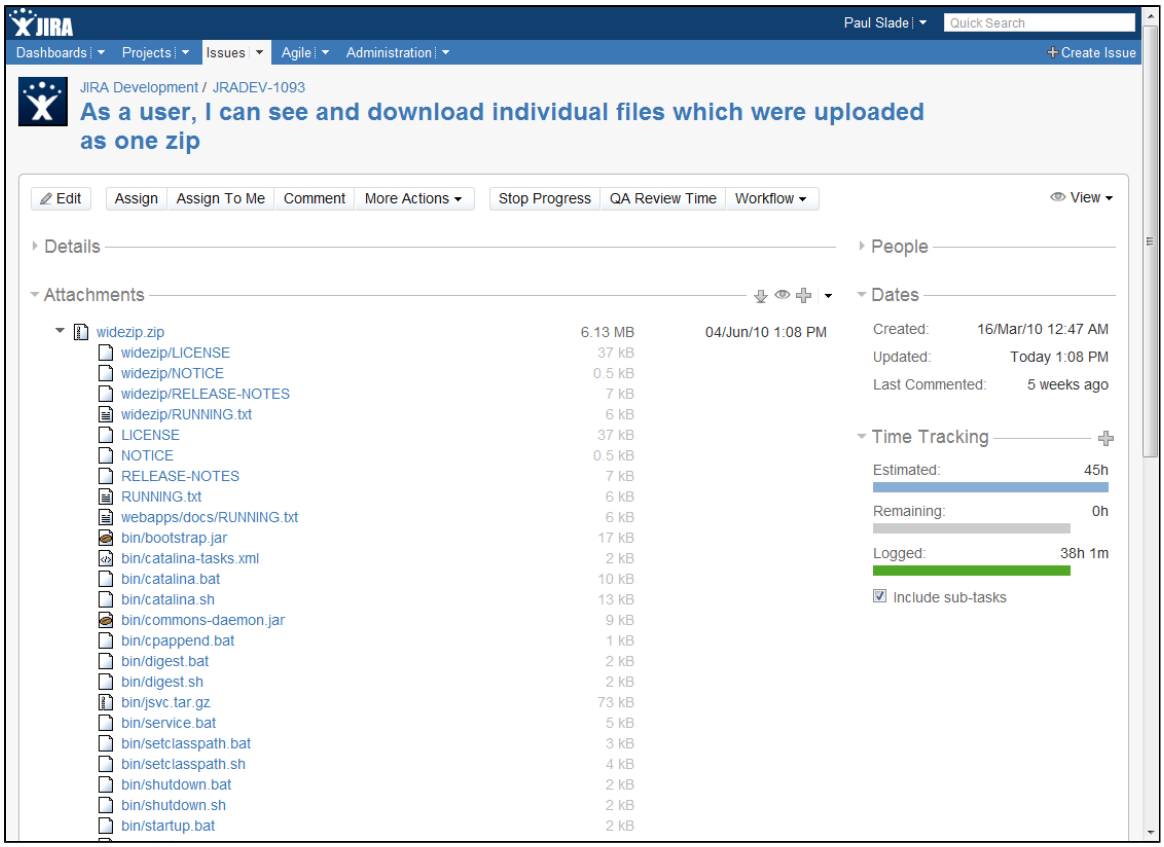
Attachment Sorting by Date or Name

Attachments can now be sorted by date or by name, in ascending or descending order.



Viewable Files in Zipped Attachments

You can expand an attached zip file to see its contents. The first 30 files will be shown for larger zip files.

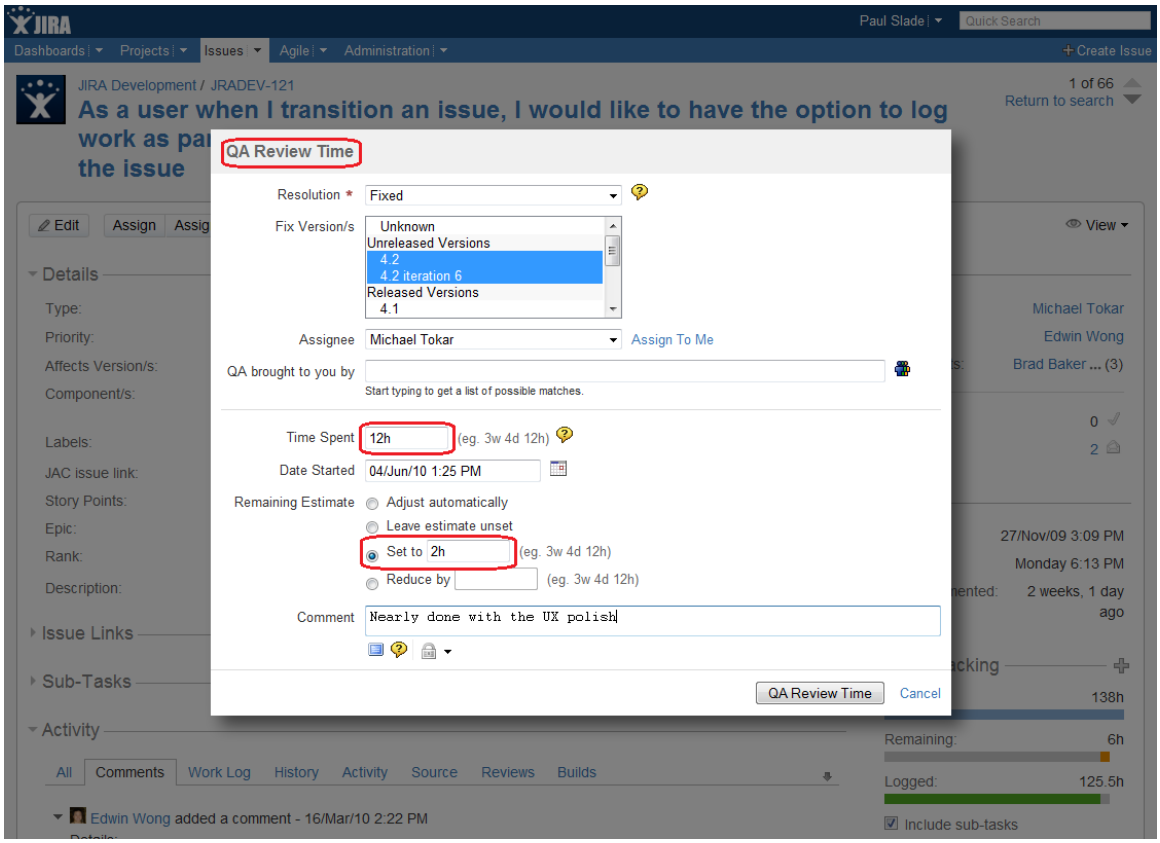


^Top

'Log Work' Improvements

You can now log work when you resolve an issue, saving you a step. You can also set the remaining time to 0 upon resolving an issue.

You also now have the ability to edit the Original Estimate, even if work has been logged on an issue (not shown here).

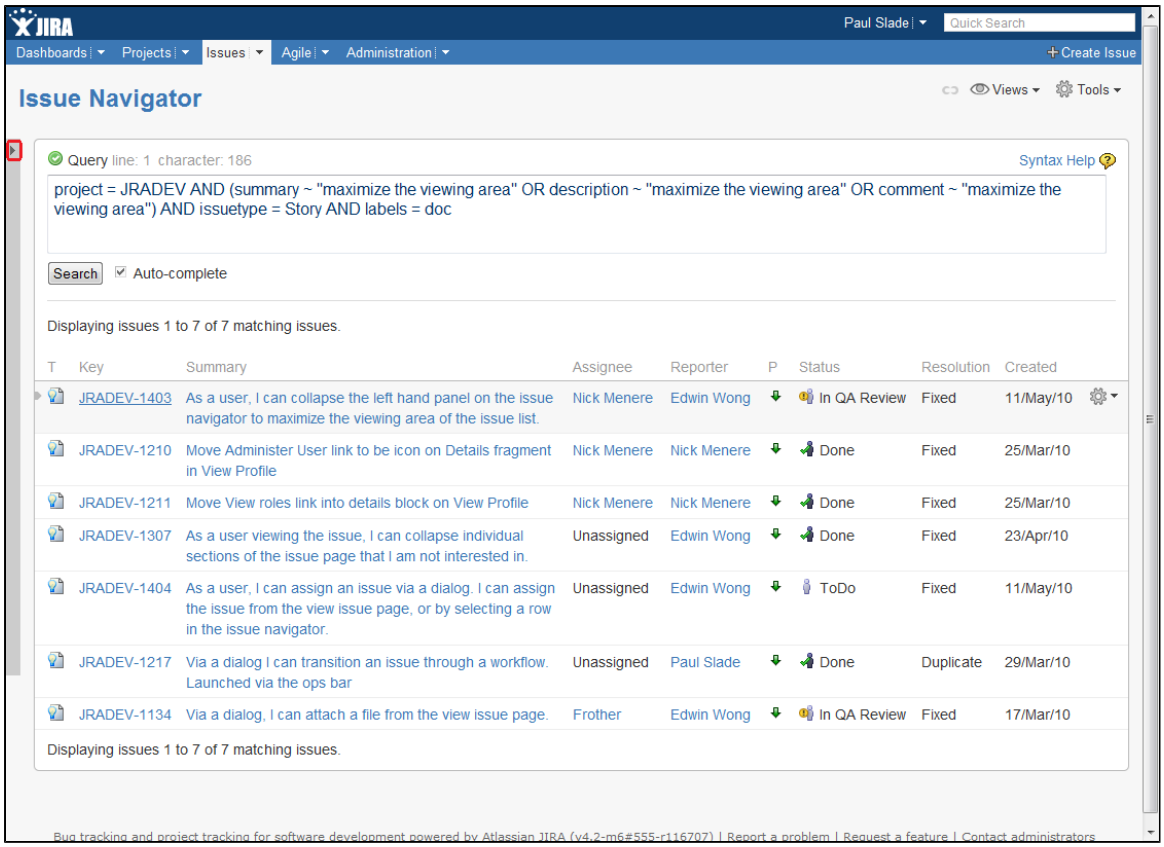


^Top

8

Improvements to the Issue Navigator

The Issue Navigator has been updated to match the look and feel of an issue. We've also added a collapsible search form to increase the size of the results area.



^Top

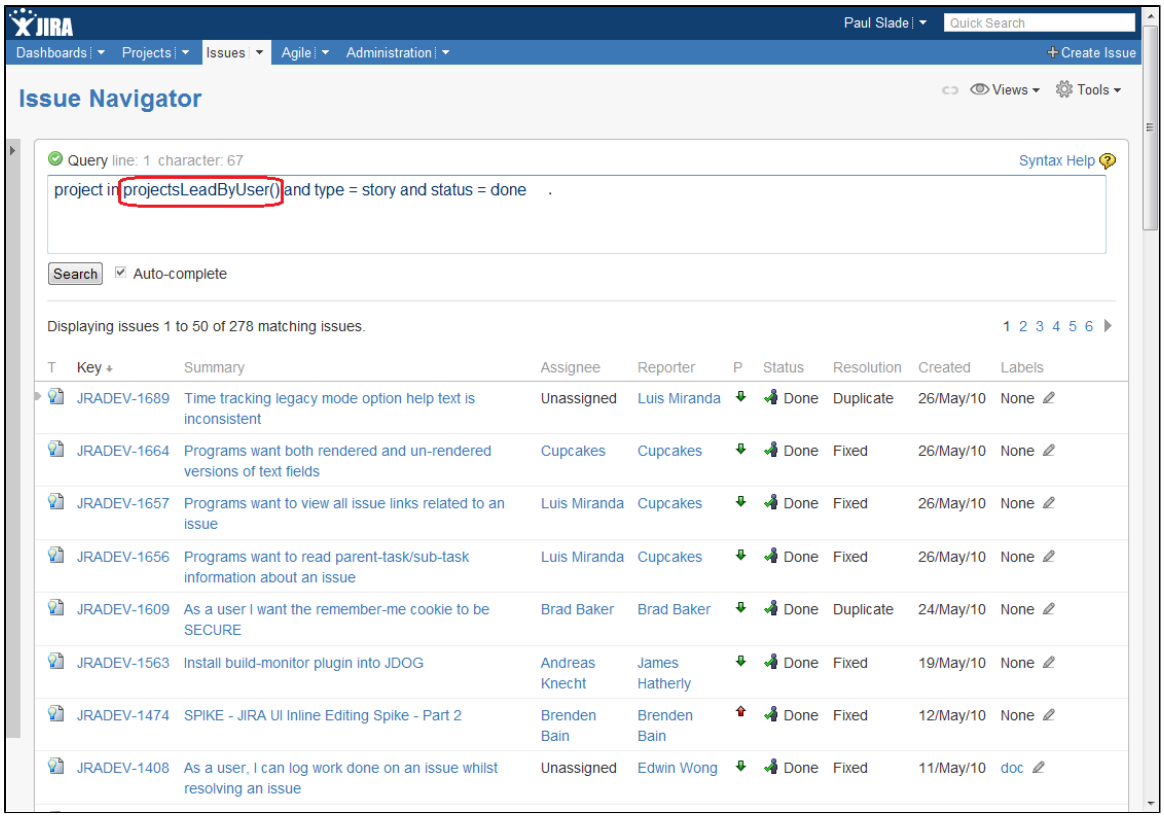
Separate Searching in Multiple Tabs

This is for those of us who are addicted to having lots of browser tabs open at once. You can now have a different JIRA search in each tab!

^Top

New JQL Functions

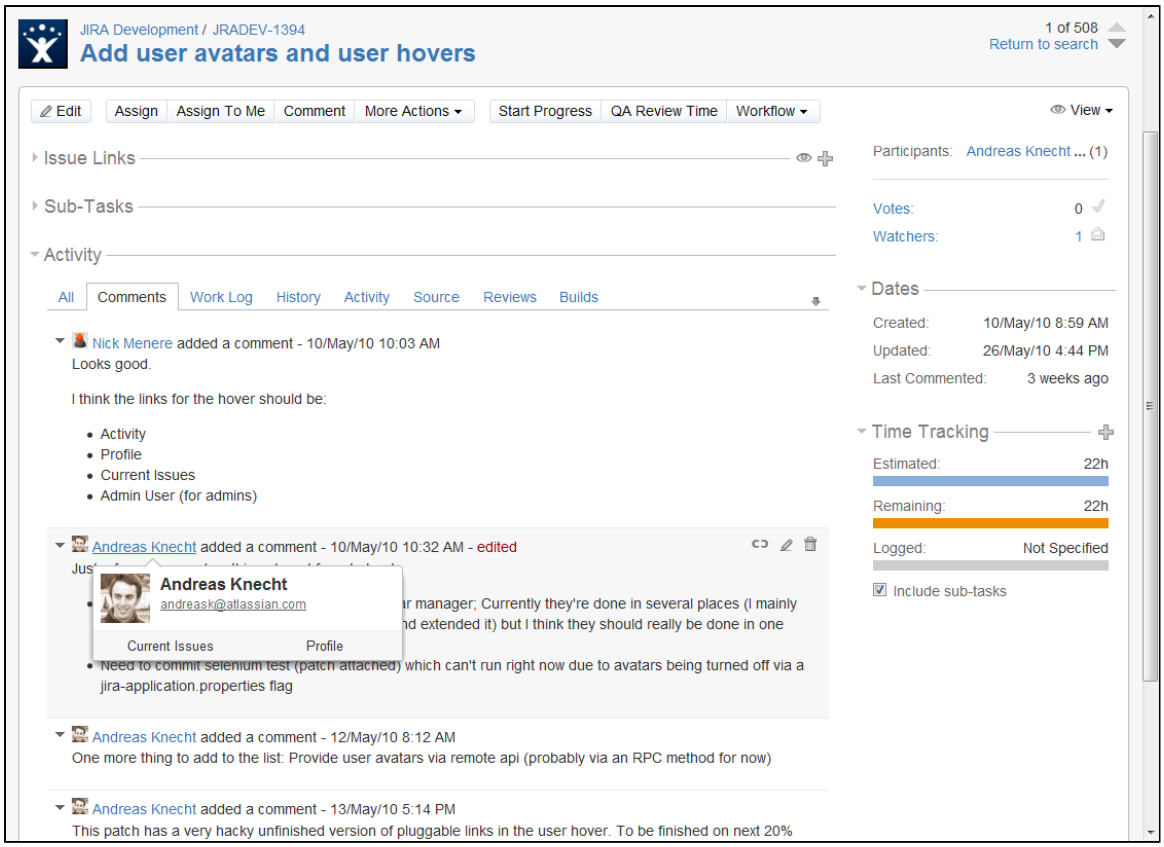
Additional JQL functions for projectLeadBy and componentsLeadBy — useful for finding all issues where a particular user is the lead of a component, or a project.



^Top

User Avatars and User Hover

JIRA users can now add an avatar to their profile. You can then mouse-hover over a user's name to show information about that user — this is available when viewing an issue, using the Issue Navigator, browsing a project, viewing activity, viewing a user profile and performing some administration tasks.



^Top

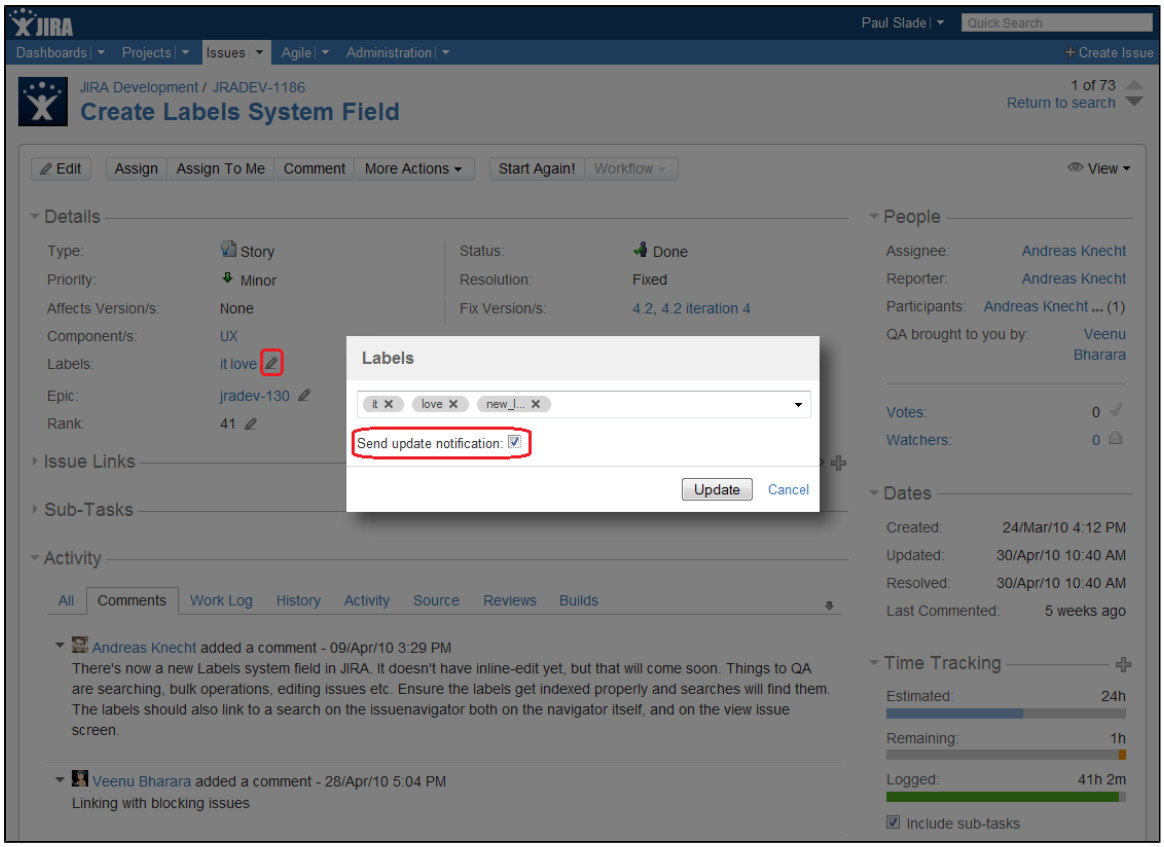
'Filter' Gadget creation via the Issue Navigator

When viewing search results, you can quickly add a 'Filter Results' gadget to your dashboard via the 'View' menu.

^Top

Labels Now Included

The functionality from the Labels plugin is now included in JIRA out-of-the-box. The labels dialog has a great new UI and issue notifications are now optional when updating labels for an issue.



^Top

Other Enhancements and Fixes

The top 50 most popular issues resolved in JIRA 4.2 are listed below. Click here for the full list.

^Top

JIRA 3.11 EAP Release Notes

Do not use in production

EAP releases should not be used in production environments as they are not officially supported.

Please also take note of the following information:

- **EAP Releases are Not Safe**— EAP releases are snapshots of the ongoing JIRA development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
- **No Upgrade Path** — Because EAP releases represent work in progress, we **can not** provide a supported upgrade path between EAP releases, or from any EAP to the eventual final release. Thus, any data you store in a JIRA EAP release may not be able to be migrated to a future JIRA release.

JIRA 3.11 EAP is a public development release leading up to JIRA 3.11.

Who should download this release?

This EAP release is being made available specifically for **JIRA plugin developers** to test their existing plugins against indexing changes in JIRA 3.11. This affects all plugins that maintain their own indexes or interact with JIRA's index (e.g. via a custom field with a custom searcher that the plugin implements).

For all production use and testing of JIRA, please use the latest official release.

Plugins that depend on indexing will have to be updated to remove the use of any methods that were deprecated in Lucene 1.9.1. These plugins will have to be re-compiled against Lucene 2.2.0 and re-released for JIRA 3.11.

What's new?

This EAP release includes the following features:

- Aggregate Time Tracking information across sub-tasks
- Lucene upgrade to v2.2.0

JIRA 3.11 aims to upgrade Lucene to version 2.2.0. A number of deprecated methods that JIRA and JIRA plugins relied on where removed from this version of Lucene. Aside from the API changes, there were also changes to how certain fields are stored in the index themselves. Luckily Lucene 2.2.0 is backwards compatible and is able to read indexes created by earlier versions of Lucene (indexes will be converted to the Lucene 2.2.0 format as soon as documents are added). Once an index has been upgraded to Lucene 2.2.0 any previous version of Lucene will not be able to read this index any longer.

One change that may require you to re-index are how dates are stored in the index. For example, DateFields have been deprecated in favour of DateTools, which stores dates in a human readable form. If you convert your plugins to store dates using DateTools, you will need to advise all users that they will need to perform a re-index (currently the only way to do this for plugins is to delete the plugin's index files).

What should I do?

If you have developed custom plugins that use Lucene, either publicly in the JIRA Plugin Library, or privately for you company's internal use, it is critical that you test those plugins against the JIRA 3.11 EAP. Because of the upgrade Lucene 2.2 and the large number of deprecations, it is likely that you will need to make changes.

You should make any necessary changes to the plugin, and be prepared to release thhose fixes in conjunction with JIRA 3.11, so that users will have a smooth upgrade experience.

If you have any trouble with the EAP or the upgrade process, please contact developer-support@atlassian.com for assistance.

The EAP is available for download here.

Security Advisories

As a public-facing web application, JIRA's application-level security is important. This document contains links to version-specific security advisories and related documents for the JIRA application.

> This document is intended to provide information to system administrators about the security of the JIRA application. It does not address JIRA's internal security model — user management and permissions — except as it relates to the overall application security.

On this page:

- Finding and Reporting a Security Vulnerability
- Publication of JIRA Security Advisories
- Severity Levels
- Our Patch Policy
- Security Advisories

Finding and Reporting a Security Vulnerability

Atlassian's approach to reporting security vulnerabilities is detailed in How to Report a Security Issue.

Publication of JIRA Security Advisories

Atlassian's approach to releasing security advisories is detailed in Security Advisory Publishing Policy.

> **Latest security advisory:**
>
> - JIRA Security Advisory 2011-09-27

Severity Levels

Atlassian's approach to categorising security issues is detailed in Severity Levels for Security Issues.

Our Patch Policy

Atlassian's approach to releasing patches for security issues is detailed in Security Patch Policy.

Security Advisories

- JIRA Security Advisory 2011-09-27
- JIRA Security Advisory 2011-02-21
- JIRA Security Advisory 2010-12-06
- JIRA Security Advisory 2010-06-18
- JIRA Security Advisory 2010-04-16
- JIRA Security Advisory 2009-04-02
- JIRA Security Advisory 2008-12-09
- JIRA Security Advisory 2008-10-29
- JIRA Security Advisory 2008-08-26
- JIRA Security Advisory 2008-02-21
- JIRA Security Advisory 2007-12-24

JIRA Security Advisory 2007-12-24

In this advisory:

- Security vulnerabilities
  - XSS vulnerability in Issue Actions
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - Anyone can delete a filter which is shared with them
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - Default language setting can be changed by an unauthorised user
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
- Available JIRA Patches
  - JIRA 3.12
  - JIRA 3.11
  - JIRA 3.10.2

Security vulnerabilities

XSS vulnerability in Issue Actions

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is an XSS (cross-site scripting) vulnerability in JIRA's issue actions, which potentially allows a malicious user (hacker) to insert their own HTML tags or script into an action.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker's text and script might be displayed to other people viewing the JIRA issue. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade to JIRA 3.12.1, or download the patch for JIRA 3.11 or 3.10.2, to fix the vulnerabilities described below.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

All issue actions (e.g. 'Create issue') are affected. The problem is with `500page.jsp`. It does not HTML-escape the error messages it prints out.

Fix

The fix is to escape all of the error messages rendered on the 500 page, so that no user input, which is propagated to error messages, is interpreted as HTML or CSS.

This issue has been fixed in JIRA 3.12.1. The fix is also provided as a patch for JIRA 3.12, 3.11 and 3.10.2. For more information, please see JRA-14105.

Anyone can delete a filter which is shared with them

Severity

Atlassian rates this vulnerability as **LOW**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw allows users to delete filters which are shared with them, which is an inconvenience to the user who is the true owner of the filter.

Atlassian recommends that you upgrade to JIRA 3.12.1, or download the patch for JIRA 3.12, 3.11 or 3.10.2, to fix the vulnerabilities described below.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could instruct all users to share their filters with trusted groups only (i.e. instruct them not to use 'Global' sharing).

Vulnerability

When a user commences deleting one of their own filters, if they replace their filter ID with the ID of another user's filter which is shared with them, they can delete the other user's filter.

Fix

The fix is to check that the currently logged-in user is indeed the owner of the filter, before deleting a filter.

This issue has been fixed in JIRA 3.12.1. The fix is also provided as a patch for JIRA 3.12, 3.11 and 3.10.2. For more information, please see JRA-13999.

Default language setting can be changed by an unauthorised user

Severity

Atlassian rates this vulnerability as **LOW**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw potentially allows a malicious user (hacker) to change the default language of your JIRA instance, which is potentially damaging to your company's reputation, and an inconvenience to users.

Atlassian recommends that you upgrade to JIRA 3.12.1, or download the patch for JIRA 3.11 or 3.10.2, to fix the vulnerabilities described below.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

After a JIRA instance has been setup, the first page of the Setup Wizard can still be accessed by manually browsing to the URL.

Attempting to advance beyond this screen, or import data, correctly results in the "Already Setup" page being displayed. However, the default language for the JIRA instance can be modified without any security checks.

Fix

The fix is to check that JIRA has not already been setup, when a user attempts to access the any page of the Setup Wizard. Similar checks also occur when a user attempts direct access to the setup JSPs.

This issue has been fixed in JIRA 3.12.1. The fix is also provided as a patch for JIRA 3.11 and 3.10.2. For more information, please see JRA-14086.

Available JIRA Patches

JIRA 3.12

The patches for JIRA 3.12 are available in the file jira_3_12_xss_patch.zip

Patch Zip File	jira_3_12_xss_patch.zip
Patch Instructions	jira_3_12_xss_patch_instructions.txt
Patch CheckSum	jira_3_12_xss_patch.zip.md5

JIRA 3.12 can also be fixed by upgrading to JIRA 3.12.1

JIRA 3.11

The patches for JIRA 3.11 are available in the file jira_3_11_xss_patch.zip

Patch Zip File	jira_3_11_xss_patch.zip
Patch Instructions	jira_3_11_xss_patch_instructions.txt
Patch CheckSum	jira_3_11_xss_patch.zip.md5

JIRA 3.10.2

The patches for JIRA 3.10 are available in the file jira_3_10_2_xss_patch.zip

Patch Zip File	jira_3_10_2_xss_patch.zip
Patch Instructions	jira_3_10_2_xss_patch_instructions.txt
Patch CheckSum	jira_3_10_2_xss_patch.zip.md5

Please let us know what you think of the format of this security advisory and the information we have provided.

JIRA Security Advisory 2008-02-21

In this advisory:

- Security vulnerabilities
  - XSS vulnerability in Issue Actions
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix

- Available JIRA Patches
  - JIRA 3.12.1
  - JIRA 3.11
  - JIRA 3.10.2

Security vulnerabilities

XSS vulnerability in Issue Actions

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is an XSS (cross-site scripting) vulnerability in JIRA's '**Saved Filter**', '**Filter Statistics**', '**Project Statistics**' and '**2D Filter Statistics**' portlets. This potentially allows a malicious user (hacker) to create a shared filter with special JavaScript in the name, and then create a link to run the vulnerable portlets using the shared filter. If this link was sent to a user and clicked by the user, the special JavaScript would be executed in the user's session.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen, by using the jelly runner.
- The hacker's text and script might be displayed to other people viewing the JIRA Dashboard. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade to JIRA 3.12.2, or download the patch for JIRA 3.12.1, 3.11 or 3.10.2, to fix the vulnerabilities described below.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

The '**Saved Filter**', '**Filter Statistics**', '**Project Statistics**' and '**2D Filter Statistics**' portlets are affected. The name of a shared filter is not HTML-escaped when the the portlet is viewed.

Fix

The fix is to escape the name of a shared filter when run by the '**Saved Filter**', '**Filter Statistics**', '**Project Statistics**' and '**2D Filter Statistics**' portlets, so that no content in the filter name is interpreted as HTML or CSS.

This issue has been fixed in JIRA 3.12.2. The fix is also provided as a patch for JIRA 3.12.1, 3.11 and 3.10.2. For more information, please see JRA-14277 and JRA-14357.

Available JIRA Patches

JIRA 3.12.1

The patches for JIRA 3.12.1 are available in the file jira_3_12_1_xss_patch.zip

Patch Zip File	jira_3_12_1_xss_patch.zip
Patch Instructions	jira_3_12_1_xss_patch_instructions.txt
Patch CheckSum	jira_3_12_1_xss_patch.zip.md5

JIRA 3.12.1 can also be fixed by upgrading to JIRA 3.12.2

JIRA 3.11

The patches for JIRA 3.11 are available in the file jira_3_11_xss_patch.zip

Patch Zip File	jira_3_11_xss_patch.zip
Patch Instructions	jira_3_11_xss_patch_instructions.txt
Patch CheckSum	jira_3_11_xss_patch.zip.md5

JIRA 3.10.2

The patches for JIRA 3.10 are available in the file jira_3_10_2_xss_patch.zip

Patch Zip File	jira_3_10_2_xss_patch.zip
Patch Instructions	jira_3_10_2_xss_patch_instructions.txt
Patch CheckSum	jira_3_10_2_xss_patch.zip.md5

Please let us know what you think of the format of this security advisory and the information we have provided.

JIRA Security Advisory 2008-08-26

In this advisory:

- Security vulnerabilities
  - XSS vulnerability in serving HTML attachments with the text/html MIME type
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - MailHandlers may create an infinite loop if the monitored mailbox receives notifications from the same instance of JIRA
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - Directory listings are enabled on Tomcat by default
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - Filters/Search Requests can be modified by URL Hacking
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - 'Manage Project Role Membership for Project' page can be viewed publicly
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix

Security vulnerabilities

XSS vulnerability in serving HTML attachments with the text/html MIME type

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and addressed a security vulnerability which may affect JIRA instances in a public environment. This is an XSS (cross-site scripting) vulnerability in JIRA's service of HTML attachments (or other active content, such as Javascript, Flash, etc) with the text/html MIME type, which potentially allows a malicious user (attacker) to insert their own HTML tags or script into an action.

- The attacker could take advantage of this vulnerability to steal other users' session cookies or other credentials, by sending the credentials back to the attacker's own web server.
- The attacker's text and script could be displayed to other people viewing the JIRA issue. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade to JIRA 3.13 to fix the vulnerabilities described below.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

If you judge it necessary, you can disable attachments or restrict public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

Any malicious script contained in an HTML attachment of with the text/html MIME type will be run as JIRA serves the attachment, i.e. when an admin or user clicks on the uploaded HTML attachment.

Fix

The fix is to add an administration option to force all attachments in JIRA to be downloaded rather than displayed inline. Administrators can choose from the following:

- force all attachments to be downloaded in JIRA,
- let all attachments be displayed inline, or,
- for Internet Explorer users, force the download of attachments that IE detects to be html files (via mime sniffing). Declared html attachments are also never displayed inline.

Read the documentation for further details on configuring this setting.

This issue has been fixed in JIRA 3.13 only. There are no patches available for previous versions of JIRA, for this fix.

MailHandlers may create an infinite loop if the monitored mailbox receives notifications from the same instance of JIRA

Severity

Atlassian rates this vulnerability as **MEDIUM**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw means that mailhandlers can potentially cause infinite loops if the monitored mailbox receives notifications from the same JIRA instance.

Atlassian recommends that you upgrade to JIRA 3.13 to fix the vulnerability described below.

Risk Mitigation

If you judge it necessary, you can disable disable your mail servers or disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade.

Vulnerability

User sends an email to a JIRA mailbox, where the From and To address are the same, e.g. if an email is sent to a mailbox monitored by JIRA with a 'From' email address identical to the mailbox address it is being sent to, then JIRA will pick up the email again and start an infinite loop for that issue.
This also applies to scenarios where JIRA sends emails to an address which is an alias for a mailbox that it checks.

Fix

The fix is to add a header to the outgoing email that contains a special JIRA "fingerprint" (X-JIRA-FINGERPRINT) that is unique to the JIRA instance.

This issue has been fixed in JIRA 3.13 only. There are no patches available for previous versions of JIRA, for this fix.

Directory listings are enabled on Tomcat by default

Severity

Atlassian rates this vulnerability as **LOW**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and addressed a security flaw which may affect JIRA instances in a public environment. This flaw means that directory listings on the Tomcat application server are public by default.

Atlassian recommends that you upgrade to JIRA 3.13 to fix the vulnerability described below. Alternatively, you can manually disable the directory listing (via the `<TOMCAT_HOME>/conf/web.xml` file in Tomcat directory), which will force JIRA to throw HTTP 404 errors appropriately.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade.

Vulnerability

Users can browse the directory listing on the Tomcat application server, e.g. `/images/`. Please note, the information accessible by the user is already readily available to the user, or can be obtained by downloading JIRA. The webapp directories **do not** contain any user content.

Fix

The fix is to disable directory listings in Tomcat. Please refer to JRA-11634 for details.

The directory listings are disabled by default in Tomcat 5.5.26. This version is bundled with the latest version of JIRA.

This issue has been fixed in JIRA 3.13 for JIRA Standalone and for the sample Tomcat (i.e. versions 4.1, 5.0, 5.5 and 6.0) configuration files shipped with JIRA WAR/EAR. There are no patches available for previous versions of JIRA, for this fix.

Filters/Search Requests can be modified by URL Hacking

Severity

Atlassian rates this vulnerability as **MODERATE**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and addressed a security flaw which may affect JIRA instances in a public environment. This flaw means that issue filters can be modified by hacking the URL, regardless of permissions on the filter.

Atlassian recommends that you upgrade to JIRA 3.13 to fix the vulnerability described below.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade.

Vulnerability

Users can run an issue filter, which they do not have access to, by entering the appropriate URL (although the filter will not return any issues that the user does not have permission to see). By the same means, users can edit a filter, rename a filter and access share and column selection. Filter deletion cannot be actioned purely by the URL, as it requires interaction with the user interface (which enforces permissions).

Fix

The fix is to revise the issue filter functionality as part of the Shareable Filters feature, so that URL hacks are no longer valid.

This issue has been fixed in JIRA 3.13 only. There are no patches available for previous versions of JIRA, for this fix.

'Manage Project Role Membership for Project' page can be viewed publicly

Severity

Atlassian rates this vulnerability as **LOW**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and addressed a security flaw which may affect JIRA instances in a public environment. This flaw means that the 'Manage Project Role Membership for Project' page can be viewed by users who are not logged in. Users cannot view any project role members or modify project roles.

Atlassian recommends that you upgrade to JIRA 3.13 to fix the vulnerability described below.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade.

Vulnerability

Users, who are not logged in, can manually enter the URL for the 'Manage Project Role Membership for Project' to access the page. Project role members will not be visible, nor will the user be able to modify project roles. The only new information available to the user will be the project name.

Fix

The fix is to prompt the user with the appropriate page for unauthorised access, if they are not logged in.

This issue has been fixed in JIRA 3.13 only. There are no patches available for previous versions of JIRA, for this fix.

Please let us know what you think of the format of this security advisory and the information we have provided.

JIRA Security Advisory 2008-10-29

In this advisory:

- Security vulnerabilities
  - XSS vulnerability on ViewProfile page
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - Return URL is not HTML escaped
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix

Security vulnerabilities

XSS vulnerability on ViewProfile page

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is an XSS (cross-site scripting) vulnerability in JIRA's '**ViewProfile**' page. This potentially allows a malicious user (hacker) to create a user with special JavaScript in the fullname of the user. If this user was viewed by another user in the ViewProfile page, the special JavaScript would be executed in the user's session.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen.

Atlassian recommends that you upgrade to JIRA 3.13.1 to fix the vulnerabilities described below.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

The '**ViewProfile**' page is affected. The user's 'fullname' is not HTML-escaped when the the page is viewed.

Fix

The fix is to HTML-encode the fullname of the user on the '**ViewProfile**' page, so that it cannot be used to run special scripts.

This issue has been fixed in JIRA 3.13.1 only. There are no patches available for previous versions of JIRA, for this fix. For more information, please see JRA-15733.

Return URL is not HTML escaped

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is an XSS (cross-site scripting) vulnerability in the returnURL parameter of the URL of a form (e.g. Add Comment). This potentially allows a malicious user (hacker) to hack the URL to insert special JavaScript in the returnURL parameter. A hacker could present the hacked URL to users (e.g. disguised in an email). If any users click the URL, the special JavaScript would be executed in the user's session.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen.
- The hacker's text and script might be displayed to other people on any JIRA page which has a form. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade to JIRA 3.13.1 to fix the vulnerabilities described below.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

All forms in JIRA are affected. The returnURL is not HTML-escaped when the the page is viewed.

Fix

The fix is to HTML-encode the returnURL of form URLs, so that it cannot be used to run special scripts.

This issue has been fixed in JIRA 3.13.1 only. There are no patches available for previous versions of JIRA, for this fix. For more information, please see JRA-15707.

Please let us know what you think of the format of this security advisory and the information we have provided.

JIRA Security Advisory 2008-12-09

In this advisory:

- Security Vulnerabilities
  - WebWork 1 Parameter Injection Hole
    - Severity
    - Risk Assessment

        - Risk Mitigation
        - Vulnerability
        - Fix
- Available JIRA Patches
    - JIRA 3.13.1
    - JIRA 3.12.3
    - JIRA 3.11
    - JIRA 3.10.2
    - JIRA 3.9.3
    - JIRA 3.8.1
    - JIRA 3.7.4
    - JIRA 3.6.5
    - JIRA 3.5.3
    - JIRA 3.4.x and earlier

Security Vulnerabilities

WebWork 1 Parameter Injection Hole

Severity

Atlassian rates this vulnerability as **CRITICAL**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is a parameter injection vulnerability in the implementation of the WebWork 1 web application framework in JIRA. The Webwork 1 web application framework allows for the dynamic transformation of URL parameters into method calls. This potentially allows a malicious user (hacker) to call exposed public methods in JIRA via specially formatted URLs.

Atlassian recommends that you upgrade to JIRA 3.13.2 to fix the vulnerabilities described below.

Risk Mitigation

We **strongly recommend** that you upgrade or apply the necessary patch as soon as possible. If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

All versions of JIRA are vulnerable to this security flaw.

A number of public JIRA methods are exposed to this vulnerability. These methods can be called via specially formatted URLs. The method names are not listed for security reasons.

Fix

The fix is to process parameters via a trusted implementation of the action factory in the Webwork 1 web application framework, which provides more secure method transformations.

This issue has been fixed in JIRA 3.13.2 or later. The fix is also provided as a patch for JIRA 3.12.3, 3.11, 3.10.2, 3.9.3, 3.8.1, 3.7.4, 3.6.5 and 3.5.3. There are no patches available for JIRA versions 3.4.x or earlier. We recommend that you upgrade to at least JIRA 3.5.x to apply this patch.

Available JIRA Patches

JIRA 3.13.1

The patches for JIRA 3.13.1 are available in the file jra-15664-3.13.1-patch.zip

Patch Zip File	jra-15664-3.13.1-patch.zip
Patch Instructions	jra-15664-3.13.1-patch-instructions.txt
Patch CheckSum	jra-15664-3.13.1-patch.zip.md5

If you are using a version of JIRA 3.13.x **prior to version 3.13.1**, you will need to upgrade to JIRA 3.13.1 before applying this patch.

JIRA 3.12.3

The patches for JIRA 3.12.3 are available in the file jra-15664-3.12.3-patch.zip

Patch Zip File	jra-15664-3.12.3-patch.zip
Patch Instructions	jra-15664-3.12.3-patch-instructions.txt
Patch CheckSum	jra-15664-3.12.3-patch.zip.md5

If you are using a version of JIRA 3.12.x **prior to version 3.12.3**, you will need to upgrade to JIRA 3.12.3 before applying this patch.

JIRA 3.11

The patches for JIRA 3.11 are available in the file jra-15664-3.11-patch.zip

Patch Zip File	jra-15664-3.11-patch.zip
Patch Instructions	jra-15664-3.11-patch-instructions.txt
Patch CheckSum	jra-15664-3.11-patch.zip.md5

JIRA 3.10.2

The patches for JIRA 3.10.2 are available in the file jra-15664-3.10.2-patch.zip

Patch Zip File	jra-15664-3.10.2-patch.zip
Patch Instructions	jra-15664-3.10.2-patch-instructions.txt
Patch CheckSum	jra-15664-3.10.2-patch.zip.md5

If you are using a version of JIRA 3.10.x prior to version 3.10.2, you will need to upgrade to JIRA 3.10.2 before applying this patch.

JIRA 3.9.3

The patches for JIRA 3.9.3 are available in the file jra-15664-3.9.3-patch.zip

Patch Zip File	jra-15664-3.9.3-patch.zip
Patch Instructions	jra-15664-3.9.3-patch-instructions.txt
Patch CheckSum	jra-15664-3.9.3-patch.zip.md5

If you are using a version of JIRA 3.9.x prior to version 3.9.3, you will need to upgrade to JIRA 3.9.3 before applying this patch.

JIRA 3.8.1

The patches for JIRA 3.8.1 are available in the file jra-15664-3.8.1-patch.zip

Patch Zip File	jra-15664-3.8.1-patch.zip
Patch Instructions	jra-15664-3.8.1-patch-instructions.txt
Patch CheckSum	jra-15664-3.8.1-patch.zip.md5

If you are using a version of JIRA 3.8.x prior to version 3.8.1, you will need to upgrade to JIRA 3.8.1 before applying this patch.

JIRA 3.7.4

The patches for JIRA 3.7.4 are available in the file jra-15664-3.7.4-patch.zip

Patch Zip File	jra-15664-3.7.4-patch.zip
Patch Instructions	jra-15664-3.7.4-patch-instructions.txt
Patch CheckSum	jra-15664-3.7.4-patch.zip.md5

If you are using a version of JIRA 3.7.x prior to version 3.7.4, you will need to upgrade to JIRA 3.7.4 before applying this patch.

JIRA 3.6.5

The patches for JIRA 3.6.5 are available in the file jra-15664-3.6.5-patch.zip

Patch Zip File	jra-15664-3.6.5-patch.zip
Patch Instructions	jra-15664-3.6.5-patch-instructions.txt
Patch CheckSum	jra-15664-3.6.5-patch.zip.md5

If you are using a version of JIRA 3.6.x prior to version 3.6.5, you will need to upgrade to JIRA 3.6.5 before applying this patch.

JIRA 3.5.3

The patches for JIRA 3.5.3 are available in the file jra-15664-3.5.3-patch.zip

Patch Zip File	jra-15664-3.5.3-patch.zip
Patch Instructions	jra-15664-3.5.3-patch-instructions.txt
Patch CheckSum	jra-15664-3.5.3-patch.zip.md5

If you are using a version of JIRA 3.5.x prior to version 3.5.3, you will need to upgrade to JIRA 3.5.3 before applying this patch.

JIRA 3.4.x and earlier

There are no patches available for JIRA versions 3.4.x or earlier. We recommend that you upgrade to at least JIRA 3.5.x.

Please let us know what you think of the format of this security advisory and the information we have provided.

JIRA Security Advisory 2009-04-02

In this advisory:

- Security Vulnerabilities
  - HTTP Header Injection Flaw
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix
  - Available JIRA Patches
    - JIRA 3.12.3
    - JIRA 3.11
    - JIRA 3.10.x and earlier
  - DWR XSS Security Hole
    - Severity
    - Risk Assessment

- Risk Mitigation
- Vulnerability
- Fix
- Available JIRA Patches
  - JIRA 3.12.3
  - JIRA 3.11
  - JIRA 3.10.x and earlier
- XSS vulnerability in various JIRA parameters
  - Severity
  - Risk Assessment
  - Risk Mitigation
  - Vulnerability
  - Fix
- Security Vulnerabilities — JIRA Plugins
  - JIRA Charting Plugin XSS Security Hole
    - Severity
    - Risk Assessment
    - Risk Mitigation
    - Vulnerability
    - Fix

Security Vulnerabilities

HTTP Header Injection Flaw

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is a HTTP Header injection vulnerability in JIRA. This potentially allows a malicious user (hacker) to hack the header response to insert malicious code. A hacker could present the hacked URL to users (e.g. disguised in an email). If any users click the URL, the malicious code would be executed in the user's session.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen.
- The hacker could redirect the user to undesirable web sites. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade to JIRA 3.13.3 to fix the vulnerabilities described below.

Risk Mitigation

We strongly recommend that you upgrade or apply the necessary patch as soon as possible.

If you are unable to do this, you may wish to consult the vendor of your application server to see whether your application server is immune to header injection vulnerabilities or has configuration options to prevent such attacks. For example, the Coyote (HTTP) connector in Tomcat version 5.5 and later is immune to header injection attacks, as acknowledged in this reference.

Please note, the time required to fix this vulnerability and the extent of its effectiveness will depend on your application server and its configuration.

Technical Note

In your application server, header injection vulnerabilities can be mitigated if the `setHeader()`, `addHeader()`, and `sendRedirect()` methods in the `HttpServletResponse` class have their parameters properly checked for header termination characters. You may wish to forward this information to the vendor of your application server to help them advise whether they have any countermeasures to protect your application server against header injection attacks.

Vulnerability

All versions of JIRA are vulnerable to this security flaw.

Fix

The fix updates the Seraph framework to a version which correctly encodes and validates redirect URLs before sending them back to the user.

This issue has been fixed in JIRA 3.13.3 or later. The fix is also provided as a patch for JIRA 3.12.3 and 3.11. There are no patches available for JIRA versions 3.10.x and earlier. We recommend that you upgrade to at least JIRA 3.11 to apply this patch.

Available JIRA Patches

JIRA 3.12.3

A replacement seraph jar for JIRA 3.12.3 is available here: atlassian-seraph-0.38.3.jar

Replace JIRA's existing seraph jar with the updated one:

1. Delete the existing seraph jar in WEB-INF/lib/atlassian-seraph-0.37.2.jar
2. Place the replacement atlassian-seraph-0.38.3.jar into WEB-INF/lib

jar file	atlassian-seraph-0.38.3.jar
MD5 sum	atlassian-seraph-0.38.3.jar.md5

JIRA 3.11

A replacement seraph jar for JIRA 3.11 is available here: seraph-0.7.21.1.jar

Replace JIRA's existing seraph jar with the updated one:

1. Delete the existing seraph jar in WEB-INF/lib/seraph-0.7.21.jar
2. Place the replacement seraph-0.7.21.1.jar into WEB-INF/lib

jar file	seraph-0.7.21.1.jar
MD5 sum	seraph-0.7.21.1.jar.md5

JIRA 3.10.x and earlier

There are no patches available for JIRA versions 3.10.x or earlier. We recommend that you upgrade to at least JIRA 3.11.

DWR XSS Security Hole

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a security flaw which may affect JIRA instances in a public environment. This flaw is a XSS vulnerability in the DWR library in JIRA. This potentially allows a malicious user (hacker) to hack the URL to insert special JavaScript. A hacker could present the hacked URL to users (e.g. disguised in an email). If any users click the URL, the special JavaScript would be executed in the user's session.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen.
- The hacker's text and script might be displayed to other people on any JIRA page which has a form. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade to JIRA 3.13.3 to fix the vulnerabilities described below.

Risk Mitigation

We recommend that you upgrade or apply the necessary patch as soon as possible. If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

All versions of JIRA are vulnerable to this security flaw.

Fix

The fix is to upgrade the DWR library shipped with JIRA to version 2.0.3. This version of the DWR library does not have this security flaw.

This issue has been fixed in JIRA 3.13.3 or later. The fix is also provided as a patch for JIRA 3.12.3 and 3.11. There are no patches available for JIRA versions 3.10.x or earlier. Please see JRA-16072 for further details.

Available JIRA Patches

JIRA 3.12.3

The patches for JIRA 3.12.3 are available in the file jra-16072-3.12.3-patch.zip

Patch Zip File	jra-16072-3.12.3-patch.zip
Patch Instructions	jra-16072-3.12.3-patch-instructions.txt
Patch CheckSum	jra-16072-3.12.3-patch.zip.md5

If you are using a version of JIRA 3.12.x prior to version 3.12.3, you will need to upgrade to JIRA 3.12.3 before applying this patch.

JIRA 3.11

The patches for JIRA 3.11 are available in the file jra-16072-3.11-patch.zip

Patch Zip File	jra-16072-3.11-patch.zip
Patch Instructions	jra-16072-3.11-patch-instructions.txt
Patch CheckSum	jra-16072-3.11-patch.zip.md5

JIRA 3.10.x and earlier

There are no patches available for JIRA versions 3.10.x or earlier. We recommend that you upgrade to at least JIRA 3.11.

XSS vulnerability in various JIRA parameters

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a number of security flaws which may affect JIRA instances in a public environment. The flaws are all XSS (cross-site scripting) vulnerabilities in various JIRA parameters. Each vulnerability potentially allows a malicious user (hacker) to embed their own JavaScript into a JIRA page.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen.

Atlassian recommends that you upgrade to JIRA 3.13.3 to fix the vulnerabilities described below.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system until you have applied the necessary patch or upgrade. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

A hacker can inject their own JavaScript into various JIRA parameters, described in the table below. If rogue JavaScript is injected into a parameter of a URL, the JavaScript will be executed when a user invokes the URL for the page.

JIRA page	Description
lazyLoader *(portlet loader)*	portletId
`CreateIssueDetails.jspa`	duedate
`EditIssue.jspa`	duedate
`jira.issueviews:searchrequest-fullcontent/temp/SearchRequest.html`	sorter/field, sorter/order
`jira.issueviews:searchrequest-printable/temp/SearchRequest.html`	sorter/order

For more information, please see JRA-16369.

Fix

The fix is to HTML-encode the vulnerable parameters to prevent scripts from being executed from them.

This issue has been fixed in JIRA 3.13.3 only. There are no patches available for previous versions of JIRA, for this fix.

Security Vulnerabilities — JIRA Plugins

JIRA Charting Plugin XSS Security Hole

Severity

Atlassian rates this vulnerability as **HIGH**, according to the scale published in the JIRA Security documentation. This scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed two security flaws in the JIRA Charting plugin which may affect JIRA instances in a public environment that use this plugin. These flaws are XSS vulnerabilities in view actions for the JIRA Charting plugin. This potentially allows a malicious user (hacker) to hack the URL to insert special JavaScript. A hacker could present the hacked URL to users (e.g. disguised in an email). If any users click the URL, the special JavaScript would be executed in the user's session.

- The hacker might take advantage of this flaw to steal other users' session cookies or other credentials, by sending the credentials back to the hacker's own web server.
- The hacker could also gain control over the underlying system, based on the privileges of the user whose session cookie has been stolen.
- The hacker's text and script might be displayed to other people on any JIRA page which has a form. This is potentially damaging to your company's reputation.

Atlassian recommends that you upgrade your JIRA Charting plugin to version 1.4.1 to fix the vulnerabilities described below.

Risk Mitigation

We recommend that you upgrade your JIRA Charting plugin as soon as possible. If you judge it necessary, you can disable public access (i.e. anonymous access and public signup) to your JIRA system. For even tighter control, you could restrict JIRA access to trusted groups only.

Vulnerability

JIRA instances that use the JIRA Charting plugin (any version) are vulnerable to this security flaw.

Fix

The fix is to HTML encode the appropriate values in the JIRA Charting plugin actions. Please see JCHART-256 and JCHART-257 for further

details.

This issue has been fixed in the JIRA Charting plugin 1.4.1 or later. Please see the plugin page to check compatibility with your JIRA version.

Please let us know what you think of the format of this security advisory and the information we have provided.

JIRA Security Advisory 2010-04-16

Several security vulnerabilities have been exposed on JIRA. Please refer to this document **before proceeding** to determine if your system has been compromised.

In this advisory:

- Privilege Escalation Vulnerabilities
  - Severity
  - Risk Assessment
  - Risk Mitigation
  - Vulnerability
  - Fix
- XSS Vulnerabilities in JIRA
  - Severity
  - Risk Assessment
  - Risk Mitigation
  - Vulnerability
  - Fix
- Available Patches

Privilege Escalation Vulnerabilities

Severity

Atlassian rates these vulnerabilities as **critical**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed several privilege escalation vulnerabilities, which may affect JIRA instances. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of JIRA.

An attacker, who has gained administrator access to a JIRA instance, could set the attachment, index or backup paths to a location within the JIRA web application directory. Once this has been done, the attacker can upload malicious code that can execute in the context of the user running the application server in which JIRA is deployed. The attacker could potentially modify JIRA's files and capture user credentials. If you have followed standard guidelines for hardening your application servers, then your instance should be less susceptible to this vulnerability.

The JIRA web application directory is either the `atlassian-jira` subdirectory (for JIRA Standalone installations) or the `webapps` subdirectory for JIRA WAR installations on Tomcat. For other application servers, please consult that application server's relevant documentation for discovering the web application directory.

Risk Mitigation

We strongly recommend either upgrading or patching your JIRA installation to fix these vulnerabilities. Please see the 'Fix' section below.

Note: If you are an Atlassian JIRA Studio or Hosted customer, we have assessed that your system is secure and implemented additional protections for it.

We also strongly recommend that you secure your JIRA instance by following these instructions, even if you are not in a position to apply the patches immediately.

Vulnerability

All versions of JIRA are affected by these privilege escalation vulnerabilities.

As a consequence of these security fixes, the following changes to JIRA's behaviour have occurred.

- JIRA now recognises a new variable called (`jira.paths.set.allowed`) in the `jira-application.properties` file that collectively enables or disables the following capabilities through the JIRA user interface:

- Setting the attachments directory
- Setting the indexing directory
- Setting the backup directory for the backup service
- Restoring XML data from a JIRA XML backup
- Setting the directory in the "Create issues from local files" service
- Viewing the list of administrators through the "Contact Administrators" link in the footer.

  On initial application of this patch, the `jira.paths.set.allowed` property will not be present in this file and all settings above will be disabled by default. We recommend that this property be absent from your `jira-application.properties` file or if it is present, set its value to `false`.

- JIRA now recognises another new variable called (`jira.paths.safe.backup.path`) in the `jira-application.properties` file which specifies a safe path for XML backup. This property only applies to the 'Backup Data to XML' function and not the scheduled backup service. If the property is not present, 'Backup Data to XML' will not be allowed. The file name specified in the user interface will be appended to the safe path and used to determine the destination of the backup file. Please ensure that the safe path is separate from your web application directory.

  On initial application of this patch, the `jira.paths.safe.backup.path` property will not be present in this file.

- System logs and customer data from generated support requests have been removed. The automatically generated support request sent to Atlassian will no longer include system logs and the XML backup.

Fix

These issues have been fixed in JIRA 4.1.1 and later.

These fixes are also provided as a patch for JIRA 4.1 and previous versions of JIRA. See Available Patches (below) for the complete list of available patches.

These patches are also available from JIRA issue JRA-21004. These patches also address the XSS vulnerabilities described below.

In addition to patching your instance, we strongly recommend that you also review these instructions on securing your JIRA instance (and any other web application).

XSS Vulnerabilities in JIRA

Severity

Atlassian rates these vulnerabilities as **critical**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed several cross-site scripting (XSS) vulnerabilities in JIRA, which may affect JIRA instances. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of JIRA.

- The attacker might take advantage of the vulnerability to steal other users' session cookies or other credentials, by sending the credentials back to the attacker's own web server.
- The attacker's text and script might be displayed to other people viewing a JIRA page. This is potentially damaging to your company's reputation.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Risk Mitigation

We strongly recommend either upgrading or patching your JIRA installation to fix these vulnerabilities. Please see the 'Fix' section below.

We also strongly recommend that you secure your JIRA instance by following these instructions, even if you are not in a position to apply the patches immediately.

Vulnerability

All versions of JIRA are affected by these XSS vulnerabilities.

An attacker can inject their own JavaScript into the JIRA components listed in the table below. Each of the actions is invoked when a user performs a specific function in JIRA, such as clicking a link or a button. The actions can also be invoked by simply entering the URL into the browser address bar. The rogue JavaScript will be executed when a user invokes the URL.

JIRA page	Routes of XSS attack

Colour Picker (colorpicker.jsp)	XSS code injection into the 'element' or 'defaultColor' URL parameters.
User Picker (userpicker.jsp)	XSS code injection into the 'formName' or 'element' URL parameters. The full name field is another route, in which XSS scripts in this field can be executed when a user views its field content via the User Picker.
Group Picker (grouppicker.jsp)	XSS code injection into the 'formName' or 'element' URL parameter. The group name field is another route, in which code in this field can be executed when a user views its field content via the Group Picker.
Announcement Banner Preview	If the URL parameter 'announcement_preview_banner_st' is appended to the URL for most pages in JIRA, it is a potential route for exploitation by XSS scripts.
Support-related JSP pages	The following JSP pages can be exploited by XSS scripts. We have disabled these pages in JIRA and they are no longer available. • .../secure/admin/groupnames.jsp • .../secure/admin/indexbrowser.jsp • .../secure/admin/debug/classpath-debug.jsp • .../secure/admin/viewdocument.jsp • .../secure/admin/cleancommentspam.jsp
runportleterror.jsp	XSS code injection into the 'portletKey' URL parameter.
issuelinksmall.jsp	XSS scripts appended to the end of the URL.
screenshot-redirecter.jsp	XSS code injection into the 'afterURL' URL parameter.
500page.jsp	XSS code injection into the 'Referrer' HTTP request header.

Fix

These issues have been fixed in JIRA 4.1.1 and later.

These fixes are also provided as a patch for JIRA 4.1 and previous versions of JIRA. See Available Patches (below) for the complete list of available patches.

These patches are also available in JIRA issue JRA-21004. The patches also address the privilege escalation vulnerabilities described above.

> In addition to patching your instance, we strongly recommend that you also review these instructions on securing your JIRA instance (and any other web application).

Available Patches

The available patches address both the Privilege Escalation and XSS Vulnerabilities. They can be obtained from JRA-21004, or directly downloaded from the table below. To install the patch, please follow the instructions in the patch file.

> The patches below override the patches previously available at JRA-20994 and JRA-20995. We have incorporated both patches into one. Please ensure that you install this unified patch regardless of whether you have previously applied patches at JRA-20994 or JRA-20995 as it contains additional improvements. You do not need to uninstall previous patches.

Version	File
4.1	patch-JRA-21004-4.1.zip
4.0.2	patch-JRA-21004-4.0.2.zip

4.0.1	patch-JRA-21004-4.0.1.zip
4.0	patch-JRA-21004-4.0.zip
3.13.5	patch-JRA-21004-3.13.5.zip
3.13.4	patch-JRA-21004-3.13.4.zip
3.13.3	patch-JRA-21004-3.13.3.zip
3.13.2	patch-JRA-21004-3.13.2.zip
3.13.1	patch-JRA-21004-3.13.1.zip
3.13	patch-JRA-21004-3.13.zip
3.12.3	patch-JRA-21004-3.12.3.zip
3.12.2	patch-JRA-21004-3.12.2.zip
3.12.1	patch-JRA-21004-3.12.1.zip
3.12	patch-JRA-21004-3.12.zip

Several security vulnerabilities have been exposed on JIRA. Please refer to this page to determine if your system has been compromised.

Security Addendum 2010-04-16 - Determining if your public JIRA instance has been compromised

This information only applies to **JIRA instances accessible from the Internet**.

If you are an Atlassian JIRA Studio or Hosted customer, we have assessed that your system is secure and implemented additional protections for it.

On this page:

- Overview
- 1. Check for modified files on the server
- 2. Check your access logs for the attack vectors
  - 2.1 Administrative setting changes from unknown IP addresses
  - 2.2 Check for an unusually large number of login attempts
  - 2.3 Cross-Site Scripting attacks
- 3. Verify the integrity of existing JIRA administrator accounts
- Additional Resources
- If you suspect that your JIRA instance has been compromised

Overview

In April 2010, some public JIRA sites were attacked via security vulnerabilities in JIRA. This document provides instructions on how to determine if your JIRA instance has been compromised. Please refer to the JIRA Security Advisory 2010-04-16 for more information about these vulnerabilities and patching your JIRA instances.

The attacker would require web access to your JIRA instance. If your JIRA instance is behind a firewall and you are maintaining usual security measures to restrict external access to this JIRA instance (for example, removing user accounts of individuals who no longer require access to it), then there is low risk of your JIRA instance being attacked.

If your JIRA instance was compromised, the attacker would have initially gained administrative privileges via an XSS attack or by successfully discovering a JIRA administrator's password. Once the system is compromised, the attacker would be able to read and modify files and database information.

IMPORTANT!

If it is determined that your JIRA instance has been compromised, our advice is to immediately shut down JIRA and disconnect the server from the network/Internet. Also, you may want to immediately shut down any other systems which potentially share a userbase or have common username/password combinations with the compromised system. Do not apply the patch described in JIRA Security Advisory 2010-04-16 until you have worked with your local security team to identify the scope of the breach and your recovery options.

To determine if your JIRA instance has been compromised, please do the following:

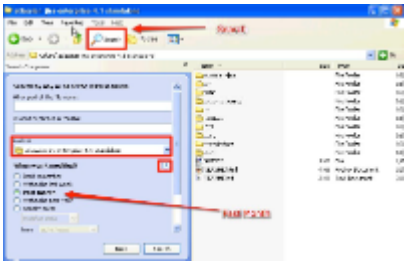
1. Check the server running JIRA for recently modified files
2. Check your access logs for the attack vectors
3. Verify the integrity of existing JIRA administrator accounts

1. Check for modified files on the server

Running the following command in UNIX-based systems (for example, Linux and Mac OS X) will show all files modified in the last fifteen days:

```
find /path/to/JIRA -mtime -15
```

On Windows, you can search for files using the graphical search utility:



Check for any files in the JIRA installation that have not been modified by you or one of your known administrators within this time period.

The files which are likely to have been affected by these attacks include the following:

- a modified `WEB-INF/web.xml` file, such as the addition of new servlet filters
- newly added or modified JAR files in `WEB-INF/lib/`
- newly added JSP files at various places inside the web application
- newly added GIF files in `/images/`

This information only refers to known exploits. You should check *all* modified files in the web application, including files for which you do not have records of having changed and compare them to an unmodified copy of a JIRA distribution, such as one downloaded from the Atlassian website.

If you need more information, please contact Atlassian support using the **Get Support** link below.

2. Check your access logs for the attack vectors

JIRA does not keep access logs unless you have manually configured it. However, many web servers like Apache HTTPD are set up to capture access logs by default. If your web server or application server or JIRA has been configured to generate access logs, you can use these logs to check for the access patterns below.

To check for patterns in access logs on Unix-based systems (e.g. Linux, Mac OS X), you can use the `grep` tool on one or more files:

```
grep 'search-string' *.log
```

To check for patterns in access logs on Windows, you can use the `findstr` tool on the command line:

```
findstr "search-string" *.log
```

2.1 Administrative setting changes from unknown IP addresses

The attacker may have modified the attachment directory configured in the JIRA administration area. To check for access to the vulnerable settings pages, search the access logs for:

- `secure/admin/EditAttachmentSettings`
- `secure/admin/IndexReIndex`
- `secure/admin/EditService`
- `secure/admin/XmlBackup`
- `secure/admin/XmlRestore`
- `secure/admin/JiraSupportRequest`

Below is an example of access logs which contain changes to the attachment settings on a JIRA instance.

```
...
...
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "GET
/secure/admin/jira/EditAttachmentSettings!default.jspa HTTP/1.1" 200 7259 "-" "-" 518092
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST
/secure/admin/jira/EditAttachmentSettings.jspa HTTP/1.1" 302 20 "-" "-" 50425
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "GET
/secure/admin/jira/EditAttachmentSettings!default.jspa HTTP/1.1" 200 7288 "-" "-" 53665
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST
/secure/admin/jira/EditAttachmentSettings.jspa HTTP/1.1" 302 20 "-" "-" 13190
...
...
```

If you need more information, please contact Atlassian support using the **Get Support** link below.

2.2 Check for an unusually large number of login attempts

It is possible that administrative access may have been gained via a brute-force attack. To get a list of all login attempts, search the access logs for:

- `/login.jsp`

Evidence of such an attack will look like this in the access logs where the timestamps between requests are very close together and the number of attempts is extremely high:

```
...
...
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5547 "-" "-"
126482
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5542 "-" "-"
119285
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5547 "-" "-"
119801
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5557 "-" "-"
117931
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5546 "-" "-"
116953
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5547 "-" "-"
125371
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5549 "-" "-"
117773
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5548 "-" "-"
119681
xxx.xxx.xxx.xxx - - [xx/Apr/2010:00:00:00 -0500] "POST /login.jsp HTTP/1.1" 200 5545 "-" "-"
126654
...
...
```

If you need more information, please contact Atlassian support using the **Get Support** link below.

2.3 Cross-Site Scripting attacks

Cross-Site Scripting (XSS) attacks were attempted on sites at Apache. These XSS attacks were attempts to steal other users' session

cookies or other credentials, by sending the credentials back to the attacker's own web server.

For more information about XSS attacks, please refer to the relevant articles on the cgisecurity, CERT websites.

If XSS attacks have occurred on your JIRA instance, the following strings may be present in your access logs:

- `<script`
- `%3Cscript`
- `</script>`
- `document.cookie`
- `document.write`
- `window.location`

More advanced patterns to identify XSS requests (and those of other injection-type attacks) from access logs can be found on Symantec's Detection of SQL Injection and Cross-site Scripting Attacks.

If you need more information, please contact Atlassian support using the **Get Support** link below.

3. Verify the integrity of existing JIRA administrator accounts

The attacker could gain administration access via a brute-force attack to determine an administrator's password. Once access is gained, a number of different actions could be performed, including:

- The addition of a new administrative account
  or
- Modification of an existing user account's email address or password and subsequent use of that account

You should check that all emails are valid and that all administrator accounts are known users.

If you need more information, please contact Atlassian support using the **Get Support** link below.

Additional Resources

- Apache Blog
- cgisecurity article
- CERT article

If you suspect that your JIRA instance has been compromised

If you suspect your JIRA instance has been compromised we strongly recommend involving your local security team for further investigation. Atlassian is happy to review your customer log files and provide an opinion on whether your system has been compromised. To request this please file a support request via http://support.atlassian.com/.

Please note, however, that the final determination of whether your JIRA instance has been compromised and what actions to take as a result remains with you the customer.

If it is determined that your JIRA instance has been compromised, our advice is to immediately shut down JIRA and disconnect the server from the network/Internet. Also, you may want to immediately shut down any other systems which potentially share a userbase or have common username/password combinations with the compromised system. Do not apply the patch described in JIRA Security Advisory 2010-04-16 until you have worked with your local security team to identify the scope of the breach and your recovery options.

Security Addendum 2010-04-16 - Preventing security attacks

In April 2010, JIRA sites were attacked via security vulnerabilities in JIRA. These vulnerabilities will be fixed in JIRA 4.1.1, and patches are available for earlier versions of JIRA.

For more information:

- about these vulnerabilities and patching your JIRA instance, see JIRA Security Advisory 2010-04-16
- on how to determine whether your public JIRA instance has been compromised please refer to the detection guide.

Note: If you are an Atlassian JIRA Studio or Hosted customer, we have assessed that your system is secure and implemented additional protections for it.

To the best of our knowledge, the following guidelines will help prevent attacks of the kind recently experienced.

- 1. Use Strong Passwords
  - 1.1 Administrators should use Strong Passwords
  - 1.2 Administrators should have Different Passwords for Different Systems
- 2. Apply JIRA Security Patches
- 3. Protect Against Brute Force Attack
  - 3.1 Upgrade to JIRA 4.1
  - 3.2 Enable Brute Force Login Protection on your Web Server
- 4. Restrict Network Access to Administrative Sections of Applications
- 5. Restrict File System Access by the Application Server

- 6. Disable Jelly

1. Use Strong Passwords

1.1 Administrators should use Strong Passwords

All your JIRA administrators, JIRA system administrators and administrators of all Atlassian products should have strong passwords. Ask your administrators to update their passwords to strong passwords.

Do not use passwords that are dictionary words. Use mixed-case letters, numbers and symbols for your administrator passwords and make sure they are sufficiently long (e.g. 14 characters). We encourage you to refer to the Strong Password Generator for guidelines on selecting passwords.

Using strong passwords greatly increases the time required by an attacker to retrieve your passwords by brute force, making such an attack impractical.

1.2 Administrators should have Different Passwords for Different Systems

As well as choosing a strong password, administrators should have *different* strong passwords for different systems.

This will reduce the impact the attacker can have if they do manage to obtain administrator credentials on one of your systems.

2. Apply JIRA Security Patches

Apply the patches found in JIRA Security Advisory 2010-04-16 for your version of JIRA.

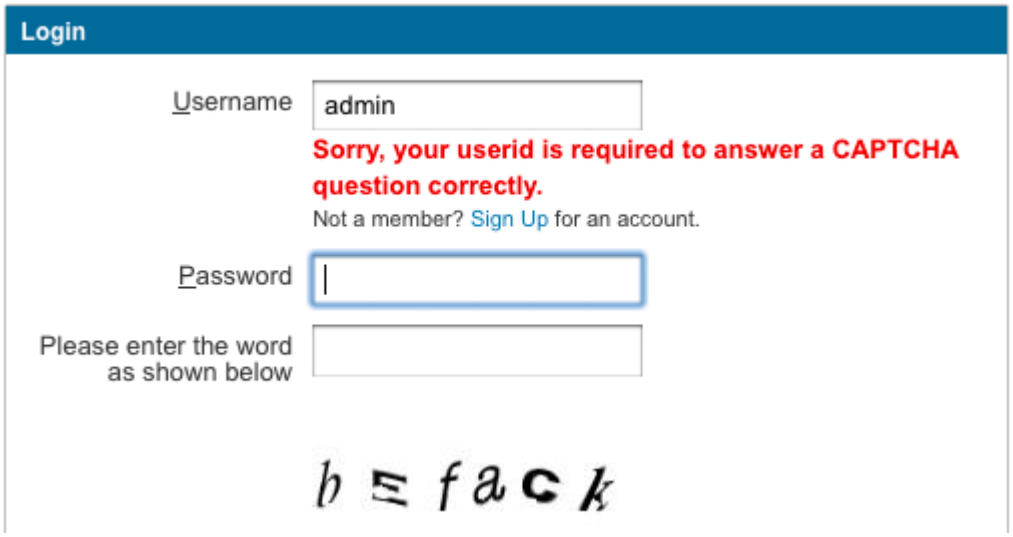
These patches protect JIRA from recently detected privilege escalation and XSS vulnerabilities.

3. Protect Against Brute Force Attack

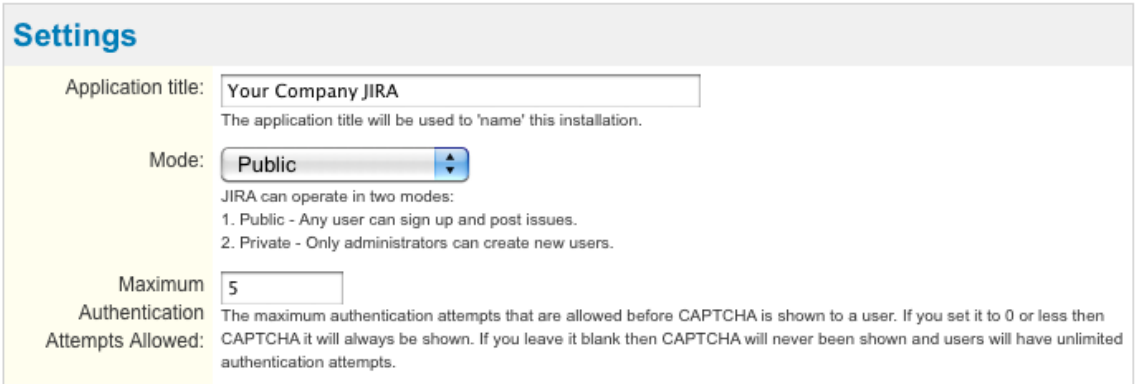
You can also actively protect your systems against repeated unsuccessful login attempts, known as "brute force" login attacks.

3.1 Upgrade to JIRA 4.1

JIRA 4.1 contains built-in protection for brute force attacks by displaying a CAPTCHA after a number of failed authentication attempts.



In JIRA 4.1.1 this option is enabled by default. (Please refer to the JIRA 4.1.1 Upgrade Guide for details.) To enable this protection in JIRA 4.1, log in as an administrator and navigate to Administration -> General Configuration and set the "Maximum Authentication Attempts Allowed" to a small number (e.g. 5).



For more details, see Configuring JIRA Options.

3.2 Enable Brute Force Login Protection on your Web Server

It is possible to also enable brute force login protection on your web server by detecting repeated authentication failures in application logs. Once repeated login failures have been detected, you can set up an automated system to ban access to your web server from that particular IP address.

For more information on how to configure an automated approach to this kind of login prevention, refer to Using Fail2Ban to limit login attempts.

4. Restrict Network Access to Administrative Sections of Applications

An Atlassian application's administration interface is a critical part of the application; anyone with access to it can potentially compromise not only the application instance but the entire machine. As well as limiting access to only users who really need it, and using strong passwords, you should consider limiting access to it to certain machines on the network.

For more information on how to implement Apache blocking rules to restrict access to administrative or sensitive actions in:

- JIRA, refer to Using Apache to Limit Access to the JIRA Administration Interface
- Confluence, refer to Using Apache to limit access to the Confluence administration interface

You can use a similar approach to protecting all Atlassian applications.

5. Restrict File System Access by the Application Server

The application server (e.g. Tomcat) runs as a process on the system. This process is run by a particular user and inherits the file system rights of that particular user. By restricting the directories that can be written to by the application server user, you can limit unnecessary exposure of your file system to the application.

For example, ensure that only the following directories can be written to by JIRA's application server:

- The following subdirectories of your JIRA Installation Directory for 'recommended' JIRA distributions (or for JIRA WAR distributions, the installation directory of the Apache Tomcat application running JIRA):
  - `logs`
  - `temp`
  - `work`
- Your JIRA Home Directory.

For detailed instructions, please see Tomcat security best practices.

6. Disable Jelly

Jelly is disabled in JIRA by default. If you need to use Jelly, you should enable it immediately prior to use and disable it immediately afterwards. See the JIRA Jelly Tags documentation for details.

JIRA Security Advisory 2010-06-18

In this advisory:

- XSS Vulnerabilities in URL Query Strings
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- JIRA Standalone Vulnerability with Session Cookies
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation

        - Fix
- Users without the 'JIRA Users' Permission can Login via Crowd Single Sign On
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix
- XSRF Vulnerability in 'Logout' Action
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix
- Security Vulnerabilities in FishEye Plugin
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix
- Security Vulnerabilities in Bamboo Plugin
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix

XSS Vulnerabilities in URL Query Strings

Severity

Atlassian rates these vulnerabilities as **critical**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed several cross-site scripting (XSS) vulnerabilities in JIRA, which may affect JIRA instances. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of JIRA.

- An attacker might take advantage of the vulnerability to steal other users' session cookies or other credentials, by sending the credentials back to the attacker's own web server. The attacker could potentially then gain control over the underlying JIRA system and/or the underlying operating system, based on the privileges of the user whose credentials had been stolen.
- The attacker's text and script might be displayed to other people viewing a JIRA page. This is potentially damaging to your company's reputation.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Vulnerability

Some values from JIRA URLs were not correctly HTML-escaped, potentially enabling an attacker to add scripts to another user's response.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix these vulnerabilities. Please see the 'Fix' section below.

Fix

These issues have been fixed in JIRA 4.1.2 and later. If you absolutely cannot upgrade, a patch that has been tested on JIRA 4.0.2 is available on the following holding bug: http://jira.atlassian.com/browse/JRA-21624

JIRA Standalone Vulnerability with Session Cookies

Severity

Atlassian rates this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and incorporated an enhancement in JIRA Standalone distributions in the handling of session cookies. This has security implications which are especially important for anyone running publicly accessible instances of JIRA.

- An attacker might take advantage of this vulnerability to steal other users' session cookies, by sending the session ID credentials contained within them back to the attacker's own web server. The attacker could potentially then gain control over the underlying JIRA system and/or the underlying operating system, based on the privileges of the user whose credentials had been stolen.

Vulnerability

If an attacker makes a successful XSS attack, this vulnerability could allow the attacker to use JavaScript to access the session ID contained within a session cookie.

Risk Mitigation

We recommend upgrading your JIRA installation to fix this vulnerability. Please see the 'Fix' section below.

Fix

Cookies are now set to 'HttpOnly' in the Standalone distributions of JIRA 4.1.2 and later. 'HttpOnly' session cookies dramatically reduce the likelihood of privilege escalation through XSS attack vectors. Therefore, please upgrade to this version of JIRA to mitigate this risk.

If you are running a JIRA EAR-WAR distribution or an earlier version of JIRA, please refer to the Preventing Security Attacks guide for information on how to implement 'HttpOnly' session cookies with specific examples for configuring Tomcat version 5.5.27+.

Users without the 'JIRA Users' Permission can Login via Crowd Single Sign On

Severity

Atlassian rates this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a vulnerability in JIRA, relating to login permission. This vulnerability has security implications and is especially important for anyone running publicly accessible instances of JIRA.

- A user might take advantage of the vulnerability to login to a JIRA instance which they are not authorised to view.

Vulnerability

This vulnerability only relates to JIRA instances that are connected to Atlassian Crowd and are using Crowd Single Sign On (SSO).

When JIRA is using the Crowd connector and Crowd SSO, a user who doesn't have the 'JIRA Users' permission can log in to JIRA using Crowd SSO.

Project-specific permissions are still enforced, so the user would only be able to see unsecured projects (that is, projects which 'Anyone' can view).

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix this vulnerability. Please see the 'Fix' section below.

Fix

This issue has been fixed in JIRA 4.1.2 and later. If you absolutely cannot upgrade, you can try replacing the `crowd-integration-client-1.6.1.jar` located in the <root-dir>/WEB-INF/lib directory with the newer version that comes with JIRA 4.1.2, namely `crowd-integration-client-2.0.4.jar`. Although this configuration has not been subjected to Atlassian's quality assurance processes, we believe the upgrade of that library should work and will fix this security bug. Customers who absolutely cannot upgrade to JIRA 4.1.2 who have any trouble with this should raise a support request at https://support.atlassian.com/ for help.

XSRF Vulnerability in 'Logout' Action

Severity

Atlassian rates this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed an XSRF (cross-site request forgery) vulnerability in JIRA, relating to the Logout action.

- An attacker might take advantage of the vulnerability to force logout. This could be used for a DOS (denial of service) attack.

You can read more about XSRF attacks at cgisecurity.

Vulnerability

An attacker could insert malicious text into an issue, which would force logout for any user who viewed that issue.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix this vulnerability. Please see the 'Fix' section below.

Fix

This issue has been fixed in JIRA 4.1.2 and later.

Security Vulnerabilities in FishEye Plugin

Severity

Atlassian rates these vulnerabilities as **critical**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

Please see the JIRA FishEye Plugin Security Advisory 2010-06-18 for details.

Vulnerability

These vulnerabilities relate to the JIRA FishEye Plugin, which is bundled with JIRA. Only JIRA instances where the JIRA FishEye Plugin is enabled are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation (or this plugin) to fix this vulnerability. Please see the 'Fix' section below.

Fix

These issues have been fixed in JIRA 4.1.2 and later. Upgrading to this version of JIRA will fix these vulnerabilities.

Alternatively, if you are running JIRA 4.1 or 4.1.1 and cannot upgrade JIRA to version 4.1.2 immediately, you can fix these vulnerabilities by upgrading the FishEye plugin. Otherwise, you can disable disable the JIRA FishEye plugin via the JIRA administration interface.

Security Vulnerabilities in Bamboo Plugin

Severity

Atlassian rates these vulnerabilities as **critical**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

Please see the JIRA Bamboo Plugin Security Advisory 2010-06-18 for details.

Vulnerability

These vulnerabilities relate to the JIRA Bamboo Plugin, which is bundled with JIRA. Only JIRA instances where the JIRA Bamboo Plugin is enabled are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation (or this plugin) to fix this vulnerability. Please see the 'Fix' section below.

Fix

These issues have been fixed in JIRA 4.1.2 and later. Upgrading to this version of JIRA will fix these vulnerabilities.

Alternatively, if you are running a version of JIRA from 4.0 to 4.1.1 (inclusive) and cannot upgrade JIRA to version 4.1.2 immediately, you can fix these vulnerabilities by upgrading the Bamboo plugin. Otherwise, you can disable disable the JIRA Bamboo plugin via the JIRA administration interface.

JIRA Security Advisory 2010-12-06

In this advisory:

- XSS Vulnerabilities in URL Query Strings
  - Severity
  - Risk Assessment
  - Vulnerability

        - Risk Mitigation
        - Fix
- XSRF Vulnerabilities
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix
- Vulnerability in Secure Tokens
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix
- Vulnerability in Component Data
    - Severity
    - Risk Assessment
    - Vulnerability
    - Risk Mitigation
    - Fix

XSS Vulnerabilities in URL Query Strings

Severity

Atlassian rates these vulnerabilities as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a number of cross-site scripting (XSS) vulnerabilities which may affect JIRA instances. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of JIRA. XSS vulnerabilities allow an attacker to embed their own JavaScript into a JIRA page. You can read more about XSS attacks at cgisecurity, the Web Application Security Consortium and other places on the web.

Vulnerability

Some values from JIRA URLs were being injected directly into JavaScript, potentially enabling an attacker to add scripts to another user's response.

All versions of JIRA prior to 4.2.1 are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix these vulnerabilities. Please see the 'Fix' section below.

Fix

These issues have been fixed in JIRA 4.2.1 and later, and are available as a patch for JIRA 3.13.5, 4.0.2 and 4.1.2 (please see JRA-22493).

XSRF Vulnerabilities

Severity

Atlassian rates this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed several cross-site request forgery (XSRF/CSRF) vulnerabilities in JIRA. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of JIRA.

- An attacker might take advantage of the vulnerability to fraudulently act on behalf of a legitimate user.

You can read more about XSRF/CSRF attacks at cgisecurity, wikipedia and other places on the web.

Vulnerability

Some JIRA administration screens did not have XSRF protection. A targetted attack on a vulnerable system could result in an attacker gaining access to user credentials, potentially giving them access to the JIRA data and system.

All versions of JIRA prior to 4.2.1 are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix these vulnerabilities. Please see the 'Fix' section below.

Fix

JIRA's XSRF protection has been extended to cover previously unprotected areas. The known XSRF issues have been fixed in JIRA 4.2.1 and later, and are available as a patch for JIRA 3.13.5, 4.0.2 and 4.1.2 (please see JRA-22493).

Vulnerability in Secure Tokens

Severity

Atlassian rates this vulnerability as **moderate**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a vulnerability relating to the creation of secure tokens, which are used in various authentication mechanisms. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of JIRA.

- Unauthorised users may be able to gain access to JIRA on behalf of a legitimate user.

Vulnerability

A highly skilled attacker could potentially forge a secure token, allowing them to impersonate a legitimate user.

All versions of JIRA prior to 4.2 are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix this vulnerabily. Please see the 'Fix' section below.

Fix

This issue has been fixed in JIRA 4.2 and later. The random number-generator that is used to generate tokens has been hardened.

Vulnerability in Component Data

Severity

Atlassian rates this vulnerability as **low**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a data vulnerability in JIRA. This vulnerability has security implications and is especially important for anyone running publicly accessible instances of JIRA.

- Unauthorised users may be able to view a list of components defined in your JIRA system.

Vulnerability

Component data could be view by unauthorised users.

All versions of JIRA prior to 4.2 are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix this vulnerabily. Please see the 'Fix' section below.

Fix

This issue has been fixed in JIRA 4.2 and later.

JIRA Security Advisory 2011-02-21

This advisory announces a security vulnerability that has been found in all versions of JIRA prior to 4.2.2 and fixed in 4.2.2 and later versions. Enterprise Hosted customers should request an upgrade by filing a ticket at http://support.atlassian.com. JIRA Studio is not vulnerable to any

of the issues described in this advisory.

Atlassian is committed to improving product security. The vulnerability listed in this advisory has been discovered by Atlassian, unless noted otherwise. The reporter may also have requested that we do not credit them.

If you have questions or concerns regarding this advisory, please raise a support request at http://support.atlassian.com/.

In this advisory:

- Parameter-Based Redirection Vulnerability
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
  - Patches

Parameter-Based Redirection Vulnerability

Severity

Atlassian rates this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low. This vulnerability is **not** critical.

This is an independent assessment and you should evaluate its applicability to your own IT environment.

Risk Assessment

Parameter-based redirection vulnerabilities allow an attacker to craft a JIRA URL in such a way that a user clicking on this URL will be redirected to a different web site. This can be used for phishing.

You can read more about link manipulation attacks at Wikipedia, and about phishing at Fraud.org and other places on the web.

Vulnerability

Some actions in JIRA redirect users to a new page after the action has been completed. It was possible to hand-craft an URL that would redirect to a site outside the current instance of JIRA. Starting with JIRA 4.2.2 all such redirections are limited to pages inside the current instance of JIRA.

All versions of JIRA prior to 4.2.2 are affected.

Risk Mitigation

We recommend upgrading your JIRA installation to fix this vulnerability. Please see the 'Fix' section below.

Fix

These issues have been fixed in JIRA 4.2.2 and later.

Patches

We have created a patch for the latest maintenance release 4.1.2 of JIRA for this vulnerability.

Please note that we have released a number of advisories about JIRA recently. We recommend that you review them and upgrade to the most recent release of the product or apply external security controls if you cannot. Most of the disclosed vulnerabilities are not critical and often present less risk when used in a corporate environment with no access from the Internet.

We usually provide patches only for vulnerabilities of critical severity, as an interim solution until you can upgrade. You should not expect that you can continue patching your system instead of upgrading. Our patches are often non-cumulative – we do not recommend that you apply multiple patches from different advisories on top of each other, but strongly recommend to upgrade to the most recent version regularly.

We recommend patching only when you can neither upgrade nor apply external security controls.

Supported JIRA Version	Issue Tracking	File Name	Downloadable Patch
4.1.2	JRA-23842	patch-JRA-23842-4.1.2-a.zip	Download

Instructions on how to apply the patch are included in the zip file

JIRA Security Advisory 2011-09-27

This advisory announces a number of security vulnerabilities that we have found in versions 4.2.x - 4.3.x of JIRA and fixed in version 4.4 of JIRA. You need to upgrade your existing JIRA installations to fix these vulnerabilities. Enterprise Hosted customers should request an upgrade by filing a ticket at http://support.atlassian.com, in the 'Enterprise Hosting Project'. JIRA Studio is not vulnerable to any of the issues described in this advisory.

Atlassian is committed to improving product security. The vulnerabilities listed in this advisory have been discovered by Atlassian, unless noted otherwise. The reporter may also have requested that we do not credit them.

If you have questions or concerns regarding this advisory, please raise a support request at http://support.atlassian.com/.

In this advisory:

- XSS Vulnerabilities in Labelling and Issue Linking
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
  - Patches
- XSS Vulnerability in Administration Interface of JIRA Bamboo Plugin
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
  - Patches

XSS Vulnerabilities in Labelling and Issue Linking

Severity

Atlassian rates the severity level of this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank the severity as critical, high, moderate or low. This vulnerability is not critical.

This is an independent assessment and you should evaluate its applicability to your own environment.

Risk Assessment

We have identified and fixed several cross-site scripting (XSS) vulnerabilities which may affect JIRA instances. XSS vulnerabilities potentially allow an attacker to embed their own JavaScript into a JIRA page. The attacker needs to have a valid user account in order to exploit this vulnerability.

You can read more about XSS attacks at cgisecurity, the Web Application Security Consortium and other places on the web.

Vulnerability

Issue linking:

- The way issue summaries were rendered when displaying issue links allows arbitrary JavaScript execution.
- Versions of JIRA 4.2.x to 4.3.x prior to 4.4 are affected.

Labelling:

- Certain issue labels could be created containing JavaScript, which then could be rendered on other pages.
- Versions of JIRA 4.2.x to 4.3.x prior to 4.4 are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix these vulnerabilities. Please see the 'Fix' section below.

Fix

These vulnerabilities have been fixed in JIRA 4.4 and later versions.

For a full description of the latest version of JIRA, see the release notes. You can download the latest version of JIRA from the download centre.

If you cannot upgrade to the latest version of JIRA, you can temporarily patch your existing installation of JIRA 4.3.x or JIRA 4.2.x using the patches listed below. We strongly recommend upgrading and not patching.

Patches

If you are running JIRA 4.3.x, you can apply the following patch to fix these vulnerabilities.

Vulnerability	Patch	Patch File Name	Instructions
Linking and Labelling	Attached to issue JRA-24773	JRA-24773-4.3.4-patch.zip	JRA-24773-4.3.4-patch-instructions.txt

If you are running JIRA 4.2.x, you can apply the following patch to fix these vulnerabilities.

Vulnerability	Patch	Patch File Name	Instructions
Linking and Labelling	Attached to issue JRA-24773	JRA-24773-4.2.4-patch.zip	JRA-24773-4.2.4-patch-instructions.txt

XSS Vulnerability in Administration Interface of JIRA Bamboo Plugin

Severity

Atlassian rates the severity level of this vulnerability as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank the severity as critical, high, moderate or low. This vulnerability is not critical.

This is an independent assessment and you should evaluate its applicability to your own environment.

Risk Assessment

We have identified and fixed a cross-site scripting (XSS) vulnerability which may affect JIRA instances. XSS vulnerabilities potentially allow an attacker to embed their own JavaScript into a JIRA page. The attacker does not need a valid user account in order to exploit this vulnerability

You can read more about XSS attacks at cgisecurity, the Web Application Security Consortium and other places on the web.

Vulnerability

JIRA administration interface (Bamboo plugin):

- There is a non-persistent XSS vector in the JIRA administration interface related to managing JIRA Bamboo settings.
- Versions of JIRA 4.3.x are affected.

Risk Mitigation

We strongly recommend upgrading your JIRA installation to fix this vulnerability. Please see the 'Fix' section below.

Fix

This vulnerability has been fixed in JIRA 4.4 and later versions.

For a full description of the latest version of JIRA, see the release notes. You can download the latest version of JIRA from the download centre.

If you cannot upgrade to the latest version of JIRA, you can upgrade only the Bamboo Plugin in your existing installation of JIRA 4.3.x or JIRA 4.2.x using the patches listed below. We strongly recommend upgrading full JIRA instance instead of a single plugin.

Patches

If you are running JIRA 4.3.x, use the plugin manager to upgrade the Bamboo plugin to a version equal to or greater than that specified in the file name below. Both Bamboo Plugin 4.2.x and 4.3.x support JIRA 4.3.x, see the compatibility matrix at Plugin Exchange.

Vulnerability	Plugin	Plugin version	Instructions
JIRA Bamboo Plugin	Plugin Exchange	4.2.1 or 4.3.3	Updating a JIRA plugin

If you are running JIRA 4.2.x, use the plugin manager to upgrade the Bamboo plugin to a version equal to or greater than that specified in the file name below. The vulnerability is not exploitable in JIRA 4.2.x, but we recommend upgrading the plugin anyway.

Vulnerability	Patch	Plugin version	Instructions
JIRA Bamboo Plugin	Plugin Exchange	4.1.5	Updating a JIRA plugin

Acknowledgement

Our thanks to **Dave B**, who reported one of the vulnerabilities in this advisory. We fully support the reporting of vulnerabilities and we appreciate it when people work with us to identify and solve the problem.

JIRA Resources

Resources for Evaluators

- Free Trial
- Feature Tour
- JIRA Sample Files
- JIRA FAQ

Resources for Administrators

- JIRA Knowledge Base
- Tips of the Trade
- Tips via Twitter
- Guide to Installing an Atlassian Integrated Suite
- The big list of Atlassian gadgets

Resources for Developers

- JIRA developer documentation: Atlassian Developers
- Reference documentation: Latest and earlier releases.
- Developer discussion forum: Atlassian Answers
- Real-time crash reports and user feedback for your iOS apps: JIRA Mobile Connect Developer Documentation

Downloadable Documentation

- JIRA documentation in PDF, HTML or XML formats
- Setting Up Local Online JIRA Documentation

Plugins

- Atlassian Plugin Exchange

Support

- Atlassian Support
- Support Policies

Training

- Atlassian Training

Mailing Lists

- Visit http://my.atlassian.com to sign up for mailing lists relating to Atlassian products, such as technical alerts, product announcements and developer updates.

Forums

- Atlassian Answers for JIRA
- Atlassian Answers for JIRA Development

Feature Requests

- Issue Tracker and Feature Requests for JIRA
- Policy for Implementing New Features

IDE Connectors

- Use the Atlassian Connector for Eclipse or the Atlassian Connector for IntelliJ IDEA to work with your JIRA issues right there in your development environment. Do you use Bamboo, Crucible or FishEye too? With the connector you can manage your builds and code reviews within your IDE, or move quickly between the IDE and a FishEye view of your source repository. **Hint:** The Atlassian IDE Connectors are free.

Support Policies

Welcome to the support policies index page. Here, you'll find information about how Atlassian Support can help you and how to get in touch with our helpful support engineers. Please choose the relevant page below to find out more.

- Bug Fixing Policy
- How to Report a Security Issue
- New Features Policy
- Patch Policy
- Security Advisory Publishing Policy
- Security Patch Policy
- Severity Levels for Security Issues

To request support from Atlassian, please raise a support issue in our online support system. To do this, see Getting Help. Our friendly support engineers will get right back to you with an answer.

Bug Fixing Policy

Summary

- Atlassian Support will help with workarounds and bug reporting.
- Critical bugs will generally be fixed in the next maintenance release.
- Non critical bugs will be scheduled according to a variety of considerations.

Raising a Bug Report

Atlassian Support is eager and happy to help verify bugs — we take pride in it! Please open a support request in our support system providing as much information as possible about how to replicate the problem you are experiencing. We will replicate the bug to verify, then lodge the report for you. We'll also try to construct workarounds if they're possible.

Customers and plugin developers are also welcome to open bug reports on our issue tracking systems directly. Use http://jira.atlassian.com for the stand-alone products and http://studio.atlassian.com for JIRA Studio and Atlassian OnDemand.

When raising a new bug, you should rate the priority of a bug according to our JIRA usage guidelines. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

How Atlassian Approaches Bug Fixing

Maintenance (bug fix) releases come out more frequently than major releases and attempt to target the most critical bugs affecting our customers. The notation for a maintenance release is the final number in the version (ie the 1 in 3.0.1).

If a bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions) then it will be fixed in the next maintenance release provided that:

- The fix is technically feasible (i.e. it doesn't require a major architectural change).
- It does not impact the quality or integrity of a product.

For non-critical bugs, the developer assigned to fixing bugs prioritises the non-critical bug according to these factors:

- How many of our supported configurations are affected by the problem.
- Whether there is an effective workaround or patch.
- How difficult the issue is to fix.
- Whether many bugs in one area can be fixed at one time.

The developers responsible for bug fixing also monitor comments on existing bugs and new bugs submitted in JIRA, so you can provide feedback in this way. We give high priority consideration to security issues.

When considering the priority of a non-critical bug we try to determine a 'value' score for a bug which takes into account the severity of the bug from the customer's perspective, how prevalent the bug is and whether roadmap features may render the bug obsolete. We combine this with a complexity score (i.e. how difficult the bug is). These two dimensions are used when developers self serve from the bug pile.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

How to Report a Security Issue

Finding and Reporting a Security Vulnerability

If you find a security bug in the product, please open an issue on http://jira.atlassian.com in the relevant project.

- Set the priority of the bug to 'Blocker'.
- Provide as much information on reproducing the bug as possible.
- Set the security level of the bug to 'Developer and Reporters only'.

All communication about the vulnerability should be performed through JIRA, so that Atlassian can keep track of the issue and get a patch out as soon as possible.

> If you discover a security vulnerability, please attempt to create a test case that proves this vulnerability locally before opening either a bug or a support issue. When creating an issue, please include information on how the vulnerability can be reproduced; see our Bug Fixing Policy for general bug reporting guidelines. We will prioritise fixing the reported vulnerability if your report has information on how the vulnerability can be exploited.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

New Features Policy

Summary

- We encourage and display customer comments and votes openly in our issue tracking systems, http://jira.atlassian.com and http://studio.atlassian.com.
- We do not publish roadmaps.
- Product Managers review our most popular voted issues on a regular basis.
- We schedule features based on a variety of factors.
- Our Atlassian Bug Fixing Policy is distinct from our Feature Request process.
- Atlassian provides consistent updates on the top 20 feature/improvement requests (in our issue tracker systems).

How to Track what Features are Being Implemented

When a new feature or improvement is scheduled, the 'fix-for' version will be indicated in the JIRA issue. This happens for the upcoming release only. We maintain roadmaps for more distant releases internally, but because these roadmaps are often pre-empted by changing customer demands, we do not publish them.

How Atlassian Chooses What to Implement

In every major release we *aim* to implement highly requested features, but it is not the only determining factor. Other factors include:

- **Direct feedback** from face to face meetings with customers, and through our support and sales channels.
- **Availability of staff** to implement features.
- **Impact** of the proposed changes on the application and its underlying architecture.
- How **well defined** the requested feature is (some issues gain in popularity rapidly, allowing little time to plan their implementation).
- Our long-term **strategic vision** for the product.

How to Contribute to Feature Development

Influencing Atlassian's release cycle
We encourage our customers to vote on feature requests in JIRA. The current tally of votes is available online in our issue tracking systems, http://jira.atlassian.com and http://studio.atlassian.com. Find out if your improvement request already exists. If it does, please vote for it. If you do not find it, create a new feature or improvement request online.

Extending Atlassian Products
Atlassian products have powerful and flexible extension APIs. If you would like to see a particular feature implemented, it may be possible to develop the feature as a plugin. Documentation regarding the plugin APIs is available. Advice on extending either product may be available on the user mailing-lists, or at Atlassian Answers.

If you require significant customisations, you may wish to get in touch with our partners. They specialise in extending Atlassian products and can do this work for you. If you are interested, please contact us.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Patch Policy

Patch Policy

Atlassian will only provide software patches in extremely unusual circumstances. If a problem has been fixed in a newer release of the product, Atlassian will request that you upgrade your instance to fix the issue. If it is deemed necessary to provide a patch, a patch will be provided for the current release and the last maintenance release of the last major version (e.g. JIRA 4.2.4) only.

Patches are issued under the following conditions:

- The bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions).
- A patch is technically feasible (i.e., it doesn't require a major architectural change)
  OR
- The issue is a security issue, and falls under our Security Patch Policy.

Atlassian does not provide patches for non-critical bugs.

Provided that a patch does not impact the quality or integrity of a product, Atlassian will ensure that patches supplied to customers are added to the next maintenance release. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

Patches are generally attached to the relevant http://jira.atlassian.com issue.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Advisory Publishing Policy

Publication of Security Advisories

When a security vulnerability in an Atlassian product is discovered and resolved, Atlassian will inform customers through the following mechanisms:

- We will post a security advisory in the latest documentation of the affected product at the same time as releasing a fix for the vulnerability. This applies to all security advisories, including severity levels of critical, high, medium and low.
- We will send a copy of all security advisories to the **'Technical Alerts' mailing list** for the product concerned.
  Note: To manage your email subscriptions and ensure you are on this list, please go to my.atlassian.com and click 'Email Prefs' near the top right of the page.
- If the person who reported the vulnerability wants to publish an advisory through some other agency, such as CERT, we will assist in the production of that advisory and link to it from our own.

Early warning of critical security vulnerabilities:

- If the vulnerability is rated critical (see our criteria for setting severity levels) we will send an early warning to the 'Technical Alerts' mailing list approximately one week before releasing the fix. This early warning is in addition to the security advisory itself, described above.
- However, if the vulnerability is publicly known or being exploited, we will release the security advisory and patches as soon as possible, potentially without early warning.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Patch Policy

Product Security Patch Policy

Atlassian makes it a priority to ensure that customers' systems cannot be compromised by exploiting vulnerabilities in Atlassian products.

Scope

This page describes when and how we release security patches and security upgrades for our products. It does not describe the whole of disclosure process that we follow. It also excludes JIRA Studio, since JIRA Studio will always be patched by Atlassian without additional notifications.

Critical vulnerabilities

When a **Critical** security vulnerability is discovered by Atlassian or reported by a third party, Atlassian will do all of the following:

- Issue a new, fixed release for the current version of the affected product as soon as possible, usually in a few days.
- Issue a binary patch for the current release.
- Issue a binary patch for the latest maintenance release of the previous version of the product.

- Patches for older versions or releases normally will not be issued.

Patches will be attached to the relevant JIRA issue. You can use these patches as a "stop-gap" measure until you upgrade your installation in order to fully fix the vulnerability.

Non-critical vulnerabilities

When a security issue of a **High, Medium or Low** severity is discovered, Atlassian will do all of the following:

- Include the fix into the next scheduled release, both for the current and previous maintenance versions.
- Where practical, provide new versions of plugins or other components of the product that can be upgraded independently.

You should upgrade your installation in order to fix the vulnerability.

Other information

Severity level of vulnerabilities is calculated based on Severity Levels for Security Issues.

Visit our general Atlassian Patch Policy as well.

Examples

Example 1: A critical severity vulnerability is found in a (hypothetical current release) JIRA 5.3.2. The last bugfix release in 5.2.x branch was 5.2.3. In this case, a patch will be created for 5.3.2 and 5.2.3. In addition, new bugfix releases, 5.3.3 and 5.2.4, which are free from this vulnerability, will be created in a few days.

Example 2: A high or medium severity vulnerability is found in the same release as in the previous example. The fix will be included into the currently scheduled releases 5.3.3 and 5.2.4. Release schedule will not be brought forward and no patches will be issued. If the vulnerability is in a plugin module, then a plugin upgrade package may still be supplied.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Severity Levels for Security Issues

Severity Levels

Atlassian security advisories include a severity level. This severity level is based on our self-calculated CVSS score for each specific vulnerability. CVSS is an industry standard vulnerability metric. You can learn more about CVSS at FIRST.org web site.

CVSS scores are mapped into the following severity ratings:

- Critical
- High
- Medium
- Low

An approximate mapping guideline is as follows:

CVSS score range	Severity in advisory
0 – 2.9	Low
3 – 5.9	Medium
6.0 – 7.9	High
8.0 – 10.0	Critical

Below is a summary of the factors which illustrate types of vulnerabilities usually resulting in a specific severity level. Please keep in mind that this rating does not take into account details of your installation.

Severity Level: Critical

Vulnerabilities that score in the critical range usually have the following characteristics:

- Exploitation of the vulnerability results in root-level compromise of servers or infrastructure devices.
- The information required in order to exploit the vulnerability, such as example code, is widely available to attackers.
- Exploitation is usually straightforward, in the sense that the attacker does not need any special authentication credentials or knowledge about individual victims, and does not need to persuade a target user, for example via social engineering, into performing

any special functions.

For critical vulnerabilities, is advised that you patch or upgrade as soon as possible, unless you have other mitigating measures in place. For example, if your installation is not accessible from the Internet, this may be a mitigating factor.

Severity Level: High

Vulnerabilities that score in the high range usually have the following characteristics:

- The vulnerability is difficult to exploit.
- Exploitation does not result in elevated privileges.
- Exploitation does not result in a significant data loss.

Severity Level: Medium

Vulnerabilities that score in the medium range usually have the following characteristics:

- Denial of service vulnerabilities that are difficult to set up.
- Exploits that require an attacker to reside on the same local network as the victim.
- Vulnerabilities that affect only nonstandard configurations or obscure applications.
- Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics.
- Vulnerabilities where exploitation provides only very limited access.

Severity Level: Low

Vulnerabilities in the low range typically have very little impact on an organisation's business. Exploitation of such vulnerabilities usually requires local or physical system access.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Local JIRA documentation

On this page:

- Why would I set up local online documentation?
- How to set up local online documentation for JIRA 4.0.x and later
  - Additional documentation spaces required
    - Why you need the additional documentation spaces
    - Determining the Version Required
    - List of Spaces Required
- How to set up local online documentation for JIRA 3.13.x and earlier
- Local field documentation

Why would I set up local online documentation?

You may wish to run the documentation locally, and have JIRA link to it. There are a few reasons you may wish to do this:

- JIRA's interface contains links ( ) to help pages, some to pages within JIRA, but many to the online documentation on www.atlassian.com. For deployments in environments without an internet connection, a local copy of the documentation is desirable.
- If you have customised JIRA, you may wish to update the documentation to reflect your changes, or add new pages.
- You can change the look and feel of the documentation to integrate into your company's intranet.

How to set up local online documentation for JIRA 4.0.x and later

1. Install Atlassian Confluence. (If you don't already have Confluence, ask for a free Evaluation License. You can use 'Anonymous' access to allow your users to view the documentation.)
2. Download the JIRA Documentation's XML source. Note that the Confluence version of the XML source needs to be the same major Confluence version as your local Confluence site.
3. Import the XML file into your Confluence site. (Note: if there is already a 'JIRA' space in your Confluence site, it will be overwritten.) For detailed instructions, see the Confluence documentation on Restoring a Space.
4. If you are importing the documentation for JIRA 4.1 or later, you will need to remove or adjust the customised header, footer and left-hand navigation bar in your new space.
   Explanation: When you create your new space from our XML source code, the space will inherit the Confluence 'Documentation' theme. The XML source code also includes the customisations we have made to the header, footer and left-hand navigation bar. These customisations include references to our Atlassian Documentation space. Since your Confluence site does not have that space, you will see errors like this in the left-hand navigation bar, header and footer in your new space:

   ```
   Unable to render {include} Couldn't find a space with key: ALLDOC
   ```

   To fix these errors, take one of the following steps:
   - Customise the navigation, header and footer sections to suit your Confluence site or environment. See our documentation

on configuring the Documentation theme.
- Or restore the default left-hand navigation bar, by removing all content from the navigation, header and footer sections and selecting the '**Page Tree**' check box. See our documentation on configuring the Documentation theme.
- Or change the theme of your space to the Confluence default theme or another theme of your choice.

5. Download the XML source code for the additional documentation spaces listed below and import them into your Confluence site too.
6. *(Optional)* If you want JIRA's help links ( ) to point to your local documentation, you will need to:
   a. edit JIRA's `/WEB-INF/classes/help-paths.properties` file and change the `url-prefix` line so that it points to the 'JIRA' space in your local Confluence site, e.g.:

```
url-prefix=http://confluence.mycompany.com/display/JIRA/
```

   b. restart JIRA.

Additional documentation spaces required

Why you need the additional documentation spaces

The JIRA documentation shares some content with other Atlassian products, such as Confluence. For the sake of efficiency, we reuse the same content across documentation spaces. You will notice that some of our pages contain an {include} macro that draws in content from another space.

For example, the following macro includes content from the Application Links (APPLINKS) space into the JIRA documentation space:

```
{include:APPLINKS:_securityTrustedApps}
```

You will need to import those documentation spaces into your Confluence site, to ensure that the reused content is accessible in your JIRA documentation.

Determining the Version Required

We supply different versions of the documentation, for each version of the software or plugin concerned. To see which version you need, take a look at the space key in the {include} macro concerned.

- If the space key has a number at the end, that number indicates the version. For example, `012` means version 1.2, and `011` means version 1.1.
- If the space key does not include a number, you need the latest version of the documentation.

Here is an example of an include macro that requires version 1.2 of the Application Links documentation:

```
{include:APPLINKS012:_securityTrustedApps}
```

This example requires the **latest** version of the Application Links documentation:

```
{include:APPLINKS:_securityTrustedApps}
```

List of Spaces Required

Retrieve the relevant version of the XML backups from these pages:

- Application Links
- Universal Plugin Manager
- User Management

How to set up local online documentation for JIRA 3.13.x and earlier

JIRA licensees can download the XML source for the documentation.

To build JIRA's docs locally:

1. Download Apache Forrest 0.5.1 (zip, tar.gz), used to render the docs.
2. Download the JIRA Documentation's XML source (6.2Mb). Note: the download is restricted to JIRA license holders.
3. Follow the JIRA_DOCUMENTATION.txt instructions in the package.

Local field documentation

If you just want to document usage of a custom field, please see Creating Help for a Custom Field.

JIRA FAQ

- What does JIRA mean?
- How is JIRA pronounced?
- Licensing

For technical articles please see the JIRA Knowledge Base.

What does JIRA mean?

Like all good names in the software industry, it started as an in-house code name.

We originally used Bugzilla for bug tracking and the developers in the office started calling it by the Japanese name for Godzilla, **Gojira** (the original black-and-white Japanese Godzilla films are also office favourites). As we developed our own bug tracker, and then it became an issue tracker, the name stuck, but the **Go** got dropped - hence JIRA!

Further investigation into the name has revealed that **Gorira** is Japanese for "gorilla", whilst **Kujira** is Japanese for "whale". So **Gojira** is roughly translated to mean "gorilla the size of a whale"! (Thanks to yusuke_arclamp — Oct 2002)

For those who care - it sounds best if you yell it loudly, as though charging into battle. C'mon - try it!

Related

1. Filed your request in JIRA. What does that mean?
2. Talking that JIRA-slang language, or jiralang, if I may

How is JIRA pronounced?

We pronounce it 'JEEra', based on the pronunciation of 'Kujira' (see What does JIRA mean?)

Licensing

Please see our licensing FAQ

Related

JIRA 4.0 introduced new licensing — please see Licensing Changes.

How JIRA Documentation Updates are Published

Documentation is Organised by Major Product Version

As you can see from this web site, Atlassian's Technical Writers use Confluence to prepare and publish documentation for Atlassian's products.

The documentation for each major version of JIRA is housed in its own Confluence space. Examples of major versions of JIRA are '4.3.x', '4.4.x' or '5.0.x', where 'x' represents a minor version/release number. Any documentation relating to specific features in minor versions/releases of a major JIRA version are also housed in the documentation space for that major version.

While the names of spaces can be changed in Confluence, each space is identified by a unique 'space key', which remains static.

✅ When viewing a Confluence page, the 'space key' of the space to which the page belongs can be found immediately after the `display/` part of that Confluence page's URL.

Traditional Documentation Publishing Process

⚠️ This process is no longer being employed to publish JIRA documentation updates. All version numbers mentioned in this section are used solely for demonstrating this process and do not reflect our current online JIRA documentation.

Documentation for the *latest official major version* of JIRA (for example, JIRA 4.2.x) is housed in the Confluence space on this site with the 'JIRA' space key.

Documentation updates for the *next major version* of JIRA (for example, JIRA 4.3.0) are drafted in this same Confluence space (with the 'JIRA' space key). Each draft page created is hidden from public view by applying page viewing restrictions.

Just prior to the release of the *next major version* of JIRA, we begin 'branching' the documentation. This involves the following steps:

1. We create a copy of the 'JIRA' space on this site using Confluence's 'Copy Space' plugin. The copied space is given a space key which reflects the *latest official major version* of JIRA, based on the format 'JIRA0XY' (where X and Y reflect the first two numbers that constitute this major version of JIRA).
   Documentation in the new 'JIRA0XY' space, which is initially hidden from public view via space permissions, will apply to the *previous major version* of JIRA once the *next major version* of JIRA is officially released.
2. Immediately after the *next major version* of JIRA is officially released, we perform the following sub-steps:
   a. The 'JIRA' space is renamed to reflect the **new** *latest official major version* of JIRA, for example, 'JIRA 4.3'.
   b. The 'JIRA0XY' space (for example, 'JIRA042') is revealed to the public by adjusting space permissions and we ensure it is renamed to reflect the *previous major version* of JIRA, for example, 'JIRA 4.2'.
   c. We publish the drafted content for what is now the **new** *latest official major version* of JIRA in the 'JIRA' space.

For more information about this process, please refer to 'From draft to published document' and 'Documentation Release Management' in our blog series on 'Technical Writing in a Wiki'.

Recent Documentation Publishing Process

Since the release of JIRA 4.3.0, we have modified the traditional documentation publishing process above.

Instead of waiting until the *next major version* of JIRA is about to be officially released, we begin the 'branching' steps much earlier during development of the *next major version* of JIRA.

For JIRA 4.3.0, we performed the branching steps just prior to the release of JIRA 4.3 Beta 1. For sub-step 2a of the branching process, the 'JIRA' space was renamed 'JIRA 4.3 Beta'.

For JIRA 4.4.0, we performed this branching even earlier — just prior to the release of JIRA 4.4 EAP 2. For sub-step 2a of the branching process, the 'JIRA' space was renamed 'JIRA 4.4 EAP'. However, when JIRA 4.4 Beta 1 was released, we renamed this space to 'JIRA 4.4 Beta'.

Implications of this New Process

Branching our documentation early during early development of the *next major version* of JIRA has the following implications:

- A dedicated documentation space for the *latest official major version* of JIRA becomes available. For instance, the 'JIRA 4.3' documentation space (with space key 'JIRA043') became available when JIRA 4.4 EAP 2 was released.
- The 'JIRA' space reflects JIRA documentation for the *next major version* of JIRA. For instance, the 'JIRA 4.4 EAP' documentation (with space key 'JIRA') became available when JIRA 4.4 EAP 2 was released.

Customers searching our documentation via Google may find that their Google search results selectively choose pages in the 'JIRA' space for the *next major version* of JIRA, as mentioned in JRA-24805.

Why have we Adopted this New Process?

Branching our documentation early offers the following important benefits:

- Customers who wish to try out early versions of the *next major version* of JIRA, as well as developers who wish to update their JIRA plugins for compatibility with that JIRA version will have access to documentation for that JIRA version (if available).
- Publishing documentation updates is easier since we work with live content in documentation spaces with 'EAP' or 'Beta' in their name, rather than working on draft pages which are hidden from public view.

If you reach a JIRA documentation page for the *next major version* of JIRA (for example, via a Google search), most headers of these pages have a link that will lead you to the equivalent page for *latest official major version* of JIRA.

For more information about our reasons for adopting this new process, please refer to our response to JRA-24805.

How to Reference JIRA Documentation from External Sources

If you wish to reference our JIRA documentation from an external source, you may wish to refer to version-specific pages of our documentation, by ensuring that you use the appropriate space key in your URL. For example:

- http://confluence.atlassian.com/display/JIRA042/Managing+Groups (for JIRA 4.2.x)
- http://confluence.atlassian.com/display/JIRA043/Managing+Groups (for JIRA 4.3.x)

Sometimes, if the functionality of a particular aspect of JIRA has changed, we may need to change the name of a page. For example, due to the significant changes in 'advanced JIRA configuration' in JIRA 4.4, we've had to change the name of:

- http://confluence.atlassian.com/display/JIRA043/Advanced+JIRA+configuration+with+jira-application.properties (for JIRA 4.3.x)
  to
- http://confluence.atlassian.com/display/JIRA/Advanced+JIRA+configuration (for JIRA 4.4.x)

We almost never change the names of pages in earlier versions of documentation (unless of course they are incorrect).

JIRA Administrators FAQ

For more articles, please see the JIRA Knowledge Base.

> *The information on the FAQs linked below may not apply to **JIRA OnDemand.***

JIRA Administrators FAQ

Performance FAQ

- Is your JIRA Running Slowly

Usage FAQ

- Modifying the JIRA Footer
- 'Road Map', 'Change Log ', and 'Versions' Project Tabs Are Not Visible
- Why Do Linked Issues in JIRA Appear with a Strike-Through
- How to Enable the FishEye Plugin from the Plugin Administration Screen
- How to Add the Priority Field Into the Email Subject
- How to Change the Number of Rows Allowed in the Text Type Custom Field Renderer
- How to Make a Federated JIRA Instance
- How to Remove 'Unknown' Option from 'Component' and 'Fix Versions'
- Automatic Escalation of issues
- How to Activate Header Row for Subtask List in Issue Detail View
- How to Limit the Number of Characters Entered in a Summary Field
- JIRA's Timestamp Doesn't Match the System Time
- How to Attach a File During Issue Creation
- How to Remove Duplicate Entries for Names and Groups
- How to Export Users to CSV from JIRA
- How Come JIRA does Not Show Direct SQL Data Modifications
- How to get JIRA Pages to Render when URL Contains an Underscore
- Certain Action Words Show Incorrect Language Translations
- How to disable or enable the GOT FEEDBACK button
- Changing the number of users synchronised from LDAP to JIRA
- How do I reduce my user count in JIRA
- Adding custom content to the front page
- Allow editing of Closed Issues
- Allowing users to create issues anonymously
- Anonymising JIRA Data
- Appending Email Addresses to Comments Made by Anonymous Users when Using a Mail Handler
- Asking for an attachment on the Create Issue page
- Automatically Populate Timezone from a Created Issue
- Can I store customer details, like company, address and contact information, in JIRA?
- Change JIRA Browser Icon
- Changing Custom Field Types
- Changing Templates Used by Export to Excel from the Issue Navigator
- Changing the default attachment size limit
- Changing the default session timeout
- Changing the Default Tab Panel from Comments to All
- Changing the Due Date Input Format
- Changing the Project Key
- Changing the Project Key Format
- Changing the Size and Content of the Components Select List
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size of the Text Area Custom Field
- Changing the Temporary Directory
- Changing Usernames in JIRA
- Configuring project specific security
    - Controlling project visibility
    - Using Project Level Security with Project Roles — This tutorial provides a step-by-step guide for creating project roles and using them in an issue security scheme. We recommend creating a test project and two test users for this tutorial.
    - Using Project Level Security with User Groups
- Connecting to SSL services
- Creating a Custom Workflow
- Creating an Unassigned Issue
- Creating Issues via direct HTML links
- Current Reporter Browse Project Permission
- CVS ssh Jira Integration
- Disabling Form Token Checking
- Displaying a Field Based on Another Field Selection
- Editing a custom field option
- Escalating issues (or sending email notifications) when the set turnaround time is exceeded
- Field Layout Schemes in JIRA 3.x

- Fields Allowing Custom HTML or JavaScript
- Finding the Id for Issue Types
- Group Name Guidelines for JIRA
- How can I control the editing of issue fields via workflow?
    - Using a Workflow to control edit of an issue by changing Workflow XML
    - Using Templates to control edit of an issue
- How do I assign issues to multiple users
- How do I delete a user account?
- How do I disable Firebug for JIRA?
- How Do I Use an SSL Certificate Generated Using openssl?
- How the CreateOrCommentHandler works?
- How to change Multi Select Custom field size using script
- How to change the location of stdout and stderr logs
- How to clear the resolution field when the issue is reopened
- How to configure sub-task to have a specific screen?
- How to convert types using Jelly
- How to create a download link to a file
- How to disable the Resolve issue screen while resolving issues
- How to display a different format for the Number customfield
- How to ensure the Road Map tab is visible
- How to Get Unicode 'non-ASCII' Characters in HTTPS URL to Appear Correctly
- How to have long component version names display properly in the Issue Navigator
- How to Remove 'NONE' from a Select List Custom Field
- How to Rename the 'Priority' Field in the Issue Navigator
- How to re-order statuses
- How to re-order the list of issue operation in an issue
- How to resize Free Text Field customfield
- How to resize the 'Components' and 'Affects Versions' fields in the Issue Navigator
- How to Restrict the Subversion Commits Tab to Selected Projects or Users
- How to search by number range in the Issue Navigator
- How to show a transition only when the Assignee is different from the Current User
- Importing data
- Importing user from LDAP
    - Neat JIRA LDAP tricks — Gianugo has an interesting blog up about how to take the pain out of migrating users from LDAP
- JIRA as a Support System
    - Jelly Escalation
    - Simple Escalation
- Letting customers only create issues — This page describes a minor JIRA modification which redirects users to an arbitrary page after creating issues (and potentially other operations). It is mainly of interest to JIRA Professional and Standard users.
- Linking to local file under Firefox
- Login problems
- Mail error - Unable to relay
- Making JIRA login case insensitive for JIRA 3.13.x
- Outward Link Description and Inward Link Description
- Parsing utf-7 emails
- Project-specific email templates
- QuickSearch guesses the issue key prefix (sometimes)
- Receiving a Daily Summary of Updated Issues
- Recieving Notification for Select Issues or Updates
- Removing Commas for Values Held in Number Field Custom Field Type
- Removing invalid characters from XML backups — In older versions of JIRA it was possible to cut & paste text containing control characters into JIRA issue fields. This causes problems, because JIRA's backup format is XML, and XML does not allow for the storage of most control characters .
- Removing NONE from the Issue Security Drop-Down List
- Re-order workflow transactions
- Resolved issues appearing in Open issues filters
- Restricting the Visibility of Worklog on an Issue
- Retrieving the JIRA Administrator
- Scheme Entity Relations Map — This diagram illustrates the relationships between various JIRA entities and schemes.
- Sending JIRA Data to Support
- Setting Additional Fields for Issues Created from Email
- Setting a Default Value in the Description Field
- Setting Priority field value based on customfield value
- Showing Extended Timestamp in the Created Column of the Issue Navigator
- Single Sign-on
- Tracking the Time Taken for Each Workflow Transition
- Troubleshooting Issue Creation Via Email
- Using JIRA to Manage reusable modules
- Using validators to make custom fields required on transition screens — Use the 'Fields Required' workflow validator that is packaged in the JIRA Suite Utilities.
- We already have users & groups defined elsewhere - can JIRA make use of these?
- Where are the application server logs? — A decision tree for finding JIRA logs on your system
- Why doesn't JIRA have a Severity field like Bugzilla?
- Workflows Guidebook
- XML format for import & export files

Installation Notes

- Configuring IIS with Tomcat
- Database Notes — These pages contain notes on configuring JIRA with various databases.
    - Incorrect database type specified
    - Restarting JIRA from the Setup Wizard
    - Database limitations on number of projects
    - JIRA and HSQL
        - Running SQL commands in a HSQL database — On rare occasions, one may wish to run raw SQL queries on a JIRA or Confluence database. This page describes how to obtain a SQL console for hsqldb databases, which are built into JIRA and Confluence for evaluation purposes.
    - JIRA and MS SQL Server 2005
        - Connecting to named instances in SQL Server
        - Error caused by SET NOCOUNT in MS SQL Server
        - MS SQL Server 2000 Startup errors
        - Setting Up a SQL Server 2005 database for JIRA
    - JIRA and MS SQL Server 2008
    - JIRA and MySQL
        - Configuring MySQL 5.1 to store non-ASCII characters
        - JIRA Cannot Connect to MySQL with Named Pipes Enabled
        - JIRA Cannot Create Issues when Using MySQL with Binary Logging
        - MySQL Administrator and Data Truncation Errors
        - MySQL Data Access Exception - Errcode - 17 occurs with JIRA
        - Setting Up a MySQL Database on Linux for JIRA
    - JIRA and Oracle
        - Configuring Datasource for Oracle 10g JDBC drivers
        - Restoring data using I-Net (Oranxo) Driver for Oracle
        - Store Workflow on Disk with Oracle 8 — A workaround for the problem of > 4000 character workflows in Oracle 8 is to store these on disk, instead of in the database.
    - JIRA and PostgreSQL
        - Setting up a PostgreSQL Database on Linux for JIRA
- How to Set Up SMTP Relay in Exchange 2007
- Installation Troubleshooting Guide
- Installing a LDAP server on Debian Linux for use with JIRA
- Installing Java on Ubuntu or Debian
- Installing JIRA on Mac OS X
    - Configure JIRA as service on Mac OS X
- Is Clustering or Load Balancing JIRA Possible
- java.lang.NoClassDefFoundError
- JVM and Appserver configuration info
- LicenseFactory error after upgrading JIRA
- Logging request headers
- Running multiple instances of JIRA on one machine
- Solaris ClassNotFoundException
- Windows cannot find -Xms128m

Also check out the JIRA Community Space and the Forums

Performance FAQ

Search the Performance FAQs:

FAQs

- Is your JIRA Running Slowly

Is your JIRA Running Slowly

This page is deprecated. Please see Crashes and Performance Issues Troubleshooting instead.

Usage FAQ

Search the Usage FAQs:

FAQs

- Modifying the JIRA Footer
- 'Road Map', 'Change Log ', and 'Versions' Project Tabs Are Not Visible
- Why Do Linked Issues in JIRA Appear with a Strike-Through
- How to Enable the FishEye Plugin from the Plugin Administration Screen
- How to Add the Priority Field Into the Email Subject

- How to Change the Number of Rows Allowed in the Text Type Custom Field Renderer
- How to Make a Federated JIRA Instance
- How to Remove 'Unknown' Option from 'Component' and 'Fix Versions'
- Automatic Escalation of issues
- How to Activate Header Row for Subtask List in Issue Detail View
- How to Limit the Number of Characters Entered in a Summary Field
- JIRA's Timestamp Doesn't Match the System Time
- How to Attach a File During Issue Creation
- How to Remove Duplicate Entries for Names and Groups
- How to Export Users to CSV from JIRA
- How Come JIRA does Not Show Direct SQL Data Modifications
- How to get JIRA Pages to Render when URL Contains an Underscore
- Certain Action Words Show Incorrect Language Translations
- How to disable or enable the GOT FEEDBACK button
- Changing the number of users synchronised from LDAP to JIRA
- How do I reduce my user count in JIRA
- Adding custom content to the front page
- Allow editing of Closed Issues
- Allowing users to create issues anonymously
- Anonymising JIRA Data
- Appending Email Addresses to Comments Made by Anonymous Users when Using a Mail Handler
- Asking for an attachment on the Create Issue page
- Automatically Populate Timezone from a Created Issue
- Can I store customer details, like company, address and contact information, in JIRA?
- Change JIRA Browser Icon
- Changing Custom Field Types
- Changing Templates Used by Export to Excel from the Issue Navigator
- Changing the default attachment size limit
- Changing the default session timeout
- Changing the Default Tab Panel from Comments to All
- Changing the Due Date Input Format
- Changing the Project Key
- Changing the Project Key Format
- Changing the Size and Content of the Components Select List
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size of the Text Area Custom Field
- Changing the Temporary Directory
- Changing Usernames in JIRA
- Configuring project specific security
    - Controlling project visibility
    - Using Project Level Security with Project Roles — This tutorial provides a step-by-step guide for creating project roles and using them in an issue security scheme. We recommend creating a test project and two test users for this tutorial.
    - Using Project Level Security with User Groups
- Connecting to SSL services
- Creating a Custom Workflow
- Creating an Unassigned Issue
- Creating Issues via direct HTML links
- Current Reporter Browse Project Permission
- CVS ssh Jira Integration
- Disabling Form Token Checking
- Displaying a Field Based on Another Field Selection
- Editing a custom field option
- Escalating issues (or sending email notifications) when the set turnaround time is exceeded
- Field Layout Schemes in JIRA 3.x
- Fields Allowing Custom HTML or JavaScript
- Finding the Id for Issue Types
- Group Name Guidelines for JIRA
- How can I control the editing of issue fields via workflow?
    - Using a Workflow to control edit of an issue by changing Workflow XML
    - Using Templates to control edit of an issue
- How do I assign issues to multiple users
- How do I delete a user account?
- How do I disable Firebug for JIRA?
- How Do I Use an SSL Certificate Generated Using openssl?
- How the CreateOrCommentHandler works?
- How to change Multi Select Custom field size using script
- How to change the location of stdout and stderr logs
- How to clear the resolution field when the issue is reopened
- How to configure sub-task to have a specific screen?
- How to convert types using Jelly
- How to create a download link to a file
- How to disable the Resolve issue screen while resolving issues
- How to display a different format for the Number customfield
- How to ensure the Road Map tab is visible
- How to Get Unicode 'non-ASCII' Characters in HTTPS URL to Appear Correctly
- How to have long component version names display properly in the Issue Navigator
- How to Remove 'NONE' from a Select List Custom Field
- How to Rename the 'Priority' Field in the Issue Navigator
- How to re-order statuses
- How to re-order the list of issue operation in an issue

- How to resize Free Text Field customfield
- How to resize the 'Components' and 'Affects Versions' fields in the Issue Navigator
- How to Restrict the Subversion Commits Tab to Selected Projects or Users
- How to search by number range in the Issue Navigator
- How to show a transition only when the Assignee is different from the Current User
- Importing data
- Importing user from LDAP
    - Neat JIRA LDAP tricks — Gianugo has an interesting blog up about how to take the pain out of migrating users from LDAP
- JIRA as a Support System
    - Jelly Escalation
    - Simple Escalation
- Letting customers only create issues — This page describes a minor JIRA modification which redirects users to an arbitrary page after creating issues (and potentially other operations). It is mainly of interest to JIRA Professional and Standard users.
- Linking to local file under Firefox
- Login problems
- Mail error - Unable to relay
- Making JIRA login case insensitive for JIRA 3.13.x
- Outward Link Description and Inward Link Description
- Parsing utf-7 emails
- Project-specific email templates
- QuickSearch guesses the issue key prefix (sometimes)
- Receiving a Daily Summary of Updated Issues
- Recieving Notification for Select Issues or Updates
- Removing Commas for Values Held in Number Field Custom Field Type
- Removing invalid characters from XML backups — In older versions of JIRA it was possible to cut & paste text containing control characters into JIRA issue fields. This causes problems, because JIRA's backup format is XML, and XML does not allow for the storage of most control characters .
- Removing NONE from the Issue Security Drop-Down List
- Re-order workflow transactions
- Resolved issues appearing in Open issues filters
- Restricting the Visibility of Worklog on an Issue
- Retrieving the JIRA Administrator
- Scheme Entity Relations Map — This diagram illustrates the relationships between various JIRA entities and schemes.
- Sending JIRA Data to Support
- Setting Additional Fields for Issues Created from Email
- Setting a Default Value in the Description Field
- Setting Priority field value based on customfield value
- Showing Extended Timestamp in the Created Column of the Issue Navigator
- Single Sign-on
- Tracking the Time Taken for Each Workflow Transition
- Troubleshooting Issue Creation Via Email
- Using JIRA to Manage reusable modules
- Using validators to make custom fields required on transition screens — Use the 'Fields Required' workflow validator that is packaged in the JIRA Suite Utilities.
- We already have users & groups defined elsewhere - can JIRA make use of these?
- Where are the application server logs? — A decision tree for finding JIRA logs on your system
- Why doesn't JIRA have a Severity field like Bugzilla?
- Workflows Guidebook
- XML format for import & export files

Modifying the JIRA Footer

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

'Powered by Atlassian'

Also see clause 7 of the Atlassian End User License Agreement. This states that you must not remove the "Powered by Atlassian" link at the end of this file.

How to Modify the Footer

The footer can be modified by editing `<install directory>/atlassian-jira/WEB-INF/classes/templates/plugins/footer/footer.vm`.

```
...
<span #if($smallFooter) class="smallfooter" #else class="poweredbymessage" #end>
    #if ($longFooterMessage)Powered by #end<a
href="$externalLinkUtil.getProperty('external.link.jira.product.site')"
class="smalltext">Atlassian JIRA</a>
    the Professional <a
href="$externalLinkUtil.getProperty('external.link.jira.product.site')">Issue Tracker</a>.
    <span #if($smallFooter)class="smallgreyfooter" #else style="color: #666666;"
#end>($buildInformation)</span>#if($smallFooter)<br/>#end
    - <a
href="$externalLinkUtil.getProperty('external.link.jira.raise.issue')">Bug/feature request</a>
    - <a
href="$externalLinkUtil.getProperty('external.link.atlassian.news')">Atlassian news</a>
    - <a href="${req.contextPath}/secure/Administrators.jspa">Contact
Administrators</a>
</span>

\#if($smallFooter)
    </span>
\#end

</div>
```

Keep in mind the considerations about Modifying Jira Templates and JSPs. Restart JIRA after your customization in order for it to take effect.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

'Road Map', 'Change Log ', and 'Versions' Project Tabs Are Not Visible

Sometimes users cannot view the `Road Map`, `Change Log`, and `Versions` project tab panels under the `Browse Project` page. This is usually because the `Fix version/s` field is configure as `hidden` or the Road Map Panel, Change Log Panel, and Versions Panel plugin has been disabled in Project Panels Plugin.

In order to display those project tab panel, user needs to `unhide` the `Fix version/s` field. This setting can be changed from the `Field Configurations` section of the Administration Panel, which can be found under the `Issue Fields` heading.

For more information, please refer to Generating Reports and Specifying Field Behaviour.

To enable the plugin modules, go to `Administration >> System >> Plugins >> Project Panels Plugin` and enable the Road Map Panel, Change Log Panel, and Versions panel plugin so that the `Road Map`, `Change Log`, and `Versions` project tab panels will appear again in the `Browse Project` page.

Why Do Linked Issues in JIRA Appear with a Strike-Through

> ⚠ Always back up your data before performing any modification to the database.

JIRA shows linked issues as closed, even though they are open. There are no corresponding error messages found in JIRA's logs. This is caused by issues in the JIRA database having an invalid resolution. The following SQL query can be executed to confirm this:

```
select pkey, resolution from jiraissue where resolution not in (select id from resolution);
```

To fix the problem, run this SQL statement and reset the resolution to UNRESOLVED for the issues with invalid resolution.

```
UPDATE jiraissue SET resolution = NULL where resolution not in (SELECT id FROM resolution);
```

How to Enable the FishEye Plugin from the Plugin Administration Screen

When navigation to the Fisheye Plugin page under, Administration > Plugins > FishEye Plugin, JIRA shows that the plugin is disabled and the enable link to activate the plugin is missing. This occurs if the FishEye plugin was disabled in a previous version of JIRA. After an upgrade of JIRA, the enable link in Plugin Administration is missing. First paste the following URL into your web browser as the JIRA Administrator:

http://<JIRA--BASE_URL>/secure/admin/jira/ViewPlugins.jspa?mode=enable&pluginKey=com.atlassian.jira.ext.fisheye

If that fails to resolve the issue, then run the following SQL Query:

Please backup your data before running any SQL queries.

```
update propertystring set propertystring.propertyvalue = 'true' where propertystring.ID IN
(select ID from propertyentry where propertyentry.PROPERTY_KEY =
'jira.plugin.state-.com.atlassian.jira.ext.fisheye');
```

How to Add the Priority Field Into the Email Subject

You wish to add the priority into the Email Notification subject line that JIRA sends out when an issue is created or updated. This feature is available as of JIRA 4.1. More information on this feature can be found in our blog or in Customising Email Content.

How to Change the Number of Rows Allowed in the Text Type Custom Field Renderer

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

The default `Free Text Custom Field` editor renders as a `textarea` that is 4 lines long. This may be too small for comfortable data entry and editing. The functionality that controls the behavior of that field is hardcoded in file `<install directory>>atlassian-jira/WEB-INF/classes/templates/plugins/fields/edit/edit-textarea.vm`. Edit all occurrences of parameter `rows` in the `edit-textarea.vm` file to the desired value and restart JIRA for the changes to take effect.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

How to Make a Federated JIRA Instance

This page discusses some of the options around integrating one JIRA instance with another, when trying to update an issue in one instance based on an update from another.

The solution is fairly complex, and Atlassian recommends working with a partner on a solution.

Options

- Option one is to use a notification scheme from the original instance, then a Create or Comment Mail Handler on the destination instance, to update the tickets across instances.
- Option two is to use a Jelly Service and the remote Api script to do something similar. You'd have to watch out for both problems above as well.

Challenges
A couple challenges with either approach:

1. You can run into a loop, where one instance updates the other, then vice-versa, ad infinitum.
2. You have a challenge of mapping which issue from the source maps to which issue from the destination.
   We thought you might be able to use a custom field from the source instance, then populate that with the issue from the destination instance, but you'd still need a bit of customisation from your mail handler to parse the email to do that mapping.

Consult our partner network for guidance.

How to Remove 'Unknown' Option from 'Component' and 'Fix Versions'

There is always an `Unknown` option in `Component` and `Fix Version/s` fields when creating/editing an issue.

To remove the field, make the `Components` and `Fix Version/s` field a required field in field configuration `Administration >> Issue Fields >> Field Configurations`. This will cause the `Unknown` option to disappear and also make the field as a mandatory field on `Create Issue screen`. For more information on field configuration, please refer to our documentation on Specifying Field Behaviour.

Automatic Escalation of issues

JIRA does not have the ability to auto-escalate issues meeting a certain criteria. There are a two solutions on how to implement Automatic Escalation in JIRA:

1. By using a Jelly script, issues that meet a certain criteria from a filter can be made to perform an action as described in the Jelly Escalation documentation.
2. Users/groups can also be notified with subscription to filters (e.g. Users can be periodically notified if an issue has been update for the last 2 days). Please refer to Receiving Search Results via Email for more information.

How to Activate Header Row for Subtask List in Issue Detail View

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

Some times it is nice to display the header row when viewing a list of sub-tasks in the Issue Detail view. There was a post about this in our forum community: http://forums.atlassian.com/thread.jspa?threadID=27500.

In order to accomplish this, it is necessary to modify `<install directory>/atlassian-jira/includes/panels/issue/view_subtaskissues.jsp`. Then implement the following method. setDisplayHeader as follows:

```
layout.setDisplayHeader(true);
```

Restart JIRA for the change to take effect.

How to Limit the Number of Characters Entered in a Summary Field

To limit the number of characters entered in a summary field modify, the following velocity file `<installation directory>/atlassian-jira/WEB-INF/classes/templates/jira/issue/field/summary-edit.vm`. Please restart JIRA after the changes have been made. Note, these changes will also affect the size of summary fields for existing issues.

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

JIRA's Timestamp Doesn't Match the System Time

This can occur from one of two reasons:

First Scenario:

Timestamps shown in the JIRA UI don't match the users' timezone but that of the server location. The timezone used is a JVM system property which defaults - unless specified - to that of the hosting operating system

It is not possible to localize the timezone on a per-user basis. It is possible to change the JVM timezone by setting the following JVM command line property: `-Duser.timezone=<TZ>`. Possible values are in the *zoneinfo* format (please see this list of zoneinfo time zones).

To set a command line option, see Setting properties and options on startup.

Global organizations may consider setting the value to UTC.

Second Scenario:

JIRA's date/time may not have been updated after day light savings has taken affect - this is generally an issue with older versions of JAVA.

Upgrading JVM to the latest version will solve the problem. Please download the latest version from java.sun.com.

For further reference, see the following document:

- http://www.atlassian.com/software/jira/docs/latest/java.html

How to Attach a File During Issue Creation

It is possible to attach files during the issue creation screen. Attachment is a field which must be configured to display on the issue creation screen.

1. Under `Administration >> Issue Fields >> Field Configuration`, ensure that the `Attachment` field is not hidden.
2. Inspect JIRAs `Screen Schemes` and see this article on how Screens are associated with Issue Operations. Once you've identified the screen that's used for Issue Creation, ensure that the `Attachment` field is there.

How to Remove Duplicate Entries for Names and Groups

There are duplicate entries for names and groups in User browser and Group Browser. This is caused by the `<install directory>WEB-INF/classes/osuser.xml` file having the providers defined twice.

Make sure there are no duplicate entries in `WEB-INF/classes/osuser.xml`. If the JIRA installation type is WAR/EAR, please re-deploy WAR and restart JIRA. If the solution above does not address problem, try re-indexing JIRA, followed by a restart.

How to Export Users to CSV from JIRA

Sometimes it is useful to get a list of users exported to CSV for various purposes. JIRA doesn't currently have this functionality but you can leverage various database functionalities to do this.

Run one of the following queries specific to your database. The output will consist of the id, username and full name of the users.

Always back up your data before performing any modification to the database.

MySQL

```
SELECT u.id, u.username, ps.propertyvalue INTO OUTFILE '/path/to/userlist.csv'
    FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
    LINES TERMINATED BY '\n'
FROM userbase u JOIN propertyentry pe ON pe.entity_id = u.id JOIN propertystring ps ON ps.id =
pe.id
WHERE  property_key = 'fullName' ORDER BY propertyvalue ASC;
```

PostgreSQL

```
SELECT u.id, u.username, ps.propertyvalue INTO TEMPORARY TABLE userdetails
FROM userbase u JOIN propertyentry pe ON pe.entity_id = u.id JOIN propertystring ps ON ps.id =
pe.id
WHERE  property_key = 'fullName' ORDER BY propertyvalue asc;
```

```
copy userdetails to '/path/to/userlist.csv' using delimiters ',' CSV QUOTE AS '"';
```

How Come JIRA does Not Show Direct SQL Data Modifications

Direct database modifications are not supported by Atlassian. Always back up JIRA's database before performing any modification to it.

When making direct database modification queries (INSERT, UPDATE, DELETE), the changes are not reflected in the data presented on the application UI. The JIRA application has been written under the assumption that one instance will have exclusive access to the database schema. Some of the data is cached by the application and those caches are updated only when the application is aware that these have been changed. For the changes to be reflected:

1. Restart the application.
2. Perform any automated, scripted or mass edit through the application itself using one of the provided facilities:

- JIRA RPC plugin
- Jelly Scripting
- Bulk Operations

How to get JIRA Pages to Render when URL Contains an Underscore

When logging into JIRA through Internet Explorer with a URL that includes underscore (http://mycompany_jira.com), it will redirect every login attempt back to the login page. Logging in through through the Firefox, renders correctly.

This problem is specific to Internet Explorer. Internet Explorer is compliant with the RFC that defines the validity of an URL. An URL considered invalid in the RFC rules if the URL contains an underscore ("_"). This also could be caused by the base URL and Apache.

This can be resolved with either of the following methods:
1. Connect JIRA using the IP address or use "localhost" if it is on the same machine.
2. For a long term solution change the defined URL that doesn't contains an underscore.

Certain Action Words Show Incorrect Language Translations

Due to the way certain Java classes handle internationalization, some action words are dependent on the locale set by the JVM (which uses the system default locale).

For example, if the default system locale is set to French but JIRA's default language is set to English some words will still appear in French. Fortunately you can set the following properties to over-ride the default locale used by the JVM:

```
-Duser.language=en -Duser.country=US
```

See this page for a sample of the supported languages and country codes

How to disable or enable the GOT FEEDBACK button

By default, JIRA provides a '**GOT FEEDBACK**' link at the top right of all pages. This link provides an easy way for any JIRA user to provide Atlassian with feedback about JIRA, which importantly influences how JIRA is improved in future versions.

Some JIRA administrators, however, prefer that such feedback is coordinated via them and so prefer to remove this feedback link.

To remove the 'GOT FEEDBACK' link:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **'Administration' > 'Plugins' > 'Plugins'**. The Universal Plugin Manager will be displayed, showing the plugins installed on your JIRA site.
   Keyboard shortcut: **'g' + 'g' +** start typing **'plugins'**
3. Click the '**Show System Plugins**' link to reveal the list of system plugins.
4. Search for the 'JIRA Feedback Plugin' and click its name to expand the details of this plugin.
5. Click the '**Disable**' button.

To re-enable the '**GOT FEEDBACK**' link, follow the same steps above but at the last step, click the '**Enable**' button.

Changing the number of users synchronised from LDAP to JIRA

If you have connected JIRA to an LDAP directory for authentication, user and group management, you may want configure JIRA to synchronise a subset of users from LDAP rather than all users. There are two reasons for why you might make this change:

- Improving performance — If you have performance issues during synchronisation process, you may be able to improve this by synchronising a subset of data instead. See this knowledge base article for more information: Performance Issues with Large LDAP Repository - 100,000 users or more.
- Reducing your user count *(not recommended)* — You can synchronise a subset of users to JIRA from LDAP to reduce your user count. This will allow you to count less users against your JIRA license. However, synchronising a subset of users to JIRA from LDAP is not the recommended method for reducing your user count in JIRA. See this FAQ for more information: How do I reduce my user count in JIRA.

Procedure

The procedure for configuring JIRA to synchronise a different number of users from LDAP depends on how you initially set up your LDAP directory. For example, if you have all your JIRA users in one organisational unit and your non-JIRA users in another organisational unit, then you can simply configure JIRA to only synchronise users against a particular DN (distinguished name). However, if your setup is not so simple (e.g. you have your JIRA users and non-JIRA users in the same node), you will need to define an LDAP filter to synchronise the relevant users. Both of these methods are outlined below.

Synchronising against Base DN, Additional User DN and Additional Group DN

1. Log in as a user with the **JIRA Administrators** global permission.
2. Select **Administration > Users > User Directories**.
   Keyboard shortcut: **g + g +** start typing **directories**.
3. Update the **Base DN** field, and optionally the **Additional User DN** and/or **Additional Group DN** to query against the directory server as desired.
   For example, if you have configured all of your JIRA users in the jira-users organisational unit only, for your company at mycompany.example.com, your configuration would look like this:
   - **Base DN** — `dc=mycompany,dc=example,dc=com`
   - **Additional Group DN** — `ou=jira-users`

Defining an LDAP filter

1. Log in as a user with the **JIRA Administrators** global permission.
   Select **Administration > Users > User Directories**.
   Keyboard shortcut: **g + g +** start typing **directories**
2. Update **User Object Filter** and/or **Group Object Filter** fields as desired. The syntax for LDAP filters is not simple and your query will depend on how you have set up your LDAP directory.
   For example, if you have configured only JIRA groups to have 'jira' in the CN, you can use a wildcard search in your filter to find them by setting the **Group Object Filter** = `(objectCategory=group)(cn=*jira*)`
   More information on defining LDAP filters is available in the pages linked in the *Related Topics* section below.

Related topics:

How do I reduce my user count in JIRA

Performance Issues with Large LDAP Repository - 100,000 users or more

Unable to Create Issues Due to Exceeded License

How to write LDAP search filters

MSDN guide to LDAP search filter syntax

How do I reduce my user count in JIRA

You may want to reduce your user count in JIRA if you have *exceeded your user count* or if you want to *change to a lower-tier license* to reduce costs. If you only have a subset of your total users using JIRA, you can reduce your user count by simply configuring JIRA to allow less users to log in to JIRA. The methods for reducing your user count in JIRA are described below.

Procedure

The *recommended method* for reducing your user count in JIRA is to remove users from all groups with the 'JIRA Users' global permission. This is described in the following knowledge base article: Unable to Create Issues Due to Exceeded License.

Alternatively, if you have connected JIRA to an LDAP directory, you may want configure JIRA to synchronise a subset of users from LDAP rather than all users. This is described in the following FAQ: Changing the number of users synchronised from LDAP to JIRA. However, this can be a complicated procedure and we recommend that you do not use this method unless necessary.

Related topics:

Unable to Create Issues Due to Exceeded License

Changing the number of users synchronised from LDAP to JIRA

Adding custom content to the front page

Custom HTML content can be easily added to the dashboard by a JIRA administrator.

For example, to customise the text that appears on users' dashboards, click on General Configuration, click '**Edit Configuration**', and edit the **Introduction** field.

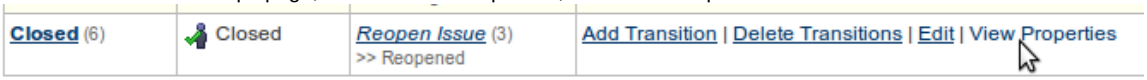
Note that look and feel can also be customised (e.g. add your organisation's logo and/or preferred colour scheme).

Additionally, the announcement banner is useful for sending broadcasts to all JIRA users.

Allow editing of Closed Issues

By default, it is not possible to edit an issue while in the "Closed" state. If you would like to allow editing of closed issues, this can be done by editing the workflow (see Configuring Workflow), and removing the `jira.issue.editable` flag from the **Closed** step. The steps are as follows:

1. Find the active workflow that applies to the issues you wish change. This is most easily done by going to the 'Workflow Schemes' admin page, then clicking on the Workflow link in the row applying to the issues' project and issue type.
2. It is not possible to edit an active workflow, so you will need to either make a copy (if using the default `jira` system workflow) or draft of the workflow, and edit that.
3. On the View Workflow Steps page, in the 'Closed' step's row, click 'View Properties'.



4. You should see a `jira.issue.editable` property with value `false`. Delete it, or set the value to true.
5. Publish your draft workflow, or if editing a copy, activate the workflow by creating a new workflow scheme associated with the edited workflow, and then associating it with your project.

> **Note**
>
> You can use the `jira.issue.editable` flag to enable/disable editing of issues at any step (not just the 'Closed' step).

This property and a number of others are also discussed at Workflow Properties.

Allowing users to create issues anonymously

JIRA can be configured to allow users to create issues without having logged into JIRA. There are two related actions:

1. Allowing users to browse and search issues in the project without logging in.
2. Allowing users to create issues in that project without logging in.

These can be achieved by adding the *Anyone* group to the **Browse Project** and **Create Issue** permissions in the permission scheme for the project. Additionally, *Reporter*, in the project's field configuration scheme, must be set as optional.

Any issue created by a user who is not logged in will display 'Anonymous' for the reporter of the issue.

Anonymising JIRA Data

For JIRA versions 3.7 through to 4.1, data sent via Administration -> Support Request is anonymised by default, and it is thus the easiest route to sending us anonymised data (but be sure your mail server has a username/password specified, so relaying is allowed).

As of JIRA 4.1.1, it is no longer possible to send data via the Administration -> Support Request page. To do so, please login to Atlassian Support and attach the data to the existing issue.

Support requests are resolved **much** faster if a data export is provided. However, with sometimes this is not an option because the data contains sensitive information.

In JIRA 3.7.x to 4.1, JIRA automatically anonymises data sent to Atlassian from the Administration -> Support Request page. For earlier or later versions, or people who want to anonymise JIRA data from the command-line, we've created a data 'anonymiser', which replaces most text in JIRA XML backups with **x**'s.

The anonymiser can be downloaded from here.

Unzip the package, then open a console and in the `jira_anon` directory run:

```
$ java -jar joost.jar <name of your backup file.xml> anon.stx > <name of the anonymised backup
file to be generated.xml>
```

For example:

```
$ java -jar joost.jar backup.xml anon.stx > anon-backup.xml
```

Then zip the generated backup XML file, and attach it to a support case on https://support.atlassian.com

The anonymiser currently replaces the following text with x's:

- Issue summary, environment, and description
- Comments, work logs, change logs
- Project descriptions
- Descriptions for most elements (notification schemes, permission schemes, resolutions)
- Attachment file names.
- "Unlimited text" custom fields

Check anon-backup.xml to ensure it's clean enough for your needs before you send to us.

Problems?

Invalid XML Characters

If, when you run the anonymiser, you get an error indicating that there are invalid XML characters in the XML backup of your database, run our utility to remove invalid XML characters first before anonymising.

Out of Memory / Heap Space Errors

If creating your anon-backup.xml partway through, you are likely facing a memory limitation with running the 'java' command with the default settings. To allow the command more memory for the command, simply add arguments after the 'java' command, like so:

```
$ java -Xms512m -Xmx512m -jar joost.jar backup.xml anon.stx > anon-backup.xml
```

Note: you may need to adjust the memory allocation beyond '512m' if the process continues to fail.

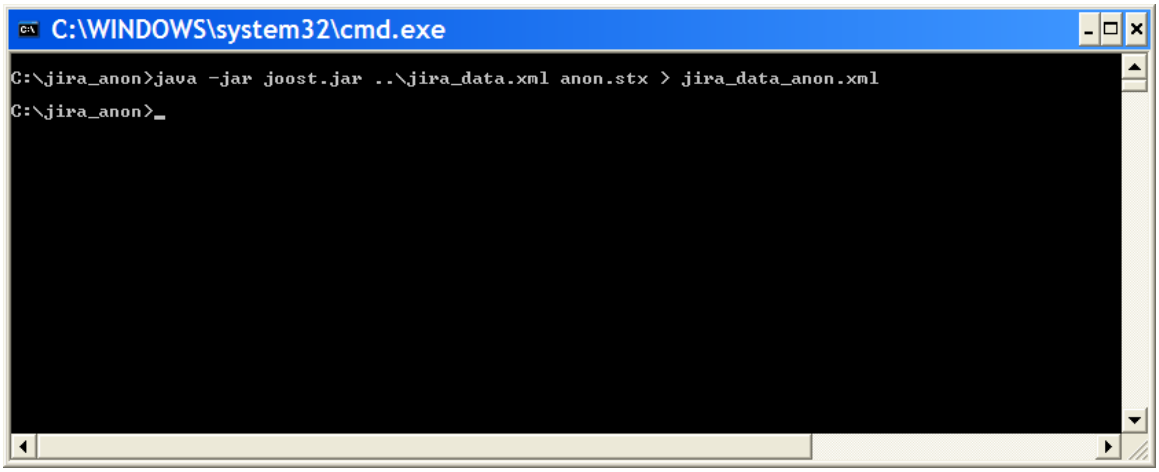
Java Version

You will need Java 1.4 or above to run this. You can check your Java version by running `java -version`, eg:

```
$ java -version
java version "1.5.0_07"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_07-b03)
Java HotSpot(TM) Client VM (build 1.5.0_07-b03, mixed mode, sharing)
```

If you find yourself using JDK 1.3 or earlier, check your path (`echo %PATH%` on Windows, `echo $PATH` on Unix) and ensure that the right version of Java is at the beginning. See the docs for more info on setting up Java.

The screenshot below is a simple example of how it is run in the command prompt of Windows XP:



Appending Email Addresses to Comments Made by Anonymous Users when Using a Mail Handler

To append email addresses to comments made by anonymous users when using a mail handler, follow the instructions below.

There is a feature request at JRA-21468. Vote to have this included in the product.

Follow the instructions in How to make a JIRA patch.

1. Modify AbstractCommentHandler.java by adding 6 more lines after line 81, after the code block under `if (body != null)`:

```
// append the From address at the end of  body
                    body += "\n[Commented via e-mail ";
                    if (message.getFrom() != null &&  message.getFrom().length > 0)
                        body += "received from: " +  message.getFrom()[0] + "]";
                    else
                        body += "but could not establish  sender's address.]";
```

2. Place the compiled .class file under `<jira-install` /WEB-INF/classes/com/atlassian/jira/web/action/issue/AbstractCommentableIssue.class
3. Restart JIRA

Tested on JIRA 3.13.5

Asking for an attachment on the Create Issue page

To prompt for an attachment on the Create Issue page, go to the relevant Field Configuration page (eg. Administration -> Issue Fields -> Field Configurations -> Default Field Configuration), and unhide the 'Attachment' field.

Automatically Populate Timezone from a Created Issue

Atlassian Support likes to use customers' timezones to respond more effectively to support tickets. We extract timezone information automatically, so customers don't need to enter it manually. To do that, we add JavaScript to our custom field description:

```
Your timezone. The value is set automatically from your browser, so please only modify if it's
incorrect. <br>
<b><a href="http://www.worldtimezone.com/" target="timezone">World Timezones</a></b>

<script language="JavaScript">

<!--
function setCustomerTimezone()
{
 tzlist = document.getElementById("customfield_10421");
 if (tzlist) {
  if (/CreateIssueDetails.jspa/.test(tzlist.form.action)) {
// Value has not yet been set   if (tzlist.value == -1) {
    offset = new Date().getTimezoneOffset() / 60; // hours from GMT
    if (offset <= 0) { tzlist.value="GMT+"+(-offset);  }
    else { tzlist.value = "GMT-"+offset; }
//   }
  }
 }
}

window.onload = setCustomerTimezone;

//-->
</script>
```

Related Pages

Setting Priority field value based on customfield value
How to Set Default Comment Security Level
Creating Help for a Custom Field
Displaying a Field Based on Another Field Selection

Can I store customer details, like company, address and contact information, in JIRA?

JIRA itself stores only minimal user data (username, name, email, preferences). Since JIRA 3.7, you are able to store data in user 'properties' . You can store each customer detail as a separate user property, or create a wiki page for the customer and link to that instead. You could do this in Confluence by adding a Customer space and creating a page for every customer with their details. Then in JIRA, add a user property containing the link to that customer's page.

An alternative is to store user data in an LDAP server such as Active Directory or OpenLDAP. You can then authenticate users in JIRA against their LDAP password (see Configuring LDAP) and link to their full LDAP profile if available online.

There is also an open feature request for improved user properties at JRA-6354. You may wish to sign up for a user account and vote or comment to help influence our product roadmap.

Change JIRA Browser Icon

The JIRA logo ' ' is displayed in the user's browser to identify the JIRA browser tab. To use a custom image for your JIRA site:

1. Obtain or create an image in PNG file format. To maximise browser compatibility, it should be 32x32 pixels, 71x71 DPI and have 8 bit colour depth.
2. In your JIRA install, find the `<jira-installation-directory>/atlassian-jira/images/icons` directory (assuming you are using a 'recommended' distribution of JIRA).
3. Backup the file `favicon.png`

4. Replace the `favicon.png` with your custom PNG image.
   You may also need to backup and replace the following images in your JIRA Installation Directory:
   - `<jira-installation-directory>/atlassian-jira/favicon.ico`
5. Restart your application server.

Users may need to clear their browser cache to view the new image.

Changing Custom Field Types

You generally can't shift between custom field types since the data type they store may not match.

Migrating Custom Field Content Manually

One workaround is to use bulk edit operations to migrate content:

1. Create a new field
2. Using Advanced Searching, search for all the instances of the old field.
3. Using a bulk edit operation, populate the new field with the value of the old field for all the issues found. If some issues are closed, you may have to see Allow editing of Closed Issues
4. Repeat this process for all values in the field.
5. Delete the old field, or remove it from the screen scheme.

Upgrading Custom Fields

Certain fields can be safely *upgraded*, such as Version and Select lists to their multiple values counterpart. You can change the "customfieldtypekey" in the "customfield" table to whatever you need it to be. The table below lists the keys for commonly changed fields.

Custom Field Type	Type Key
Single Version	com.atlassian.jira.plugin.system.customfieldtypes:version
Multi Version	com.atlassian.jira.plugin.system.customfieldtypes:multiversion
Single Select	com.atlassian.jira.plugin.system.customfieldtypes:select
Multi Select	com.atlassian.jira.plugin.system.customfieldtypes:multiselect
Multi User	com.atlassian.jira.plugin.system.customfieldtypes:multiuserpicker

When moving back from a multi select list a select list, you **have** to make sure that only one item is selected for each multi select list.

When moving from multi-select to multi-user, you **have** to ensure that each select-list value is a username (`userbase.username` value).

For select lists, you also need to update the "customfieldsearcherkey" field to use an appropriate searcher:

- For multi-selects, it is "com.atlassian.jira.plugin.system.customfieldtypes:multiselectsearcher"
- For select lists, use "com.atlassian.jira.plugin.system.customfieldtypes:selectsearcher"
- For multi-user pickers, use "com.atlassian.jira.plugin.system.customfieldtypes:userpickersearcher"

Examples

For example if you want to update all the version custom fields to become multiple version custom fields, you can use the SQL below.

```
UPDATE customfield
    SET customfieldtypekey = 'com.atlassian.jira.plugin.system.customfieldtypes:multiversion'
WHERE customfieldtypekey = 'com.atlassian.jira.plugin.system.customfieldtypes:version'
```

Or if you wanted to convert multi-select-list custom field to a multi-user custom field, first check that all custom field values map to users:

```
select * from customfieldvalue where id=
    (select id from customfield where cfname='multisel3') and
    stringvalue not in (select username from userbase);
Empty set (0.02 sec)
```

Then you can change the custom field type:

```
UPDATE customfield
    SET CUSTOMFIELDTYPEKEY='com.atlassian.jira.plugin.system.customfieldtypes:multiuserpicker',

CUSTOMFIELDSEARCHERKEY='com.atlassian.jira.plugin.system.customfieldtypes:userpickersearcher'
  where cfname='MyMultiSelect';
```

Or if you wanted to convert text-field custom field to a free-text-field(unlimited text) custom field, first assign the value from stringvalue field to textvalue:

```
UPDATE customfieldvalue SET textvalue=stringvalue WHERE customfield=(SELECT ID FROM customfield
WHERE
customfieldtypekey='com.atlassian.jira.plugin.system.customfieldtypes:textfield' AND
cfname='Text Field');
```

Then, change the custom field type by updating the customfield table as below:

```
UPDATE customfield SET
CUSTOMFIELDTYPEKEY='com.atlassian.jira.plugin.system.customfieldtypes:textarea',
CUSTOMFIELDSEARCHERKEY='com.atlassian.jira.plugin.system.customfieldtypes:textsearcher'
where cfname='Text Field';
```

Restart JIRA. Then reindex (Administration -> Indexing) to update the search index.

Changing Templates Used by Export to Excel from the Issue Navigator

When exporting a set of issues to Excel, customisation to the layout templates are controlled in Velocity files. Velocity templates for the export formats are defined in file `/atlassian-jira/WEB-INF/classes/system-issueviews-plugin.xml`. The following files in particular define the Excel views:

```
atlassian-jira/WEB-INF/classes/templates/plugins/searchrequestviews/searchrequest-excel-header.v

atlassian-jira/WEB-INF/classes/templates/plugins/searchrequestviews/searchrequest-excel-footer.v

atlassian-jira/WEB-INF/classes/templates/plugins/searchrequestviews/searchrequest-description-he
```

Refer to Microsoft® Office HTML and XML Reference for further information on the syntax of the template contents.

Changing the default attachment size limit

To change the default size limit for attachments, see the Configuring File Attachments page.

Changing the default session timeout

To change the default session timeout (which is 300 minutes) you must edit the file `web.xml`. This file can be found in `<YOUR DEPLOYMENT>/WEB-INF/web.xml`.

✅ If you are deploying JIRA as a closed `.war` file you will need to unzip the .war, edit the file, and re-create the `.war` with exactly the same structure as it originally had.

> ⚠️ *The information on this page does not apply to **JIRA OnDemand**.*

The element you want to edit in the `web.xml` file is:

```
<web-app ...>
...
    <session-config>
        <session-timeout>300</session-timeout>
    </session-config>
...
</web-app>
```

The value within the **session-timeout** tag defines the amount of time the session will exist, in minutes.

Note that after editing the `web.xml` file you will need to restart JIRA for your change to take effect.

Changing the Default Tab Panel from Comments to All

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

The simple procedure on this page describes how to change the default view of issues from 'Comments' to 'All'. To do this, change the configuration file which controls the display of the tabs:

1. Edit the file `<jira-install\WEB-INF\classes\system-issuetabpanels-plugin.xml)`.
2. Each tab is controled by a `<issue-tabpanel>` tag. To change the default selection, the `<default>true</default>` needs to be placed in the wanted `<issue-tabpanel>` tag.

Changing the Due Date Input Format

The 'Look and Feel' page allows you to customise the time and date formats used throughout JIRA.

When specifying dates and times, they should be based on the Java SimpleDateFormat.

When you are not in edit mode on the 'Look and Feel' page, the examples in the rightmost column of the 'Date/Time Formats' section show you how the various formats will appear in JIRA.

Configuring Date Picker Formats

To set the format of date pickers in JIRA, see Configuring Advanced Settings.

The date or date/time formats are defined by two (Java and JavaScript) properties. These two properties must match for the date (or date/time) picker to work correctly.

For Java formats, specify date/time formats based on the Java SimpleDateFormat.

For JavaScript formats, specify date/time formats based on the Unix date format.

Here are some example US-based date configurations:

Preferred Date	**Value of the `jira.date.picker.java.format` property**	**Value of the `jira.date.picker.javascript.format` property**
Oct/1/10	MMM/d/yy	%b/%e/%y

10/01/10	MM/dd/yy	%m/%d/%y
Oct 1, 2010	MMM d, yyyy	%b %d, %Y
10/01/2010	MM/dd/yyyy	%m/%d/%Y

Here are some examples of date/time configurations:

Preferred Date/Time	Value of the `jira.date.time.picker.java.format` property	Value of the `jira.date.time.picker.javascript.format` property
15/Oct/10 8:50 AM	dd/MMM/yy h:mm a	%e/%b/%y %I:%M %p
10/15/10 08:50 AM	MM/dd/yy hh:mm a	%b/%e/%y %I:%M %p

Changing the Project Key

It is not currently possible to change the project key through JIRA's interface. The best way to do this is to:

1. Export your JIRA data to XML.
2. Go through the file and replace the instances of the project's key:
    - in the 'key' attribute of the Project entity
    - in the 'key' attribute of all Issue entities.
3. Search for the project key in the whole XML file. You should not find too many references. Change any that you find.
4. Rename all attachment folders, as the folder name depends on the project key. This includes one folder for each issue with an attachment, plus one top-level folder for the project.
5. Reimport your data.
6. Reindex the data by navigating to Administration -> System -> Indexing and selecting '**Re-Index**'.

Changing the Project Key Format

Please see Configuring Project Keys for details.

Although JIRA normally starts counting issue ids from 1 ('ABC-1', 'ABC-2' etc), you can adjust the starting count by editing the **project.pcounter** row in the JIRA database. Because JIRA caches this value in memory, you will need to shut down JIRA first, update this value in the JIRA database, then restart JIRA.

Changing the Size and Content of the Components Select List

This page describes how to increase the size of the Components drop-down list. Please see JRA-3028 for the full feature request.

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

This workaround will apply to all Components drop-down lists in the instance.

Increasing the size of the field

Edit `<atlassian-jira/WEB-INF/classes/templates/jira/issue/field/components-edit.vm`. Change the line:

```
<select multiple name="$field.id"  id="$field.id" size="#if ($components.size() > 3 ) 5 #else 3
#end">
```

For example, if you wanted to make it 15 (in the case where there are more than 3):

```
<select multiple name="$field.id"  id="$field.id" size="#if ($components.size() > 3 ) 15 #else
3 #end">
```

Adding a Description

From `<atlassian-jira/WEB-INF/classes/templates/jira/issue/field/components-edit.vm`, change:

```
>$textutils.htmlEncode($component.getString('name'))</option>
```

to:

```
>$textutils.htmlEncode($component.getString('name')) -
$textutils.htmlEncode($component.getString('description'))</option>
```

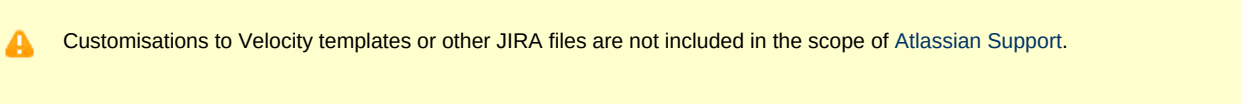
Make sure to back up the velocity file before changing it. Keep in mind the notes from Modifying JIRA Templates and JSPs.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

Changing the Size of the Fix Versions and Affects Versions Select List

This page describes how to increase the size of the Fix Version/s and Affects Version/s drop-down lists. Please see JRA-3028 for the full feature request.

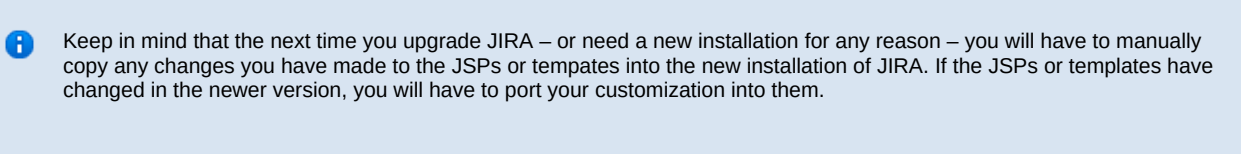


Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

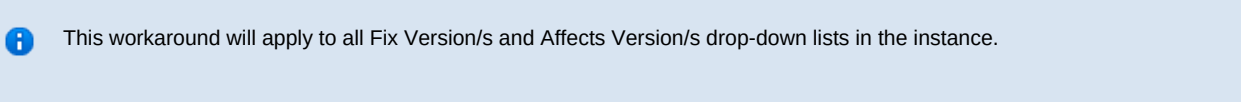
Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true



Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.



This workaround will apply to all Fix Version/s and Affects Version/s drop-down lists in the instance.

Increasing the size of the field

Edit `<atlassian-jira/WEB-INF/classes/templates/jira/issue/field/components-edit.vm`. Change the line:

```
<select multiple name="$field.id" size="#minSelectSize ($versions 1 6)" id="$field.id">
```

For example, if you wanted to make it 15 (in the case where there are more than 3):

```
<select multiple name="$field.id" size="#minSelectSize ($versions 1 15)" id="$field.id">
```

Make sure to back up the velocity file before changing it. Keep in mind the notes from Modifying JIRA Templates and JSPs.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

Changing the Size of the Text Area Custom Field

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

To work around the fixed size of a comment field, edit
`<jira-install>/atlassian-jira/WEB-INF/classes/templates/plugins/fields/edit/edit-textarea.vm`.

To change the size for all rows:

```
$!rendererParams.put("rows", "2")
```

To change the size for a particular custom field:

In this example, the custom field's id is 10220. You can get this value from editing the custom field and checking the URL.

```
#controlHeader ($action $customField.id $customField.name $fieldLayoutItem.required
$displayParameters.noHeader)

#if ($!customField.id=="customfield_10220")
  ## setup some additional parameters
  $!rendererParams.put("rows", "2")
  $!rendererParams.put("cols", "40")
  $!rendererParams.put("wrap", "virtual")

  ## let the renderer display the edit component
  $rendererDescriptor.getEditVM($!value, $!issue.key, $!fieldLayoutItem.rendererType,
$!customField.id, $!customField.name, $rendererParams, false)

#elseif ($!customField.isRenderable() && $rendererDescriptor)

  ## setup some additional parameters
  $!rendererParams.put("rows", "2")
  $!rendererParams.put("cols", "40")
  $!rendererParams.put("wrap", "virtual")

  ## let the renderer display the edit component
  $rendererDescriptor.getEditVM($!value, $!issue.key, $!fieldLayoutItem.rendererType,
$!customField.id, $!customField.name, $rendererParams, false)

#else

  <textarea name="$customField.id"
            id="$customField.id"
            class="textfield"
            rows="4" cols="40" wrap="virtual"
  >$textutils.htmlEncode($!value)</textarea>
#end

#controlFooter ($action $fieldLayoutItem.fieldDescription $displayParameters.noHeader)
```

There is a feature request to allow this cusomtisation from within JIRA at JRA-20248.

Changing the Temporary Directory

To move the temp directory, edit `<JIRA>/bin/catalina.sh`:

```
if [ -z "$CATALINA_TMPDIR" ] ; then
  # Define the java.io.tmpdir to use for Catalina
  CATALINA_TMPDIR="$CATALINA_BASE"/temp
fi
```

Replace the `"$CATALINA_BASE"/temp` with your temporary file directory.

Changing Usernames in JIRA

Currently JIRA does not have the ability to change usernames. Manually editing the database is not recommended due to the number of places the username is referenced (comments and filters for example).
The best option in this case is to edit the XML backup:

1. Create an XML Backup
2. Execute a global find and replace for the username
3. Restore JIRA with the new backup

If the username is a common word (e.g. admin), you may replace text that is not relevant to the user. So please be aware of this issue when performing the replace.

Configuring project specific security

We are often asked the following:

How do we configure the system so that a user/user group can only register/see issues on one specific project?

In order to configure the above please follow the below instructions and tweak as necessary for your organization:

1. Create a new Permission Scheme (Administration -> Schemes -> Permission Scheme) for Project External say **External_Permission_Scheme**
2. Create a new user group say **Group_External** (Administration -> Users & Groups -> Group Browser)
3. Add the External Users to that group
4. Associate **External_Permission_Scheme** to Project External (Administration -> Projects -> Project -> select Project External)

Note.

When users are created they are automatically a member of the jira-user group in order to allow them to login. The thing to note here is that the **Default Permission Scheme** grants users within the **jira-user group** certain permissions so those projects using the **Default Permission Scheme** will essentially give those users access to it.

To get around this either:

- Remove your external users from the **jira-user group** and give **Group_External** the ability to login by granting them the **global JIRA Users** permission (Administration -> Global Settings -> Global Permissions)
  OR
- Edit any Permission Schemes that grant the **jira-user group** specific permissions

It is also important to add that with the release of 3.7 and the introduction of Roles within JIRA it will not be necessary to create Groups for the above configuration.

For a detailed example using **Group Permissions** please see the following documentation: Using Project Level Security with User Groups

For a detailed example using **Project Role's** please see the following documentation: Using Project Level Security with Project Roles

Controlling project visibility

You can restrict project visibility to particular groups of users by using project permissions.

For example, if customers from Company X were put into the group "Cust-X" and given "Browse" permission for project Y, they will only be able to see Project Y (assuming you did not grant them the "Browse" permission for any other projects).

You should of course also give your developers permission to browse and operate on the project.

If you would like to restrict users to issues which they have created, set the "Browse Project" permission to be "Reporter." This way the user will have access to only the issues which they have created across all projects, but they will not be able to see any other issues.

You can also set security on an issue-by-issue basis. For more information on JIRA's Issue Level Security, please consult the documentation .

Using Project Level Security with Project Roles

This tutorial provides a step-by-step guide for creating project roles and using them in an issue security scheme. We recommend creating a test project and two test users for this tutorial.

Tutorial Goal

When completed, this tutorial will provide two issue security levels. One for issues that specific customers and your company can view, another for internal company eyes only.

1.Adding Project Roles

First we need to create project roles for our scheme to use:

Administration -> Users, Groups and Roles -> Project Role Browser -> Add Project Role

1. Create a project role called Customer A.
2. Create a project role called Customer B.
3. Create a project role called My Company.

In this example, the My Company project role will always have the same users/groups for each project. As a result, we'll set default members that will be used for all projects that use this scheme.

1. Next to the My Company project role, click Manage Default Members.
2. Add the users or groups for your company by clicking Edit next to the appropriate default (users or groups).

2. Adding an Issue Security Scheme

Next, we need to create the issue security scheme which will be hooked to our test project a bit later in this tutorial:

Administration -> Schemes -> Issue Security Schemes -> Add Issue Security Scheme

1. Add a new issue security scheme called Customers and My Company Issue Security Scheme.

3. Adding Issue Security Levels

We need to add security levels for this new issue security scheme. These levels will be available for selection to those that have permission to add issue security levels to issues. Users can only see levels of which they are members.

Administration -> Schemes -> Issue Security Schemes -> Click the Security Levels link next to Customers and My Company Issue Security Scheme

First we need to add a level for each customer:

1. Add a new issue security level called Customers and My Company.
2. Add the Customer A, Customer B and My Company project roles to this issue security level.
3. Click Default to make this the Default.

Next, we want a level for internal company eyes only:

1. Add another issue security level called My Company
2. Add the My Company project role to this issue security level.

4. Associating your Issue Security Scheme with a Project

Explain the step here and use the following syntax and color for menu notations:

Administration -> Project -> Projects -> Click on your test project name

1. For the Issue Security Scheme option, click Select.
2. Choose the Customers and My Company Issue Security Scheme from the list.
3. Click Next.
4. If you would like to associate existing issues with a security level, select it from the list, if not leave it at None.
5. Click Associate.

5. Adding project-specific members to a project role.

We need to specify the project-specific role members for the Customer A & B project roles.

Administration -> Project -> Projects -> Project Team -> Project Roles -> View Members

1. Add the first test user to the Customer A project role by clicking Edit in the Users column.
2. Add the second test user to the Customer B project role by clicking Edit in the Users column.

Project roles allow you to use the same permission scheme for multiple projects. We can change the members of project roles via the project!

Did it work?

1. Create one issue and set the Issue Security Level to My Company.
2. Create another issue and set the Issue Security Level to Customer A.
3. Create one more issue and set the Issue Security Level to Customer B.
4. Try logging in as each test user to ensure that they only see the appropriate issue.

Using Project Level Security with User Groups

This documentation is meant to give an in-depth analysis of Configuring project specific security — allowing full access to all projects for internal users, and limited access to external users by using JIRA groups and a project permission schemes. It is also possible to use Project Roles, but in this case we did not.

The example is based on the Atlassian Project Permission documentation. While that documentation tells you everything you can do, we get a lot of questions about how exactly to set your system up to have two or more classes of users:

- Internal users (such as employees at your company) who have full permission
- External users (such as customer at your company) who have limited permission

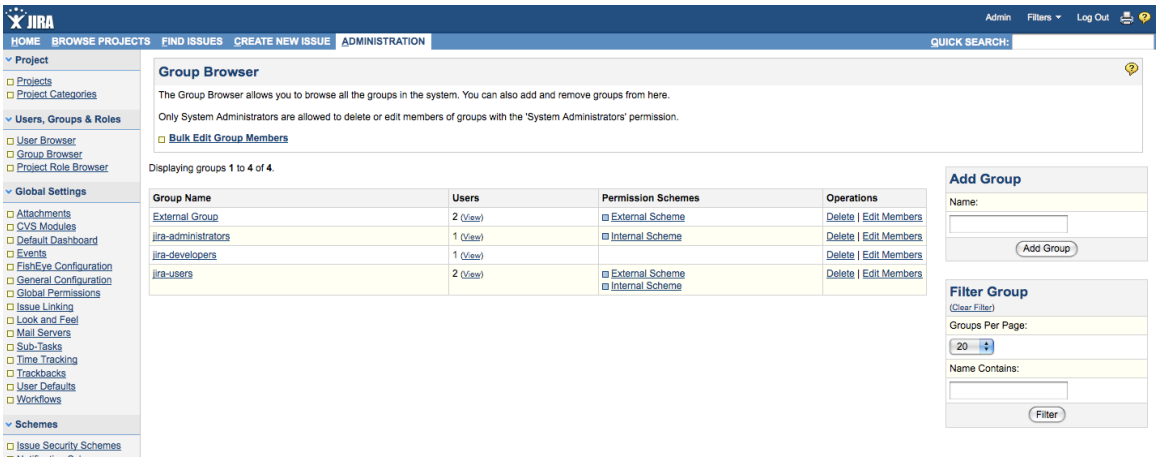
Usually, though, in order to accomplish a security configuration which fits your company exactly, it will require a good amount of time, effort, and imagination on your part. At the moment JIRA is only able to support security at a project level or issue level. **Currently there is no field level security available.**

The first step for project level security is to define user groups. In this case a group called "external group" was created. All internal users will just be in the default "jira-users" group. In a default JIRA instance, when a user is created they will automatically be put into the jira-users group. Anyone who is external will have to be manually assigned to the external group and be removed from the jira-users group. There is no way to automatically assign users to certain groups without massive customizations to the JIRA environment. The reason for taking the approach of assigning all internal users to the jira-users group, is because this documentation is assuming that clients already have many internal users. Assigning a small group of users to one group as opposed to reassigning hundreds or thousands of users is easier.

If starting from scratch, it is better to define and assign groups new groups from the beginning. For example, an "internal group" as well as an

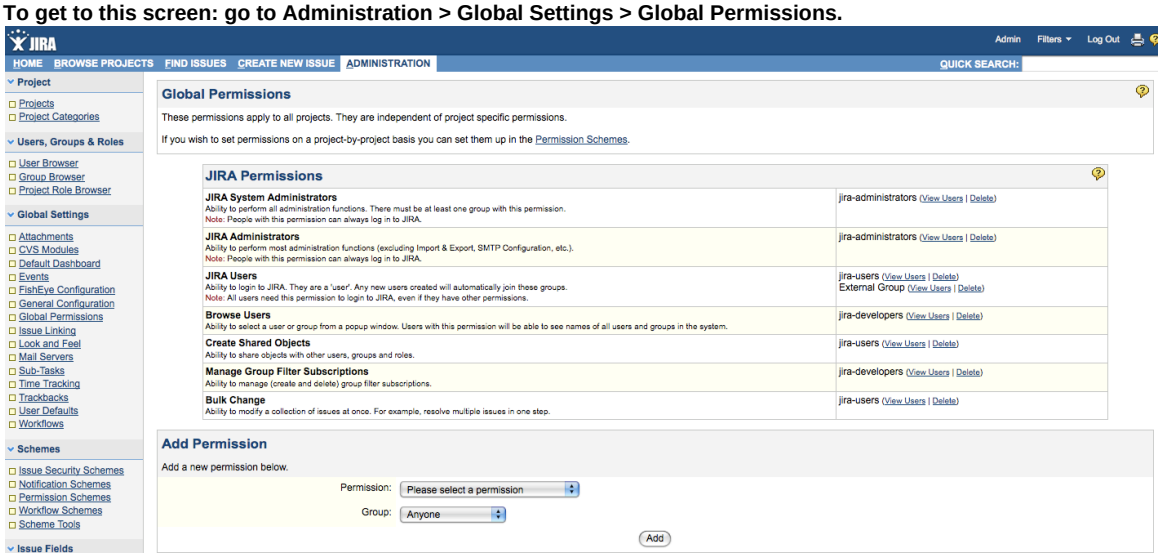
external group. But, in this example we will just look at jira-users and the external group. See the group settings in the image below for more detail:

To get to this screen: go to Administration > Users, Groups & Roles > Group Browser.



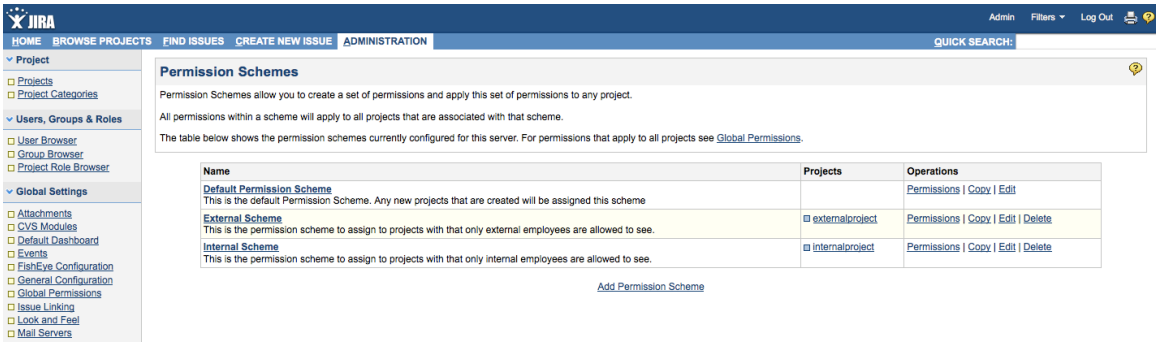
Now make sure that the External Group is added to the global JIRA Users permission so that they have access to JIRA. All users must be in the global JIRA Users group in order to access JIRA. **Note: The JIRA Users group is different from the jira-users group. JIRA Users is global while jira-users is group specific.** See the image below for more detail.

To get to this screen: go to Administration > Global Settings > Global Permissions.



After creating the desired groups, separate permission schemes for each group need to be made. In the below image two schemes were created; an internal scheme and an external scheme. Obviously the internal scheme is for internal users and the external scheme is for external users. If your company has multiple users from multiple companies, you will need to make multiple schemes and groups for each project.

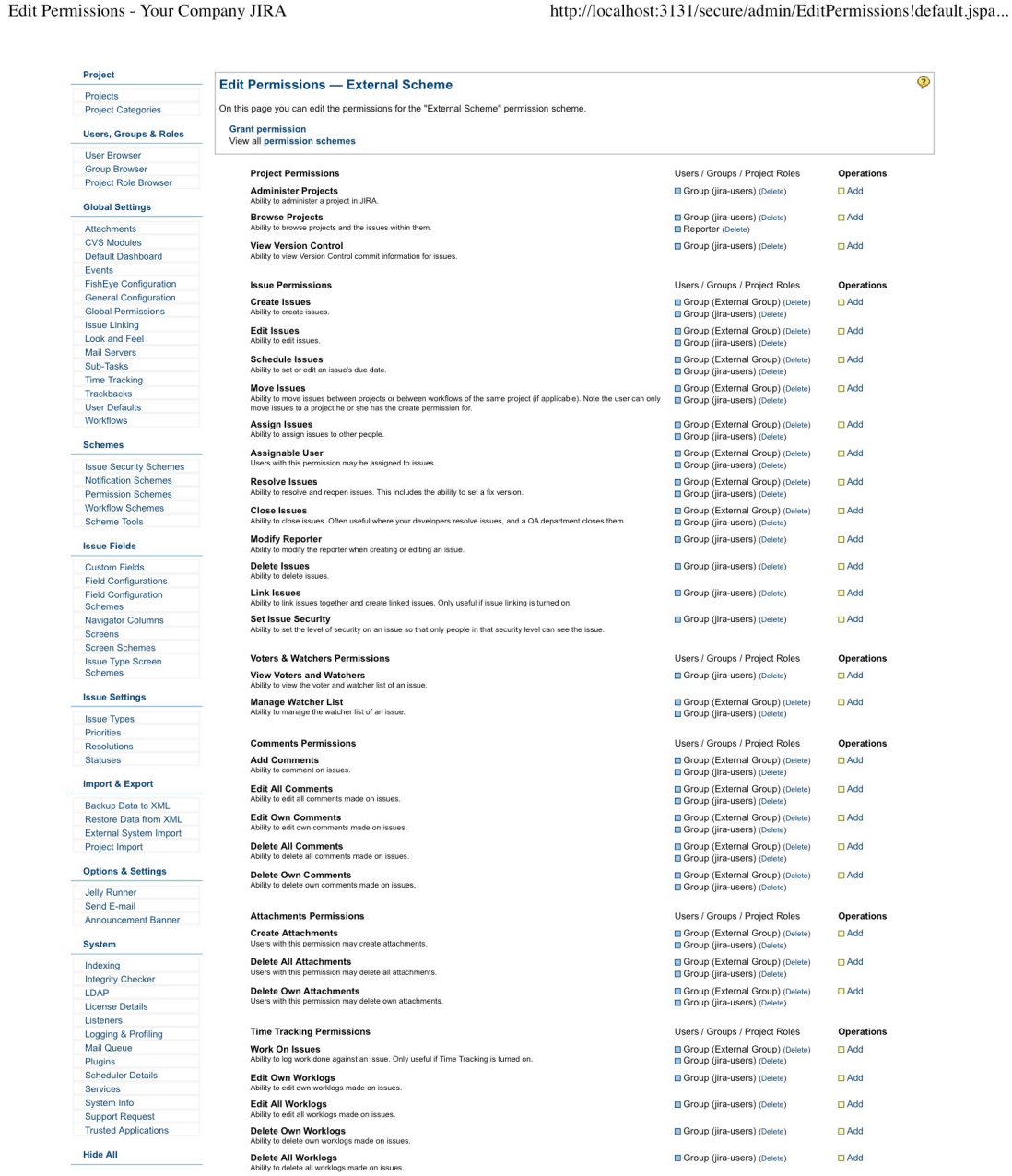
To get to this screen: go to Administration > Schemes > Permission Schemes.



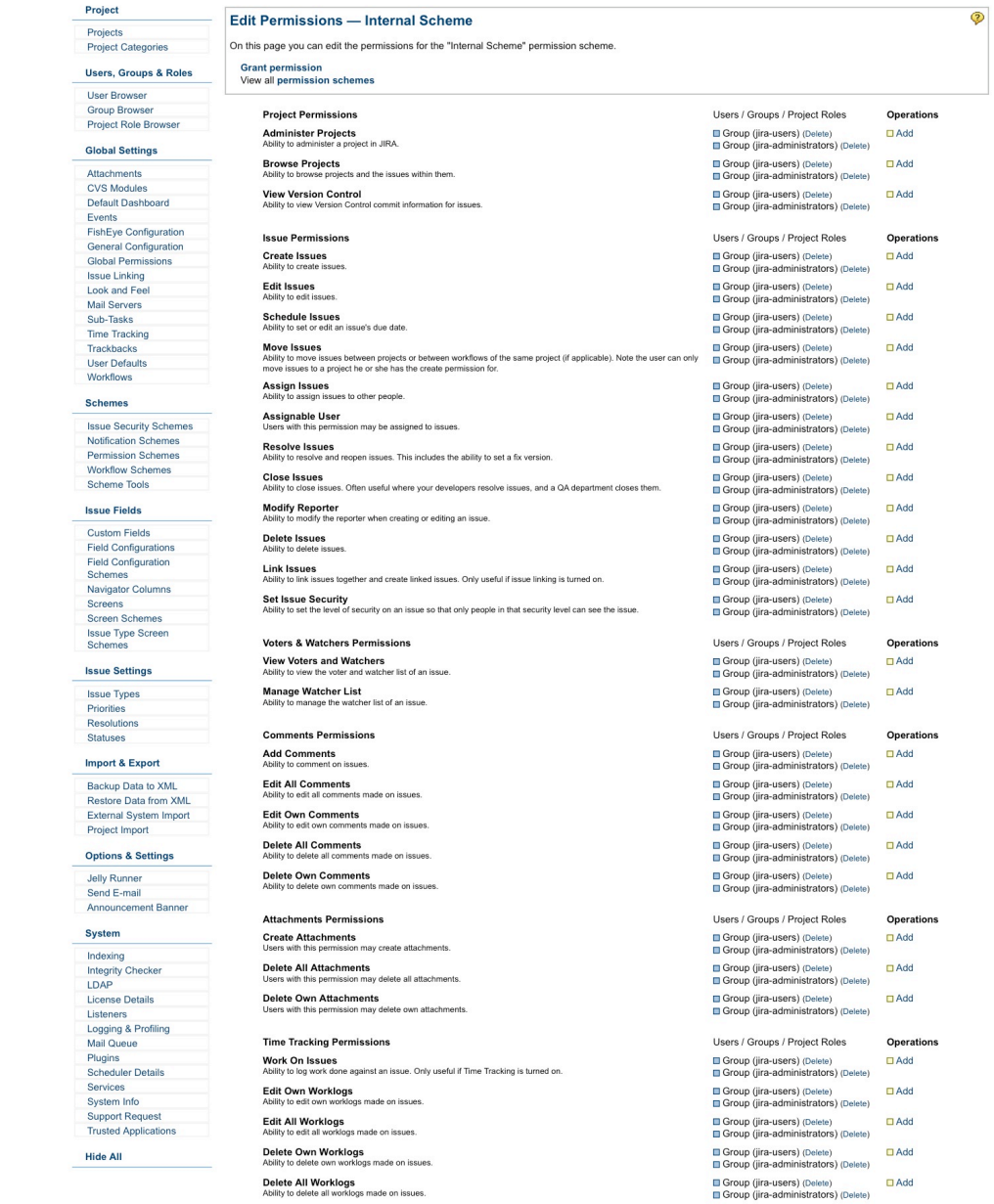
After the schemes have been created, they must be tailored to meet your needs. For example: In the external scheme attach below, jira-users are given all permissions, while the External Group is given limited rights. Both groups must be present in this permission scheme to ensure that both internal users and external users have access to whichever project this scheme is assigned to. Only jira-users should be assigned to the Internal Scheme. See images below for more detail. **Please note that in the External Permission Scheme the "Browse**

Projects" category has both jira-users and reporters (rather than External Group). This was done so external users can only see tickets they have created in the External Project and not others tickets. However, if "Reporter" is replaced with "External Group" then the External Group users will be able to view all tickets associated with the project..

To get to this screen: go to Administration > Schemes > Permission Schemes > Click on External Scheme.

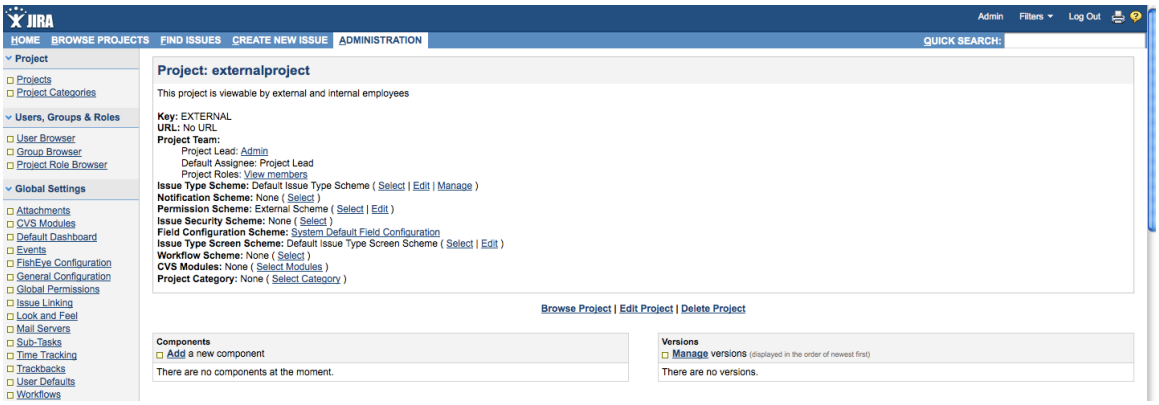


To get to this screen: go to Administration > Schemes > Permission Schemes > Click on Internal Scheme.

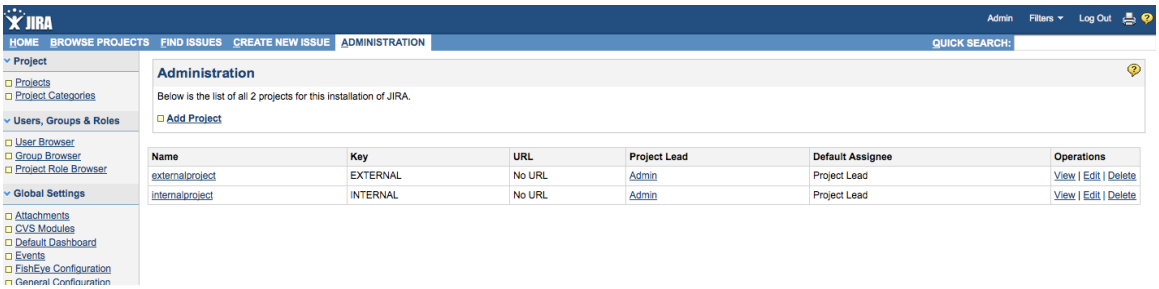


Now assign the appropriate permission scheme to the appropriate project. For this example the internal scheme will be assigned to the the internal project and the external scheme will be assigned to the external project. See the images below for more detail:

To get to this screen: go to Administration > Project > Projects > Click on External Project.

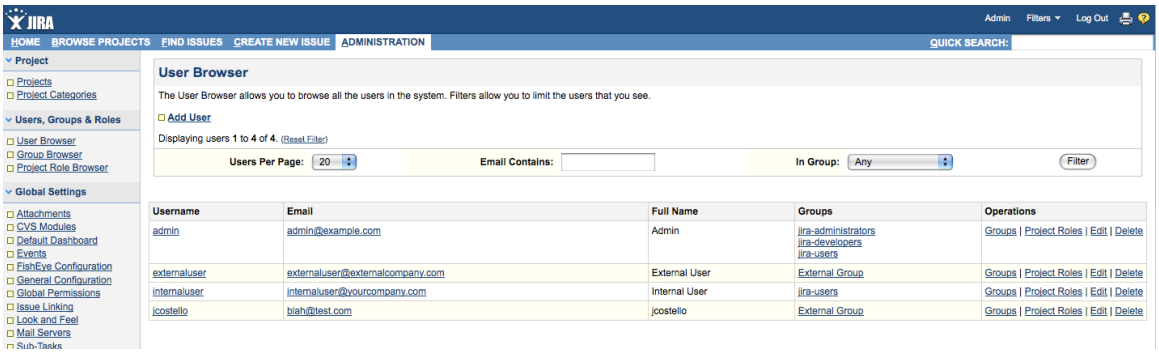


To get to this screen: go to Administration > Project > Projects > Click on Internal Project.



Once the above steps have been completed create users and add them to the appropriate group as seen in the image below. Note: When users are created will automatically be created belonging to jira-users and External Group. THe administrator will be responsible for manually removing the users from groups that the user should not belong to.

To get to this screen: go to Administration > Users, Groups & Roles > User Browser.



If done correctly the internal employees(jira-users) will have access to all projects, while the external users will only have access to their projects. Feel free to download the XML backup of this example on your local test instance.

The sample file

- Please ensure you have backed up your existing JIRA instance
- You can download the JIRA helpdesk sample file here: ExampleSecurity.zip
- Restore the sample data file. You can learn how to restore a file here

User list and logins

- All user passwords are the same: **admin**
- The main username to login with is: **admin**
  - Full JIRA admin rights
  - Access to all projects

- Internal users are: **internaluser**
- These users are in the group: **jira-users**
- Access to both the Internal and External Projects and all issues.

- External users are: **externaluser** and **jcostello**
- These users are in the group: **External Group**
- Access to External Project and Issues Created only.

Connecting to SSL services

On this page:

- Problem symptoms
- The cause
- The fix
  - Obtain the server's public key.
  - Import the public key.
  - Restart the app server
  - Note: alternative keystore locations
  - Debugging
    - Using Java
    - Using openssl
- See also

This page describes how to get web applications like JIRA and Confluence connecting to external servers over SSL, via the various SSL-wrapped protocols. For instance, you may want to:

- Refer to an `https://...` URL in a Confluence macro.
- Use an IMAPS server to retrieve mail in JIRA.
- Use SMTP over SSL (SMTPS) to send mail in JIRA.
- Connect to a LDAP directory over SSL.
- Set up **Trusted Applications** over SSL.

If you want to run JIRA *itself* over SSL, see Running JIRA over SSL or HTTPS.

Problem symptoms

Simply entering the 'https' URL, or specifying IMAPS in JIRA will result in odd java.net.ssl.* exceptions in the logs, for example:

```
javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target
at com.sun.mail.imap.IMAPStore.protocolConnect(IMAPStore.java:441)
at javax.mail.Service.connect(Service.java:233)
at javax.mail.Service.connect(Service.java:134)
....
```

The cause

The problem is that our webapp is now acting as a SSL client, and as a client, it needs to obtain and 'trust' the server's *public key*.

This is identical to what happens when you visit a https://... URL in a browser - the browser fetches the public key and (if not signed by a trusted agent) presents it to you for inspection. If you trust the key, the browser saves it, and uses it to encrypt all subsequent communication with the site. We need to emulate this process before our webapp can access https resources.

The fix

Obtain the server's public key.

To quote Microsoft; "consult your system administrator". The public/private key pair will live somewhere on the server. The public key should be located and copied to the server hosting JIRA/Confluence. For example:

```
scp root@mail.yourcompany.com:/etc/ssl/certs/imapd.pem .
```

If you have openssl installed locally, the key can be retrieved with a command like:

```
jturner@teacup:~$ openssl s_client -connect imap.atlassian.com:imaps
CONNECTED(00000003)
depth=0 /C=AU/ST=NSW/L=Sydney/O=Atlassian/CN=imap.atlassian.com/emailAddress=info@atlassian.com
.....
.....
Server certificate
-----BEGIN CERTIFICATE-----
MIICiTCCAfKgAwIBAgIBADANBgkqhkiG9w0BAQQFADB/MQswCQYDVQQGEwJBVTEM
MAoGA1UECBMDTlNXMQ8wDQYDVQQHEwZTeWRuZXkxEjAQBgNVBAoTCUF0bGFzc2lh
bjEaMBgGA1UEAxMRY3ZzLmF0bGFzc2lhbi5jb20xITAfBgkqhkiG9w0BCQEWEmlu
Zm9AYXRsYXNzaWFuLmNvbTAeFw0wNTA5MjMwNjUyNTNaFw0wNjA5MjMwNjUyNTNa
MH8xCzAJBgNVBAYTAkFVMQwwCgYDVQQIEwNOU1cxDzANBgNVBAcTBlN5ZG5leTES
MBAGA1UEChMJQXRsYXNzaWFuMRowGAYDVQQDExFjdnMuYXRsYXNzaWFuLmNvbTEh
MB8GCSqGSIb3DQEJARYSaW5mb0BhdGxhc3NpYW4uY29tMIGfMA0GCSqGSIb3DQEB
AQUAA4GNADCBiQKBgQDhwAgx/gDgKe9tBjUCj7JtVkwQSzj2Dq0PHiJu1AWUYWFW
ivbBWaWSYbt/w9vIRSL8OlGVOLnlFOH5o7QIpIBZvd3xBMv6DxMijM86/hu8QTPt
KcMuqBTGpu1T846SzNncj883wSE1hXxezCgEFCsqyC7dVX4l0Ay6zgzkt2wc3QID
AQABoxUwEzARBglghkgBhvhCAQEEBAMCBkAwDQYJKoZIhvcNAQEEBQADgYEAJOgg
O4brCcQa3IgONo8UmLcHo6Rq+Py6ZA3ueUegy/uyQ358JUeL4kktXuYL9gAPCuMc
hsC1iyaOrWY/S9S67w2ZWqc+uYX9ophFHkxK1r3YiaiMpEzMyB12VWSrOITcR0LV
7NTWfxfPLUpkDbj+Mw/66QJkI0lqBvcKn3KXI74=
-----END CERTIFICATE-----
```

Cut and paste the certificate (including BEGIN and END lines) into a local file (eg. imapd.pem).

Import the public key.

To do this, you need to use the `keytool` program that comes with the Java platform used to run JIRA.

The instructions in the remainder of this section assume you are using a JIRA installation installed from an 'archive' or the JIRA WAR distribution. If you installed JIRA using the automated 'Windows' or 'Linux' installers, please enter the `jre\bin` (or `jre/bin`) subdirectory of your JIRA Installation Directory when running the `keytool` command.

Assuming you are using a JIRA installation installed from an 'archive' or the JIRA WAR distribution, change directory to `$JAVA_HOME/bin` and then run the following:

```
jturner@teacup:~$ sudo keytool -import -alias mail.yourcompany.com -keystore
$JAVA_HOME/jre/lib/security/cacerts -file imapd.pem
Enter keystore password:  changeit
Owner: EMAILADDRESS=info@atlassian.com, CN=atlassian.com, O=Atlassian, L=Sydney, ST=NSW, C=AU
Issuer: EMAILADDRESS=info@atlassian.com, CN=atlassian.com, O=Atlassian, L=Sydney, ST=NSW, C=AU
Serial number: 0
Valid from: Fri Feb 11 14:09:05 EST 2005 until: Sat Feb 11 14:09:05 EST 2006
Certificate fingerprints:
MD5:  CB:AE:7D:5D:1A:08:06:77:93:3B:0F:53:BB:40:C0:D4
SHA1: 7C:02:44:0D:A9:8F:F9:FB:BB:7B:C6:F1:52:DE:CA:00:17:D9:3A:A0
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

This will import the public key (imapd.pem) into Java's default keystore, and marks it as trusted.

On Windows the command is similar, eg.:

```
C:\Program Files\Java\jre1.6.0_05>bin\keytool -import -file c:\certs\imapd.pem -alias
mail.yourcompany.com -keystore lib\security\cacerts
Enter keystore password:
Owner: CN=*.atlassian.com, OU=IT, O=ATLASSIAN SOFTWARE SYSTEMS PROPRIETARY LIMITED, L=Sydney,
ST=NSW, C=au
Issuer: CN=DigiCert Global CA, OU=www.digicert.com, O=DigiCert Inc, C=US
Serial number: a2d7047dc5d47ba988c9685e1efb860
Valid from: Thu Jan 10 11:00:00 EST 2008 until: Fri Jan 14 10:59:59 EST 2011
Certificate fingerprints:
         MD5:  9D:B4:9F:3D:0A:DE:6A:BD:BC:3D:95:BE:60:BD:70:02
         SHA1: 67:C6:E9:C8:3F:F1:7A:3C:66:E2:CE:62:78:A1:66:84:35:5E:62:1E
         Signature algorithm name: SHA1withRSA
         Version: 3
.....

Trust this certificate? [no]:  yes
Certificate was added to keystore

C:\Program Files\Java\jre1.6.0_05>
```

Restart the app server

Restart, and if everything is correct, your webapp should now connect to the SSL resource without problems.

Note: alternative keystore locations

Java will normally use a system-wide keystore in `$JAVA_HOME/jre/lib/security/cacerts`, but it is possible to use a different keystore by specifying a parameter, **-Djavax.net.ssl.trustStore=/path/to/keystore**, where '/path/to/keystore' is the absolute file path of the alternative keystore.

Setting this **is not recommended**, however, because if Java is told to use a custom keystore (eg. containing a self-signed certificate), then Java will not have access to the root certificates of signing authorities found in `$JAVA_HOME/jre/lib/security/cacerts`, and accessing most CA-signed SSL sites will fail. It is better to add new certificates (eg. self-signed) to the system-wide keystore (as above).

There is also a per-user truststore (~/.keystore -- at least on Linux), but its contents do not appear to be logically appended to those in the system-wide keystore; ie. it is entirely separate, and only used if one specifies `-Djavax.net.ssl.trustStore=/home/<user>/.keystore`. This has the same disadvantage described above with custom keystores, so the per-user truststore is best avoided.

Debugging

Problems are one of two forms:

- Java is not referring to the correct keystore.
- The keystore does not contain the certificate of the SSL service you're connecting to.

Using Java

The attached SSLPoke.class Java program (source) is useful for debugging. It simply connects to a SSL service, sends a byte of input, and watches the output. For instance, connecting to a local HTTPS server on port 443 (the HTTPS default) with a untrusted (self-signed) certificate:

```
jturner@psyche:~$ java SSLPoke localhost 443
sun.security.validator.ValidatorException: PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification
path to requested target
        at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:285)
        at sun.security.validator.PKIXValidator.engineValidate(PKIXValidator.java:191)
        at sun.security.validator.Validator.validate(Validator.java:218)
        at
com.sun.net.ssl.internal.ssl.X509TrustManagerImpl.validate(X509TrustManagerImpl.java:126)
        at
com.sun.net.ssl.internal.ssl.X509TrustManagerImpl.checkServerTrusted(X509TrustManagerImpl.java:209)
at
com.sun.net.ssl.internal.ssl.X509TrustManagerImpl.checkServerTrusted(X509TrustManagerImpl.java:249)
at com.sun.net.ssl.internal.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:954)
        at
com.sun.net.ssl.internal.ssl.ClientHandshaker.processMessage(ClientHandshaker.java:123)
        at com.sun.net.ssl.internal.ssl.Handshaker.processLoop(Handshaker.java:511)
        at com.sun.net.ssl.internal.ssl.Handshaker.process_record(Handshaker.java:449)
        at com.sun.net.ssl.internal.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:817)
        at
com.sun.net.ssl.internal.ssl.SSLSocketImpl.performInitialHandshake(SSLSocketImpl.java:1029)
        at com.sun.net.ssl.internal.ssl.SSLSocketImpl.writeRecord(SSLSocketImpl.java:621)
        at com.sun.net.ssl.internal.ssl.AppOutputStream.write(AppOutputStream.java:59)
        at com.sun.net.ssl.internal.ssl.AppOutputStream.write(AppOutputStream.java:73)
        at SSLPoke.main(SSLPoke.java:28)
Caused by: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid
certification path to requested target
        at
sun.security.provider.certpath.SunCertPathBuilder.engineBuild(SunCertPathBuilder.java:174)
        at java.security.cert.CertPathBuilder.build(CertPathBuilder.java:238)
        at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:280)
        ... 15 more
```

and connecting to a CA-verified certificate:

```
jturner@psyche:~$ java SSLPoke mail.atlassian.com 443
Successfully connected
jturner@psyche:~$
```

Similarly you would test port 636 to test LDAPS connections.

Make sure that the version of Java you are using is the same as that used in your production Java application. On Unix systems, 'ps -ef | grep java' will show the full command for Java processes. Check for the presence of a `-Djavax.net.ssl.trustStore` parameter. If `-Djavax.net.ssl.trustStore` is present in the command, this may well be the cause of your problems (see discussion above). You can verify whether the `-Djavax.net.ssl.trustStore` parameter is causing problems by running the `SSLPoke` test utility above with it, eg:

```
java -Djavax.net.ssl.trustStore=/my/custom/keystore SSLPoke localhost 443
```

If this fails (confirming the problem), the solution is to remove the `-Djavax.net.ssl.trustStore` parameter, import your custom keystore certificates into the main keystore with `keytool -importkeystore -srckeystore /my/custom/keystore -destkeystore $JAVA_HOME/jre/lib/security/cacerts`, and restart the application.

If you are sure the certificate is trusted and found by Java, and you are having low-level SSL problems, you can get debug information in the stdout logs by setting the `-Djavax.net.debug=all` property.

Using openssl

The `openssl` commands are very useful for debugging SSL problems. For instance, to print the server's certificate:

```
jturner@psyche:~$ openssl s_client -connect localhost:443 2>/dev/null
CONNECTED(00000003)
---
Certificate chain
 0 s:/C=AU/ST=NSW/L=Sydney/O=Atlassian/OU=Support/CN=localhost/emailAddress=jeff@atlassian.com
   i:/C=AU/ST=NSW/L=Sydney/O=Atlassian/OU=Support/CN=localhost/emailAddress=jeff@atlassian.com
---
Server certificate
-----BEGIN CERTIFICATE-----
MIICizCCAfQCCQCt7NSYJaxDETANBgkqhkiG9w0BAQUFADCBiTELMAkGA1UEBhMC
QVUxDDAKBgNVBAgTA05TVzEPMA0GA1UEBxMGU3lkbmV5MRIwEAYDVQQKEwlBdGxh
c3NpYW4xEDAOBgNVBAsTB1N1cHBvcnQxEjAQBgNVBAMTCWxvY2FsaG9zdDEhMB8G
CSqGSIb3DQEJARYSamVmZkBhdGxhc3NpYW4uY29tMB4XDTA4MDEwMzAwMTEzMVoX
DTA4MDIwMjAwMTEzMVowgYkxCzAJBgNVBAYTAkFVMQwwCgYDVQQIEwNOU1cxDzAN
BgNVBAcTBlN5ZG5leTESMBAGA1UEChMJQXRsYXNzaWFuMRAwDgYDVQQLEwdTdXBw
b3J0MRIwEAYDVQQDEwlsb2NhbGhvc3QxITAfBgkqhkiG9w0BCQEWEmplZmZAYXRs
YXNzaWFuLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEApOEOBbKKj189
15w078TUsR68gOahr6KiXg3s2zOyVaFbAepTnvhgtzej/qOJ8ESU5quCNIeBALov
PRyxUyySBLOV14i3vrhqugQLKeYUiCIehri2iRaU+/6FsX8kIgfSBIMu75uV0bnM
9A6t1zZizSdcP+1k2DzaIc8sXzkpJ2sCAwEAATANBgkqhkiG9w0BAQUFAAOBgQCA
jm1xW5DvCYvHCa76QIwmsEIxCvya2zEqiwhAYC9gLbuQ1kjFgpn17U9x2ltJ1jWE
/5BQOeRk+cBaVVZJpfrKAUZoRczsNIQcaDs3JEUrZ6qBUxXheT8MijXhhhnKN0Fl
CvOuhw5DsgtEa1MT3CWetf9XFD3K4NI5+X7ihehs9g==
-----END CERTIFICATE-----
subject=/C=AU/ST=NSW/L=Sydney/O=Atlassian/OU=Support/CN=localhost/emailAddress=jeff@atlassian.com i
```

(add `-showcerts` to print all the certificates in the chain)

save it to a local file:

```
jturner@psyche:~$ cat > localhost.pem
-----BEGIN CERTIFICATE-----
MIICizCCAfQCCQCt7NSYJaxDETANBgkqhkiG9w0BAQUFADCBiTELMAkGA1UEBhMC
QVUxDDAKBgNVBAgTA05TVzEPMA0GA1UEBxMGU3lkbmV5MRIwEAYDVQQKEwlBdGxh
c3NpYW4xEDAOBgNVBAsTB1N1cHBvcnQxEjAQBgNVBAMTCWxvY2FsaG9zdDEhMB8G
CSqGSIb3DQEJARYSamVmZkBhdGxhc3NpYW4uY29tMB4XDTA4MDEwMzAwMTEzMVoX
DTA4MDIwMjAwMTEzMVowgYkxCzAJBgNVBAYTAkFVMQwwCgYDVQQIEwNOU1cxDzAN
BgNVBAcTBlN5ZG5leTESMBAGA1UEChMJQXRsYXNzaWFuMRAwDgYDVQQLEwdTdXBw
b3J0MRIwEAYDVQQDEwlsb2NhbGhvc3QxITAfBgkqhkiG9w0BCQEWEmplZmZAYXRs
YXNzaWFuLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEApOEOBbKKj189
15w078TUsR68gOahr6KiXg3s2zOyVaFbAepTnvhgtzej/qOJ8ESU5quCNIeBALov
PRyxUyySBLOV14i3vrhqugQLKeYUiCIehri2iRaU+/6FsX8kIgfSBIMu75uV0bnM
9A6t1zZizSdcP+1k2DzaIc8sXzkpJ2sCAwEAATANBgkqhkiG9w0BAQUFAAOBgQCA
jm1xW5DvCYvHCa76QIwmsEIxCvya2zEqiwhAYC9gLbuQ1kjFgpn17U9x2ltJ1jWE
/5BQOeRk+cBaVVZJpfrKAUZoRczsNIQcaDs3JEUrZ6qBUxXheT8MijXhhhnKN0Fl
CvOuhw5DsgtEa1MT3CWetf9XFD3K4NI5+X7ihehs9g==
-----END CERTIFICATE-----
jturner@psyche:~$
```

We can now calculate the fingerprint of the certificate with `openssl x509`:

```
jturner@psyche:~$ openssl x509 -fingerprint -md5 -noout -in localhost.pem
MD5 Fingerprint=29:37:D6:C3:03:B6:0B:29:76:55:B5:35:E0:BE:21:5C
jturner@psyche:~$
```

and verify that this fingerprint matches what is in Java's keystore:

```
jturner@psyche:~$ keytool -keystore /usr/lib/jvm/java-6-sun/jre/lib/security/cacerts -list |
grep -A2 localhost
Enter keystore password:  changeit
localhost, 27/06/2008, trustedCertEntry,
Certificate fingerprint (MD5): 29:37:D6:C3:03:B6:0B:29:76:55:B5:35:E0:BE:21:5C
verisignclass3ca, 28/10/2003, trustedCertEntry,
jturner@psyche:~$
```

See also

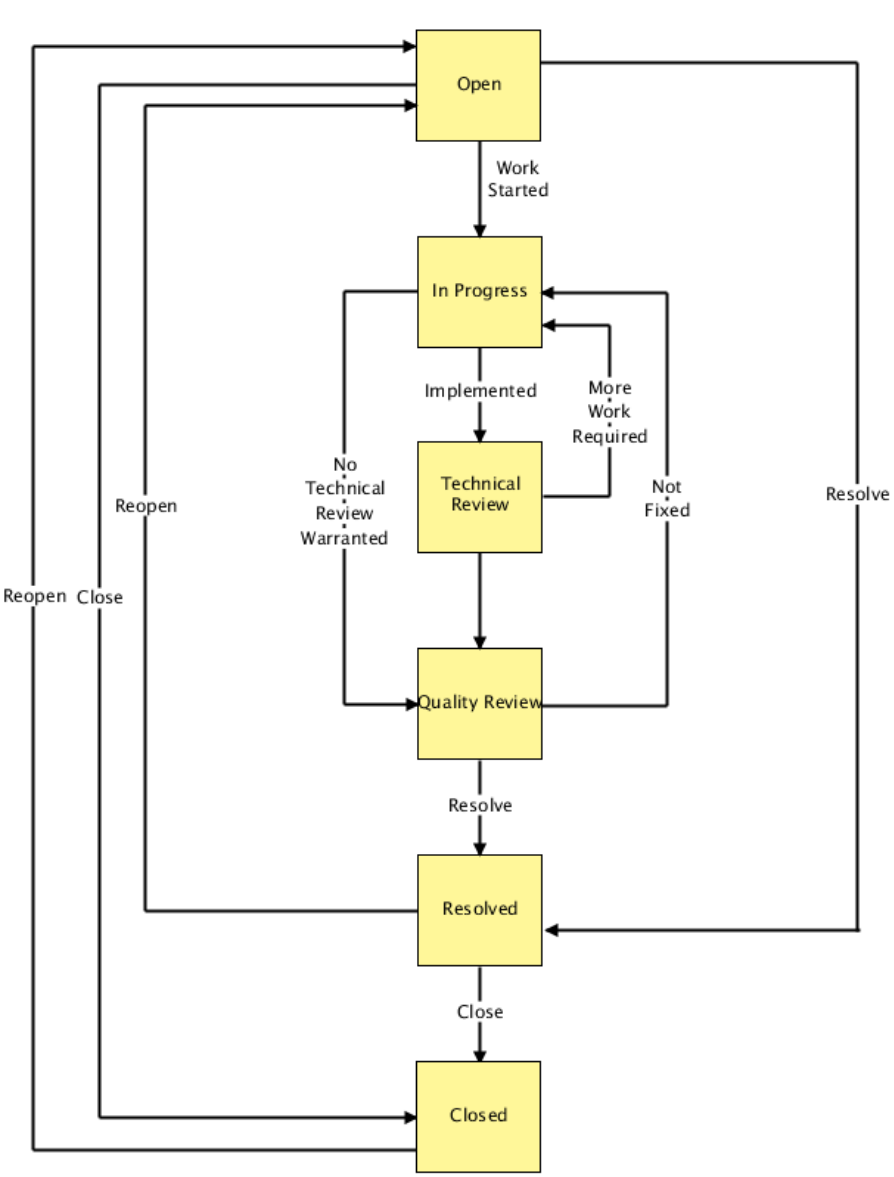
- Configuring an SSL Connection to Active Directory

Creating a Custom Workflow

Customised workflows are one of the key features within JIRA. Yet I often hear users mention that JIRA workflows are complex to implement and hard to understand. A lot of this sentiment stems from the sheer flexibility of JIRA's workflow functionality. As a JIRA QA Engineer, I can vouch that once you know how to use JIRA workflows, you will love them.

Creating a customised workflow allows JIRA to reproduce specific internal processes. At a very high level, workflows can be customised for different projects and issue types. You could for example have 'Support Requests' follow one custom workflow while 'Feature Requests' follow an entirely independent workflow. In this tutorial, I will be creating a copy of the default workflow of JIRA and customising it by adding some steps to it.

Whenever you implement a new customised workflow, it is always best to start by creating a visual representation of your workflow, ideally in the form of a flowchart. I am going to do this using Confluence, together with a diagramming plugin called Gliffy. You can see the flowchart I created here:



As you may have picked up, the workflow I am creating is for use in a software development team. In this example, I want to ensure that once development has checked-in their code, then we can conduct a 'Technical Review' (for conducting a code review in this case), and then move on to a 'Quality Review' (for testing by the quality assurance team). These are both going to be new steps within the existing default JIRA workflow.

Ensuring you have JIRA Administration privileges, head over to the JIRA Administration tab. Here you will see links to 'Workflows' and 'Workflow Schemes'. 'Workflows' let you define specific workflows whereas 'Workflow Schemes' allow you to map one or more specific workflows to certain issue types and in turn to certain projects.

To demonstrate the power of JIRA, I'm also going to add in some custom screens to display specific fields (including custom fields) within specific stages of the workflow. The tasks that I am going to focus on are:

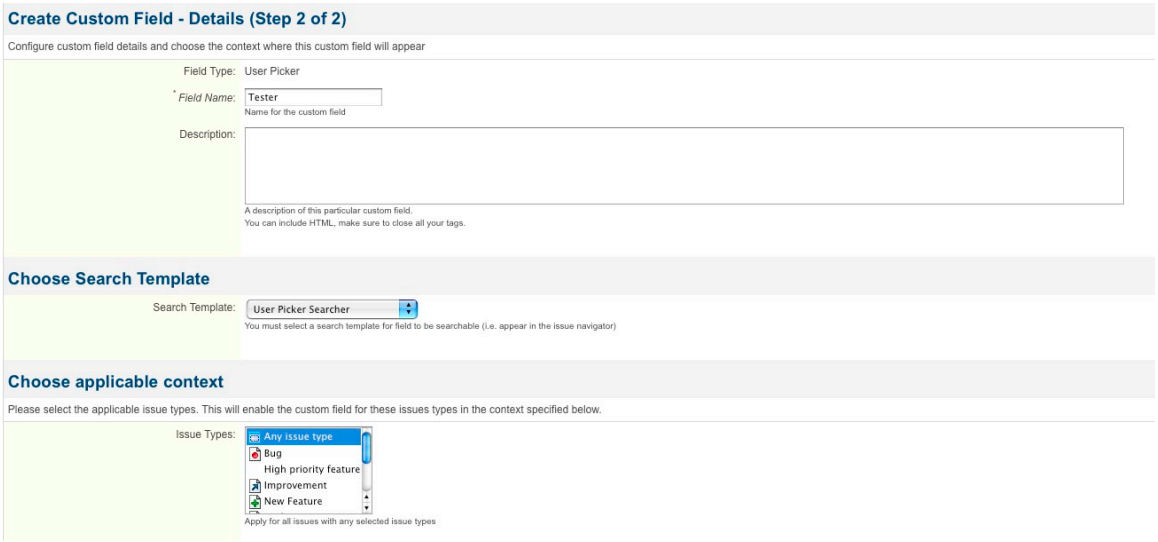
- **Add Custom Field:** I'm going to add a custom field allowing us to select a specific user or 'Tester' to conduct our Quality Review.
- **Add a Custom Screen:** I'll create a new screen to ask users to select a user as the Tester, displaying the 'Tester' custom field we are adding.
- **Add New Status:** To reflect the new statuses our issue can be in, I'll add a status for 'Technical Review' and another for 'Quality Review'.

Step 1: Add Custom Field

Let's add our own customised 'User Picker' field called 'Tester' to allow us to select a specific person for testing in the QA step.

Click the '**Custom Fields**' option under 'Issue Fields' in the 'Administration' tab.

Create a new custom field of type 'User Picker' and provide a 'Field Name' of 'Tester'. Leave other fields at their displayed defaults and click on 'Finish'. When presented with the 'Associate field Tester to screens' page, just click on 'Update' — do not select any screens at this stage (we're going to add a new screen in the next step). Let's repeat that process exactly but this time name this 'User Picker' custom field 'Reviewer'.

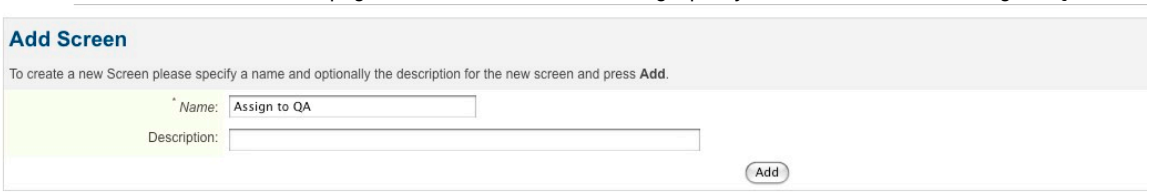


Step 2: Add New Screen:

We can now configure JIRA to create a screen for displaying our new field.

Click on '**Screens'** link under "Issue Fields" in the JIRA Administration area.

Towards the bottom of the Screens page, within the 'Add Screen' dialog, specify a new screen named 'Assign to QA' and 'Add' it.



Now 'Assign to QA' is added in the screen list. Now click 'Configure' for that screen in the right most column named 'Operations'.

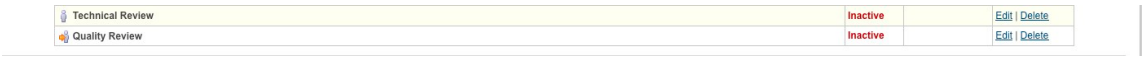
You will now be presented with the 'Configure Screen' page. Under 'Add Field', multi-select the fields named 'Fix Version' & 'Tester', and click 'Add'.

Repeat this process, specifying a new screen named 'Assign for Technical Review', configured to display the fields 'Fix Version' & 'Reviewer'.

Step 3: Add New Status:

We'll now add two new statuses that our issues can move through in our new workflow.

Click on '**Statuses**' under 'Issue Settings' and 'Add new status' named 'Technical Review'. If you wish, you can change the icon for the new status by clicking on 'select image'. Let's repeat the process for our second status, this time, 'Add new status' named 'Quality Review'.

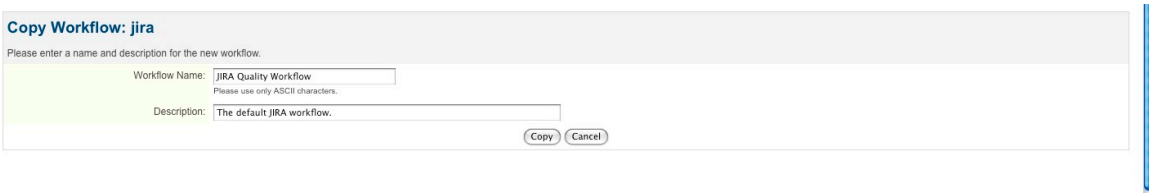


Step 4: Workflows:

We have now completed all the prerequisites for my workflow. We'll now create the new workflow that will incorporate all the changes we have made so far.

It's recommended, especially for new students of JIRA workflow configuration, that you copy an existing workflow and then start editing it, rather then creating one from scratch.

Click on the '**Workflows**' link under 'Global Settings'. Find the **jira** workflow and select 'Copy' from the rightmost 'Operations' column. Edit the 'Workflow Name' to 'JIRA Quality Workflow' and edit the description to something appropriate.



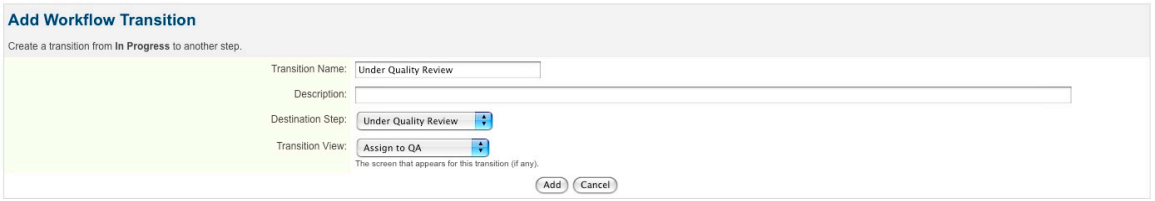
We'll add two new steps, 'Technical Review' and 'Quality Review', and choose the appropriate status for each.

Click on 'Steps', again in the 'Operations' column. In the 'Add New Step' dialog, add a 'Step Name' called 'Technical Review' and select the matching 'Linked Status', 'Technical Review'. We'll repeat this process, this time adding a new step called 'Quality Review' with a matching 'Linked Status' of 'Quality Review'.

Our next step is to add/modify the transitions according to our original flowchart.

I will be adding a transition to the 'In Progress' step, to reflect the new options that will be available to a user from the In Progress step (moving to a Technical Review). While adding transitions you can specify a transition view also — a screen presented to the user when they click on a specific workflow action or 'transition'. We'll also add transitions to our 2 new steps ('Technical Review' & 'Quality Review').

For the 'In Progress' workflow, select 'Add Transition' under the 'Operations' column. Add a 'Transition Name' of 'Conduct Technical Review', leave the 'Description' field blank, specify a 'Destination Step' of 'Technical Review' and finally set the 'Transition Step' to 'Assign for Technical Review'.



Let's add two transitions for the Technical Review using these settings:

- Transition Name: More Work Required, Description: leave blank, Destination Step: In Progress, Transition View: No view for this transition
- Transition Name: Proceed to Quality Review, Description: leave blank, Destination Step: Quality Review, Transition View: Assign to QA

We'll also add two transitions for the Technical Review using these settings:

- Transition Name: More Work Required, Description: leave blank, Destination Step: In Progress, Transition View: No view for this transition
- Transition Name: Resolve Issue, Description: leave blank, Destination Step: Resolved, Transition View: Resolve Issue Screen

We'll also delete a workflow step, so you can see how that is done. Let's delete the 'Reopened' workflow step.

If you want to remove any of the steps from the workflow, you need to make sure it is not used as a transition for any other step. We'll need to remove the transitions to 'Reopened' from the 'Resolved' and 'Closed' issue steps.

Click on 'Delete Transitions' on both the 'Resolved' and 'Closed' steps and delete the 'Reopen Issue' transitions. You will now see a new 'Operation' called 'Delete Step' appear for the 'Reopened' step. Click on 'Delete Step' and confirm on the next screen by selecting 'Delete'.

Your workflow should now look like this:

Step Name (id)	Linked Status	Transitions (id)	Operations
Open (1)	Open	Start Progress (4) >> In Progress Resolve Issue (5) >> Resolved Close Issue (2) >> Closed	Add Transition | Delete Transitions | Edit | View Properties
In Progress (3)	In Progress	No technical Review Required. (831) >> Under Quality Review Implemented (841) >> Under Technical Review	Add Transition | Delete Transitions | Edit | View Properties
Resolved (4)	Resolved	Close Issue (851) >> Closed Reopen Issue (861) >> Open	Add Transition | Delete Transitions | Edit | View Properties
Closed (6)	Closed	Reopen Issue (3) >> Open	Add Transition | Delete Transitions | Edit | View Properties
Under Technical Review (9)	Technical Review	Request Quality Review (871) >> Under Quality Review More Work required. (911) >> In Progress	Add Transition | Delete Transitions | Edit | View Properties
Under Quality Review (10)	Quality Review	Resolve Issue (901) >> Resolved Not Fixed (921) >> In Progress	Add Transition | Delete Transitions | Edit | View Properties

Step 5: Specify Workflow Transition Conditions

Another powerful workflow feature of workflow transitions is their ability to support conditions/validators/Post Functions.

Suppose I want to add a condition that only the current assignee can move issues into 'Technical Review' or 'Quality Review'.

In the 'View Workflow Steps' of our current workflow, in the 'In Progress' step, click on the 'Conduct Technical Review' transition and click the 'Add' link on the 'Conditions' tab. Select the 'Only Assignee Condition' and click 'Add'. Go back to 'View workflow steps' and repeat this same procedure for the 'Proceed to Quality Review' transition on the 'Technical Review' step.

Step 6: Activate The Workflow

Now my workflow is ready to use — I just need to tell JIRA where I want to use it, based on the issue types and projects that I want to use this workflow in.

Workflow schemes define which issue types use what workflow. Let's configure that now.

Click on the '**Workflow Schemes**' link under 'Schemes' in the JIRA Administration area. Click on 'Add workflow scheme' and add a workflow scheme named 'Software Development Workflow'.

We can now associate this workflow scheme with the relevant issue types for our 'Software Development' Project. In this case, I am going to assign this workflow scheme for use with all issues in our project. On the same Workflow Schemes page, on the 'Software Development Workflow', select 'Workflows' from the 'Operations' column. Under the 'Edit Workflows for Software Development Workflow' panel, select 'Assign a workflow'. On the next screen, under 'Add Workflow to Scheme' panel, specify the 'Issue Type' as 'All Unassigned Issue Types' and set the 'Workflow' to 'JIRA Quality Workflow'. Although in this case we are specifying the workflow for all issue types, this is where you could assign specific workflows to specific issue types.

We can now associate this issue workflow with an existing or new project. In this case, we'll create a new project. Firstly follow the steps outlined here to create your new project. Then go to the 'Projects' link under the 'Projects' section of JIRA's administration and click on the name of your new project. On the next screen, click on '(Select)' under Workflow Scheme. Next choose the 'Software Development Workflow' Scheme and finally click the 'Associate' button.

JIRA's flexible workflow engine makes almost anything possible and with great power comes great responsibility.

Creating an Unassigned Issue

You can choose to leave new issues unassigned. This can be achieved by altering the 'Allow Unassigned Issues' flag in the configuration options. To do this go to the General Configuration page of the **Administration** section. Now simply edit the configuration and turn the 'Allow Unassigned Issues' flag on. If a project already exists prior to making this change, please make sure to edit the project and change the "Default Assignee:" to be "Unassigned." Otherwise, the default assignee will continue to be whoever was originally assigned to the project. For more detail please refer to the documentation relating to this function.

This function is not enabled by default, as different companies tend to have different approaches to handling issues. We have found that many of our customers prefer to have issues always assigned to an owner, to ensure that somebody is responsible for its handling and resolution.

Creating Issues via direct HTML links

Please Note: JIRA 4.1+ now uses form tokens as an additional level of security against cross-site request forgery. See Form Token Handling for details on how your external HTML form will handle form tokens.

If you would like your users to create issues from another site, you can by putting links to your JIRA's create issue page. You can also populate the fields on the page with values to select the project, the issue type or even the summary of the issue. This document will detail how to construct theses links and how to populate the fields. This feature is available from JIRA 3.5 onwards.

How to construct the link

The minimal HTML link to create issues has the following structure:

```
<a href="[JIRA BASE URL]/secure/CreateIssueDetails!init.jspa?[ARGUMENTS]">[DESCRIPTION]</a>
```

where

Component	Description	Example
[JIRA BASE URL]	The Base URL of the JIRA you wish to create issues in	http://jira.atlassian.com
[ARGUMENTS]	List of key value pairs separated by '&' which represent the field and its value to be set in the create issue screen	pid=10420&issuetype=4
[DESCRIPTION]	The link description visible to users	create issue in Test Project

JIRA Base URL

This Base URL is the same as the JIRA Base URL you wish to create issues in. This can be found under the admin section -> General Configuration -> Settings. For example, http://jira.atlassian.com is the base URL of the JIRA running at Atlassian.

The Arguments

The list of key value pairs included define which fields will have what values set. The argument list has the following properties:

- Each key value pair is separated by an '&'
  - **For Example:** [keyValuePair]&[keyValuePair]&[keyValuePair]...
- Each key value pair has the form 'key=value' where key is a field name and the value is the desired value to be set for its corresponding field
  - **For Example:** 'pid=10420&issuetype=1&summary=helloWorld&description=greetings'...
- The list must comply with HTML link syntax - that is all characters must be escaped.
  - Characters like space cannot be used directly, they must be encoded (escaped). Hence to use a space, we would replace the space with a '+' or '%20' which is the space equivalent. An excellent *HTML URL-encoding reference* listing all the characters and their corresponding encoded symbol can be found here
  - **For Example:** 'summary=This+is+a+summary%20with%20escaped+spaces'

As you can see, constructing the argument list is relatively simple. All we need is the name of the fields we want to set values for, and just structure it as above.

Fields thats not set will simply be assigned their normal default values. And the issue is not created until the user submits the form (this includes a validation check to confirm the field values are correct).

Finding out the field names and its possible values

The key in the key-value pair is the fields name, and to set a value for that field, we first need to know its name. The name of the field can be found by examining the source code of the page in which the field is in (To view the source code of a page, right click on the browser and select 'View source' or alike). Each field has a *name* attribute which represents the fields name. So all you need to do is find that attribute.

To find the possible values you can set is a bit more tricky. For any fields which accept plain text (such as summary, description and environment) there are no restrictions. However for other fields (such as Project, Issue Type, etc which take in Id) will require you to find the Id values. The range of Id values you can set can be found examining the same source code you found the field name from.

For example, the following is the HTML source code from the create issue page. From this we know that the Components field has the key *'components'* with values *'10013'*, *'10014'* and *'10015'* for each of the 3 components.

```
...
<select multiple name="components" id="components" size=" 3 ">
  <option value="-1">Unknown</option>
  <option value="10013" title="New Component 1  - ">New Component 1</option>
  <option value="10014" title="New Component 2  - ">New Component 2</option>
  <option value="10015" title="New Component 3  - ">New Component 3</option>
</select>
...
```

The following table shows a sample list of the standard JIRA fields with their name (key), the type of value expected and an example of the

value

Display Name	Key	Value Type	Value Examples
Project	pid	Project Id	' 10420'
Issue Type	issuetype	Issue Type Id	standard JIRA issue type values range from '1' to '4'
Summary	summary	Plain Text	'issue+created%20via+link'
Priority	priority	Priority Id	standard JIRA priority values range from '1' to '5'
Due Date	duedate	Date	'15-Dec-2005' - may have different format depending on your JIRA date settings
Components	components	Component Id	'10014'
Affects Version/s	versions	Version Id	'10015'
Fix Version/s:	fixVersions	Version Id	'10015'
Assign To	assignee	Username	'admin' or 'sam@atlassian.com'
Reporter	reporter	Username	'admin' or 'sam@atlassian.com' To have the reporter field default to the currently logged in user, the user must be logged in and must not have the `Modify Reporter` permission.
Environment	environment	Plain Text	'this+is+the+environment'
Description	description	Plain Text	'this+is+the+description'

Custom Fields

Custom Fields key and value can be found by examining the source code also. There name/key are prefixed by 'customfield_' followed by their custom field id. For Example: 'customfield_10000'

Examples

Here are some simple examples to help you on your way. These examples provide links to create issue in JIRA Atlassian Test Project.

Source Code

```
To create an improvement issue in the Test project, click
<a href="http://jira.atlassian.com/secure/CreateIssueDetails!init.jspa?pid=10420&issuetype=4">here</a>
```

```
To create a task with summary 'say hello world', click
<a href="http://jira.atlassian.com/secure/CreateIssueDetails!init.jspa?pid=10420&issuetype=3&summary=s
```

```
To create a task with multiple values selected for a field, click
<a
href="http://jira.atlassian.com/secure/CreateIssueDetails!init.jspa?pid=10420&issuetype=3&summary=say+
```

A more detailed example to

create an issue. Has description, components, due date and a custom field preset.

Current Reporter Browse Project Permission

Some JIRA installations have a use-case where they want a user to only see projects they **can** report issues in. Normally when you add the "Current Reporter" group to the "Browse Project" permission of one project, this project instantly becomes visible to **all** users (via the project table portlet), even if they are unable to report an issue in that project or not.

This guide is for those who want a user to only see issues they've reported **and** also not see any projects that's irrelevant to them (they are unable to create issues for). This permission is available as an optional permission type (since JIRA 3.2). You will need to uncomment the lines below in the file the `WEB-INF/classes/permission-types.xml`. Restart JIRA and this type will be available in your standard permissions page.

If you're running a WAR deployment, you'll need to rebuild the WAR after the change and redeploy. You may need to remove your old exploded WAR directory for the new one to take effect.

```
<type id="reportercreate" enterprise="true">
    <class>com.atlassian.jira.security.type.CurrentReporterHasCreatePermission</class>
</type>
```

When using this special permission, users will only see projects where thy have create permission to and issues within that project where they are the reporter.

Why isn't this included in JIRA by default?

This permission is deliberately commented out of the `permission-types.xml` file. This is to ensure that only advanced JIRA administrators are able to access it. There are two reasons behind this:

- Firstly, the permission itself is used in fairly sophisticated scenarios.
- Secondly, the implementation of this permission is potentially dangerous. For example, it is possible to put your JIRA instance in an infinite loop by mapping this permission to the Create Issue function.

CVS ssh Jira Integration

CVS :ext: ssh Jira Integration

The following information is probably only relevant to Linux/Unix/OSX/Cygwin environments.

CVS_RSH environment variable

In order to use the :ext: method for connection to CVS, the CVS_RSH environment variable needs to be set in the environment that runs JIRA. It should be set to the path to the ssh binary.

put this in your profile

```
CVS_RSH=`which ssh`
export CVS_RSH
```

Problems Authorising when command line works

One user reported the following:

> *The problem was found to be the UsePAM directive in sshd_config on the cvs server(Debian-Sarge) - this needs to be disabled (which it wasn't) with the PasswordAuthentication enabled.*

Disabling Form Token Checking

Please refer to the Form Token Handling documentation on our developer documentation site for more information about how this feature is implemented in JIRA.

Displaying a Field Based on Another Field Selection

Please note that adding Javascript to custom fields is a customisation and not maintained as a supported part of JIRA.

In Atlassian's support JIRA, when a user creates an issue with "Critical" priority, it will display the "Priority Explanation" field.

This can be achieved by performing the following steps:

1. Create a "Free Text Field (unlimited text)" custom field type (Administration -> Issue Fields -> Custom Fields)
2. Fill in the following text into the "Description" field:

```
<script type="text/javascript">
  priority = document.getElementById('priority');
  if (priority) {
      target = document.getElementById('customfield_10000');
      // Hide the target field if priority isn't critical
      if (priority.value != 2) target.style.display='none';

      priority.onchange=function() {
          if (this.value == 2) {
                     target.style.display = '';
                     target.value="enter message here";
                } else {
                    target.style.display='none';
          }
      }
  }
 </script>
```

3. Make sure to change the custom field id and priority id. To find the custom field id, view the source of the page when viewing an issue, or check the URL when editing a custom field.

Editing a custom field option

At the moment it is not possible to rename an option of a custom field, e.g. a Select List custom field. This is fairly easy to do using SQL. Please shutdown JIRA then execute:

```
update customfieldoption set customvalue = 'New Option' where CUSTOMFIELD = <cfid> and id =
<id>;
```

To rename the option. Where `<cfid>` is the id of the custom field and `<id>` is the id of the option you would like to rename.

To get a list of all custom fields do:

```
select * from customfield;
```

Then update all issues with this value:

```
update customfieldvalue set STRINGVALUE = 'New Option' where CUSTOMFIELD = <cfid> and
STRINGVALUE = 'Old Value';
```

Replace `<cfid>` with the custom field's id and 'Old Value' with the text value of the option.

Then restart JIRA and **re-index** the data (Administration -> System -> Indexing).

For details on editing the custom field tables, see the custom field tables documentation.

Escalating issues (or sending email notifications) when the set turnaround time is exceeded

Can JIRA send notifications based on a set issue turnaround time being exceeded? Can it automatically escalate issues that have exceeded a preset turnaround time?

No, not out-of-the box — but this is exactly what services are for. In particular, a Jelly script can be written to find and escalate relevant issues, and the Jelly script can be run every day via a Jelly Service.

Field Layout Schemes in JIRA 3.x

Unknown macro: {version-warn}

JIRA 3.1.1 or earlier

Field Layout Schemes in JIRA 3.x

This document describes creating field layout schemes per issue type per project in JIRA up to (but not including) version 3.2.

Field Layout Schemes

Through the use of Field Layout Schemes, it is possible to configure the visible and required fields per issue type per project.

For example, the issue type **Bug** within project **A** could be associated with one field layout while the issue type **Improvement** also in project **A** could be associated with another field layout. Furthermore, the project default field layout (i.e. all issue types without an associated layout within a project) could be associated with yet another field layout. In this way, each issue type can be associated with a configurable field layout for each project.

System Field Layout Scheme

The **System Field Layout Scheme** governs the field layout for all issue types in all projects not associated with a specific field layout.

This field layout can be edited by navigating to the following **Administration** section:

Administration -> Issue Fields -> Field Layout (System)

The field layout is displayed and can be edited as required.

Creating a Field Layout Scheme

In order to create a specific field layout association for an issue type within a project, it is necessary to create a field layout scheme:

Administration -> Issue Fields -> Field Layout Schemes -> Add Issue Field Scheme

Once created, it is then possible to configure the field layout as required.

Field Layout Association per Issue Type per Project

With a custom field layout, it is possible to associate the field layout with a particular issue type within a particular project.

By navigating to the **Project Administration** section:

Administration -> Project - > <Project Name>

it is then possible to manage the field layout associations for that project by selecting the **Manage** link within the **Field Layout Schemes**

table or within the project summary table. From here, it is possible to create a default field layout association for all unassigned issue types within the project or to create a field layout association with a specific issue type for the selected project.

The issue type specific association will overwrite the project default association.

Fields Allowing Custom HTML or JavaScript

Please note that adding Javascript to custom fields is a customisation and not maintained as a supported part of JIRA.

There are several ways to inject JavaScript or HTML into JIRA:

1. Edit the custom field's description. See Displaying a Field Based on Another Field Selection as an example.
2. Add JavaScript to the header, by modifying a JSP from the file system. See Adding JavaScript to all pages for Google Analytics as an example.
3. You can add HTML or Javascript in the Announcement Banner.
4. Edit the field description in the Field Configuration.

Finding the Id for Issue Types

When configuring a mail handler to create issues from email, it is often useful to know the IDs of issue types.

Here is how you can find the ID of an issue type:

1. In JIRA, click **Administration > Issues > Issue Types**
2. On the 'Global Issue Types' sub-tab, hover your mouse cursor over the **Edit** operation link of an issue type and JIRA will display the issue type's id appended to the URL shown in the browser's status bar. For example, the "id" in this link represents the id of the issue type: `http://<your-jira-server>/secure/admin/EditIssueType!default.jspa?id=1`

Group Name Guidelines for JIRA

We do not have a formal set of naming conventions for groups in JIRA. However we do have some current, or recently resolved, issues related to group names in JIRA, and based on those issues, we suggest at least the following guidelines:

- Don't use commas: JRA-12675
- Don't use ampersands (&): JRA-13780
- Keep group names to less than 60 characters: JRA-13329
- Don't use group names with only one character in JIRA versions prior to 3.12.3: JRA-14495
- Don't use '#' characters in JIRA versions prior to 3.12: JRA-13509
- We suggest standardising on lower case names for groups: JRA-13798, JRA-5434

Beyond those guidelines, our more general recommendation is to keep group names simple, preferably restricting them to alphanumeric characters, and '-', '_' or a space for word separators - e.g. "jira-users".

If you use non-ASCII characters in your group names, ensure that your database character encoding scheme supports those characters. For MySQL, ensure that the database has a character set encoding of UTF8 by following our instructions for creating the database.

If you are integrating JIRA with LDAP, ensure that you conform to any naming restrictions imposed by your LDAP server.

How can I control the editing of issue fields via workflow?

Introduction

Please note that the following instructions **do not** provide a complete solution to Field Level Permissions, but allow to control who can **edit** particular fields. This is achieved with the help of Transition Conditions in a Workflow.

These instructions **do not** provide a solution for restricting who can see the values of fields. Users who have permissions to view an issue, will be able to see the values of these fields for that issue, search by them, receive notifications when these fields change, etc.

Before you read these instructions, it is important to have a good grasp of how Workflows fit into JIRA. A good source of information on Workflows can be found in JIRA's documentation: Configuring Workflow

You should also familiarise yourself with how Screens work in JIRA: Configuring Fields and Screens

Instructions

Please note that the ability to edit some System Fields is already protected by a permission:

System Field	Permission

Fix Version	Resolve Issue
Assignee	Assign Issue
Due Date	Schedule Issue
Reporter	Modify Reporter
Security Level	Set Issue Security

The easiest thing to do for the above fields is to use Permission Schemes to control who can manipulate them. For more information on permissions please see: Managing Project Permissions

However, if the field you are trying to protect is not already protected by a permission, e.g. a custom field, you can use a workflow transition. This transition will allow certain users to only edit certain items of an issue without transitioning to another step of the workflow.

Please follow these instructions:

1. Create two Screens.
2. Using Screen Schemes make sure one of the Screens is mapped to the View Issue and Edit Issue operation. This screen should contain all fields, including the protected fields. Otherwise, no one will be able to see values of fields on the View Issue page.
3. Create another Screen and map it to the Create Issue operation in the Screen Scheme. This screen should not contain the protected fields.
4. Create a workflow transition that goes to the same step as it's original step. Ensure the transition uses the same screen as the Create Issue operation.
5. Create a new group or project role for users who **should not** be able to edit protected fields.
6. Protect the transition using the "User Is In Group" or "User Is In Project Role" conditions.
7. Place users who should not be able to manipulate protected fields into the new group or project role.
8. Edit the Permission Scheme of the project in question and ensure these users do not have the Edit Issue permission. Grant other permission that you deem needed to this group or project role.
9. Ensure that a transition such as this exists for all statuses (steps) in the workflow where the protected fields need to be manipulated. All of these transitions can use the same Screen.
10. Users who are members of the group or project role will be able to execute the transition to edit fields. Other users, who should be able to edit protected fields should use the normal Edit Issue operation.

Please note that the above setup will not allow the protected fields to be populated when issues are created or edited.

Using a Workflow to control edit of an issue by changing Workflow XML

You can use a workflow "transition" to allow certain users to only edit certain fields of an issue without transitioning to another step of the workflow. This page outlines how to achieve this using direct Workflow XML manipulation. If you are not comfortable with directly editing Workflow XML please see How can I control the editing of issue fields via workflow?.

First note that JIRA's workflow editor (as of Jira 3.4.2) uses the term "transition" where as the OSWorkflow documentation refers to the same element as an "action".
Since, this article primarily deals directly with the XML of the workflow instead of the workflow editor, the term "action" will be used.

As mentioned above, this article assumes knowledge of how to write an OSWorfkow in XML.

There are two items that allow us to use the workflow in this way:

- JIRA lets users edit an issue via Workflow actions even if they don't have the "Edit Issue" permission in the permission scheme
- OSWorkflow doesn't force you to transition to a different step, when executing an action

First, you will need to create a screen containing all the fields you want (and only those fields) the user to be able to edit.

Next you need to create the XML document for the workflow. An easy way to get started is to export a workflow from JIRA as XML and then edit that.

In each step that you want a specific user group to be able to edit the issue, create an action with the following attributes:

- The 'view' should be "fieldscreen"
- "jira.fieldscreen.id" should be set to the screen id that contains the fields you want the user to be able to edit

(if you don't know the ID of the screen you want to use, just reassign the screen, after importing the XML, using JIRA's workflow editor).

- A condition of type "class" with the "class.name" as com.opensymphony.workflow.util.OSUserGroupCondition* The "group" as the JIRA user group the current user must be a member of in order to execute this action
- The resulting step set to the same ID as the step that contains the action

The following is an example:

```
<action id="2003" name="Edit Issue" view="fieldscreen">
      <meta name="jira.fieldscreen.id">10010</meta>
      <meta name="jira.description">Edit Issue (for Client)</meta>
      <restrict-to>
        <conditions>
          <condition type="class">
            <arg name="group">ourclients</arg>
            <arg name="class.name">
              com.opensymphony.workflow.util.OSUserGroupCondition
            </arg>
          </condition>
        </conditions>
      </restrict-to>
      <results>
        <unconditional-result old-status="Not Done" status="Done" step="2">
          <post-functions>
            <function type="class">
              <arg
name="class.name">com.atlassian.jira.workflow.function.issue.UpdateIssueStatusFunction</arg>
            </function>
            <function type="class">
              <arg
name="class.name">com.atlassian.jira.workflow.function.misc.CreateCommentFunction</arg>
            </function>
            <function type="class">
              <arg
name="class.name">com.atlassian.jira.workflow.function.issue.GenerateChangeHistoryFunction</arg>
</function>
            <function type="class">
              <arg
name="class.name">com.atlassian.jira.workflow.function.issue.IssueReindexFunction</arg>
            </function>
            <function type="class">
              <arg
name="class.name">com.atlassian.jira.workflow.function.event.FireIssueEventFunction</arg>
              <arg name="eventType">updated</arg>
            </function>
          </post-functions>
        </unconditional-result>
      </results>
    </action>
```

Note that version 2.8 of OSWorkflow allows common actions with a step value of "0" which should result in no change of the step value after executing the action.
However, OSWorkflow 2.8 won't be available in Jira until Jira release 3.7 (see http://jira.atlassian.com/browse/JRA-8902)

Using Templates to control edit of an issue

Overview

You can control who can edit each field by making small changes to the Velocity template files used to display fields in the Edit Issue screen.

One of the points of pain with JIRA is trying to control who can edit particular fields of an issue, as discussed in JRA-1330. Various suggestions have been made there, such as using a workflow, but the page How to create a new Custom Field Type gave me the idea of simply changing the velocity template that is used to display a field to control who can edit the field's values. This approach also provides enough flexibility to make other changes such as who is permitted read the contents of a field.

Steps

1. Decide which field you want to control, e.g. Fix Versions
2. Find the template that is used to generate that field in the Edit Issue screen. The template is probably one of the files `atlassian-jira/WEB-INF/classes/templates/jira/issue/field/*-edit.vm`, e.g. `versions-edit.vm` in this case
   If you have the source code, then you can confirm exactly which template is used by looking in `jira/src/java/com/atlassian/jira/issue/fields` for the field type you are interested in.
3. Note that some templates are used by more than one field, e.g. the `versions-edit.vm` is used for both the Affects Versions and Fix Versions fields.
4. Find the field id of the field you want to control, e.g. for Fix Versions the field id is `fixVersions`. I actually found this out by simply tweaking the template to print out the $field.id, but it's really defined in

`atlassian-jira-enterprise-3.8-source/jira/src/java/com/atlassian/jira/issue/IIssueFieldConstants.j`
.

5. Make the changes and restart JIRA.

Changes

This example shows the changes made to `versions-edit.vm` to control who can edit the Fix Versions field.

```
#controlHeader ($action $field.id $i18n.getText($field.nameKey) $fieldLayoutItem.required
$displayParameters.get('noHeader'))
```

Here is where the changes start:

```
<!-- By default, the fields are writeable -->
#set ($readonly = "no")
#if ($field.id == "fixVersions")
  <!-- This example is restricting who can change the Fix Version to
       members of the fix-version-writers group -->
  #if ($authcontext.user.inGroup('fix-version-writers'))
    #set ($readonly = "no")
  #else
    #set ($readonly = "yes")
  #end
#end
```

The following line is part of the original template

```
#if ($versions && !$versions.empty)
```

but these are the lines that change what is displayed. A "break" command would be useful in Velocity.

```
#if ($readonly == "yes")
    <!-- Display the field value -->
    #if ($currentVersions)
      #foreach ($cv in $currentVersions)
        #foreach ($version in $versions)
   #if ($cv == $version.key)
     $textutils.htmlEncode($version.value)<br>
         #end
        #end
      #end
    #else
      <!-- The Fix Version has not been set -->
      Unknown<br>
    #end
```

All the other lines in this file are unchanged except for the closing `#end` line

```
#else
    <select multiple name="$field.id" size="#minSelectSize ($versions 1 6)" id="$field.id">
        #if (!$fieldLayoutItem.required)
        <option value="$!unknownVersionId"
        #if ($currentVersions && $currentVersions.contains($!unknownVersionId))selected#end
        >$i18n.getText('common.words.unknown')</option>
        #end
        #foreach ($version in $versions)
            #if ($version.key == $unreleasedVersionId || $version.key == $releasedVersionId)
                #if ($optGroupOpen)
        </optgroup>
                #else
                    #set ($optGroupOpen = true)
                #end
        <optgroup label="$textutils.htmlEncode($version.value)">
            #else
            <option value="$!version.key"
                #if ($currentVersions && $version &&
$currentVersions.contains($version.key))selected#end
                >$textutils.htmlEncode($version.value)</option>
            #end
        #end
        #if ($optGroupOpen)
        </optgroup>
        #end
    </select>
  #end  <!-- readonly -->
#else
    $i18n.getText('common.words.unknown')
#end

#controlFooter ($action $fieldLayoutItem.getFieldDescription()
$displayParameters.get('noHeader'))

#if ($archivedVersions && !$archivedVersions.empty)
    #controlHeader ($action "" $i18n.getText($archivedVersionsTitle) false false)

    #foreach ($version in $archivedVersions)
        <a
href="$req.getContextPath()/secure/IssueNavigator.jspa?reset=true&mode=hide&sorter/order=ASC&sorter/
($velocityCount < $archivedVersions.size()),#end</a>
    #end

    #controlFooter ($action "" false)
#end
```

Diff

In case that was a bit too detailed, here is the diff for JIRA 3.8.1:

```
[jira@toolsdev plugins]$ diff -c
/data/jira/jira/atlassian-jira-enterprise-3.8-source/jira/src/etc/java/templates/jira/issue/field/v
~/atlassian-jira-enterprise-3.8-standalone/atlassian-jira/WEB-INF/classes/templates/jira/issue/fiel
/data/jira/jira/atlassian-jira-enterprise-3.8-source/jira/src/etc/java/templates/jira/issue/field/v
2007-03-09 15:49:33.000000000 -0800
---
/data/jira/atlassian-jira-enterprise-3.8-standalone/atlassian-jira/WEB-INF/classes/templates/jira/i
2007-07-04 09:24:52.000000000 -0700
***************
*** 1,6 ****
--- 1,34 ----
  #controlHeader ($action $field.id $i18n.getText($field.nameKey) $fieldLayoutItem.required
$displayParameters.get('noHeader'))

+ <!-- By default, the fields are writeable -->
+ #set ($readonly = "no")
+ #if ($field.id == "fixVersions")
+   <!-- This example is restricting who can change the Fix Version to
+        members of the fix-version-writers group -->
+   #if ($authcontext.user.inGroup('fix-version-writers'))
+     #set ($readonly = "no")
+   #else
+     #set ($readonly = "yes")
+   #end
+ #end
+
  #if ($versions && !$versions.empty)
+   #if ($readonly == "yes")
+     <!-- Display the field value -->
+     #if ($currentVersions)
+       #foreach ($cv in $currentVersions)
+         #foreach ($version in $versions)
+         #if ($cv == $version.key)
+           $textutils.htmlEncode($version.value)<br>
+           #end
+         #end
+       #end
+     #else
+       <!-- The Fix Version has not been set -->
+       Unknown<br>
+     #end
+   #else
+     <!-- All the other logic in this file is unchanged -->
      <select multiple name="$field.id" size="#minSelectSize ($versions 1 6)" id="$field.id">
          #if (!$fieldLayoutItem.required)
          <option value="$!unknownVersionId"
***************
*** 25,30 ****
--- 53,59 ----
          </optgroup>
          #end
      </select>
+   #end  <!-- readonly -->
  #else
      $i18n.getText('common.words.unknown')
  #end
```

Pros

- Simple changes to one .vm file per field to be controlled, no recompilation of source code necessary
- Uses the existing Jira group mechanism

Cons

- Need to manually apply changes to updated versions of JIRA. Happily, the changes are cleanly localised.
- Only controls fields edited using the browser, not with the SOAP API
- Need some familiarity with the Velocity template language

Troubleshooting

If you are having trouble with a hidden value being reset when the issue is edited, you can try passing it back like this:

```
<input type=hidden name="$customField.id" id="$customField.id" value="$value">
```

This may occur when a user with no write-permission for a select field edits the issue.

How do I assign issues to multiple users

JIRA is designed so that issues must be assigned to a single individual to prevent tasks from being overlooked. A team lead or manager should assign issues out to individuals, or your users will pick from a list of issues that they have the option to take on.

However, if you want to configure JIRA to allow issues to be assigned to multiple users there are a few option for doing so:

- Managing Issues via a Queue
- Managing Issues via Group Ownership
- Managing Issues via a User Account
- Managing Issue via Sub-Tasks

It is easy to still setup a queue the a group can pick from, or affiliate an issue with group in addition to having it assigned to an individual within that group:

Managing Issues via a Queue

You can configure your JIRA project to assign issues to an 'Unassigned' "queue" by default, which your users can then pick issues from.

To do this, set up the following:

1. Configure your JIRA project to allow the 'default assignee' to be 'Unassigned' (see Defining a Project).
2. Ensure that 'Allow unassigned issues' is set to ON in your General Configuration settings (**Administration > Global Settings > General Configuration**).
3. Set any issues that you want to be in the queue to be 'Unassigned'.
4. Create a dashboard page with a filter that lists all 'Unassigned' issues, share the dashboard page and request that interested members of the group display the shared page on their dashboards. See Managing Multiple Dashboard Pages for instructions.

Managing Issues via Group Ownership

You can add a custom field to store which users and groups should be associated with a given issue. This is particularly useful for projects where a team owns all issues of a particular type.

To do this, set up the following:

1. Add a group picker custom field to your issues.
2. Configure an email notification in your project's notification scheme to be sent to the 'Group Custom Field Value'.

An issue can now be "assigned" to the group by selecting the appropriate group in the group picker. An email notification will be sent to the group.

Another option is to add a user picker custom field rather than a group picker, and assign multiple users to an issue. However, you will then have both the JIRA default user field and custom user field for your assignees.

Managing Issues via a User Account

You can create a JIRA user account to represent a group of people (e.g. 'developers') and assign issues to this user.

To do this, set up the following:

1. Create a JIRA user to represent the group (see Managing Users).
2. *(Optional)* Create an email mailing list for this group (not a JIRA function) and set the mailing list email as the JIRA user's email address.
3. Create a dashboard page showing issues assigned to this user, share the dashboard page and request that interested members of the group display the shared page on their dashboards. See Managing Multiple Dashboard Pages for instructions.

An issue can now be assigned the new "user" representing the group and your users can track the issues on their dashboards. If you have set up a mailing list, your users will also be notified by email.

Managing Issue via Sub-Tasks

If you have a task managed by different users then you are able to break the combined task into individual subtasks with their own single assignees.

How do I delete a user account?

Someone has left the company. How do I delete their user account if they have reported issues?

We recommend that you **deactivate** rather than delete a user's account. Deactivating a user's account will prevent that account from being used and being able to login, but will preserve their issues history.

- If you would like to **deactivate** a user's account, please read Deactivating a User's Account.
- If you would like to **delete** a user's account, please read Deleting a User's Account.

How do I disable Firebug for JIRA?

The Firebug add-on for Firefox can significantly degrade the performance of web pages. If JIRA is running too slowly (the JIRA dashboard, in particular), we recommend that you disable Firebug.

To disable Firebug for JIRA:

1. Open the '**Firebug**' pane in the Firefox tab that has JIRA running, by clicking the Firebug icon.
2. Click the down arrow next to the '**Net**' tab, and select '**Disable monitor**' for the URL of your JIRA instance (e.g. jira.atlassian.com)
3. Repeat Step 2 on the '**Console**' and '**Script**' tabs.

How Do I Use an SSL Certificate Generated Using openssl?

You have an SSL Certificate that was generated using openssl, and you would like to use it with JIRA.

You need to have both the signed ssl certificate and the private key that was generated using openssl. Then you convert the certificate + key pair to pkcs12 format using openssl:

```
[amyers@erdinger:ssl]$ openssl pkcs12 -export -in server.cert  -inkey server.key -out
server.p12
```

When doing this, openssl should ask for a password, so remember whatever you enter here. This will convert the certificate (server.cert) and the private key (server.key) into a pkcs12 file (server.p12).

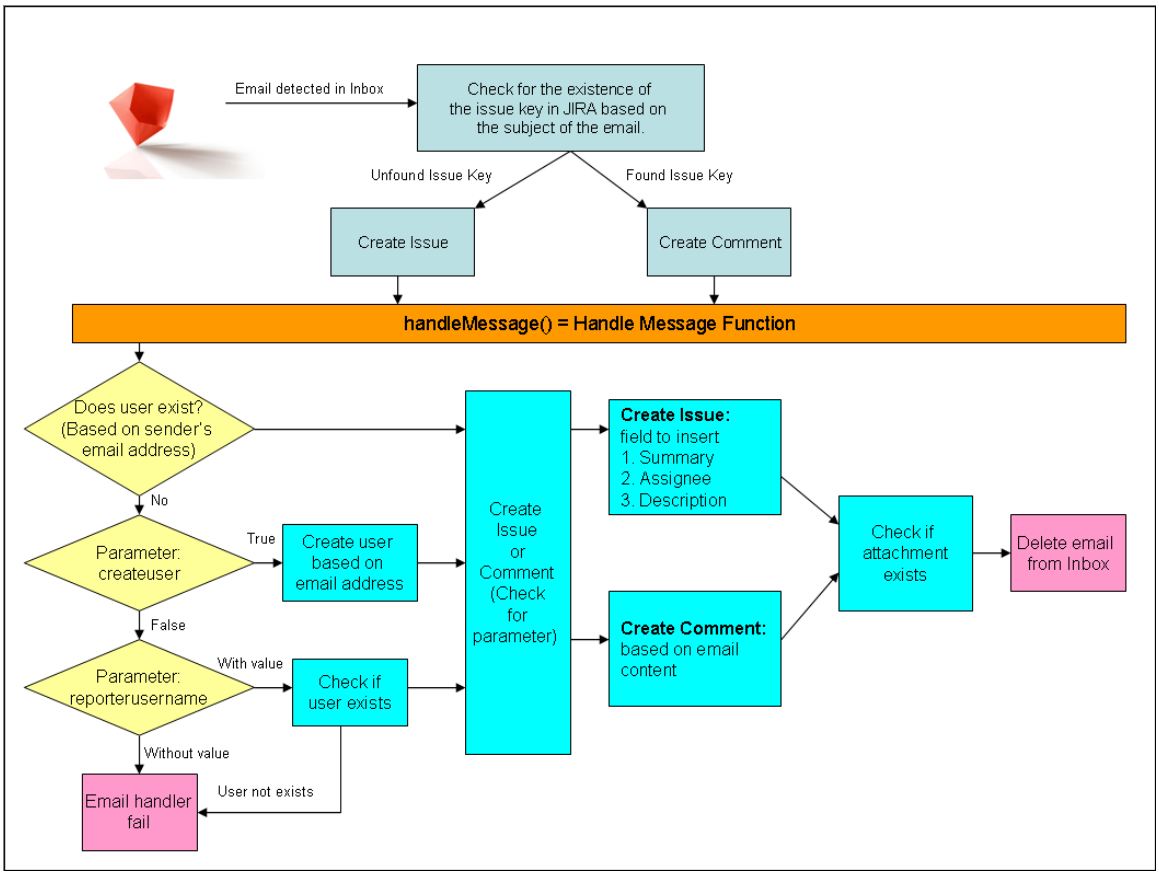
Next you simply need to configure tomcat to use the pkcs12 (.p12) file as its keystore by editing `$JIRA_HOME/conf/server.xml`:

```
<Connector port="8443" maxHttpHeaderSize="8192"
                   maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
                   enableLookups="false" disableUploadTimeout="true"
                   acceptCount="100" scheme="https" secure="true"
                   clientAuth="false" sslProtocol="TLS" SSLEnabled="true"
                   URIEncoding="UTF-8" keystorePass="changeit"
                   keystoreType="pkcs12" keystoreFile="/path/to/server.p12" />
```

The important thing to specify is that `keystoreType="pkcs12"`. The keystorePass is whatever password you gave when generating pkcs12 file, and the keystoreFile is the path to the file.

The process should be the same for Confluence (or indeed any other application running on Tomcat).

How the CreateOrCommentHandler works?



How to change Multi Select Custom field size using script

By default, Multi Select custom field will only show maximum o 5 rows (including none) of options, this is not convenient for some users who have more than 20 options in their multi select custom field. Below is the workaround to change the size of the field using simple script.

Thanks Kanishk Choraria for this workaround.

The Steps

1. In Custom fields screen, click on Edit of the custom field.
2. Refer to *?id=* for the custom field ID, for example _+http://localhost:8080/secure/admin/EditCustomField!default.jspa?id=10200 (http://localhost:8080/secure/admin/EditCustomField\!default.jspa?id=10200)+_
3. Add the ID to script below:

```
<script type="text/javascript">
mselectbox = document.getElementsByName('customfield_10200');
mselectbox[0].setAttribute("size", "20");
</script>
```

4. In example above, the size of multi select field will be expand to 20 rows
5. Put the script above to Description of custom field and update.

How to change the location of stdout and stderr logs

If JIRA is running as service in windows, stdout_*.log and stderr_*.log will be created, and over time, these files are rapidly growing and become large.

To change the stdout_.log and stderr_.log location such as D drive.

1. Uninstall the jira service.
2. Open `service.bat` with editor and find the parameter below:

```
set PR_LOGPATH=%CATALINA_BASE%\logs
set PR_STDOUTPUT=auto
set PR_STDERROR=auto
```

3. change `set PR_LOGPATH=%CATALINA_BASE%\logs` to any location you wish such as `set PR_LOGPATH=D:\logs`

4. Run `service.bat install JIRA_SERVICE_NAME` to install JIRA as service again.

How to clear the resolution field when the issue is reopened

In the default JIRA workflow, issues have their resolutions cleared upon re-opening an issue. This is important because many reports/filters could be inspecting for the presence of a *Resolution* to be considered resolved.

The *Resolution* field is typically cleared by setting a post-function in the workflow transition you'd like to have this occur. The function to use is Update Issue. The field to use is Resolution. The value to choose is None.

How to configure sub-task to have a specific screen?

By configuring a custom Issue Type Screen Scheme, it could able to have a specific screen for sub-task issue type. For example:

1. Create a screen via *Administration -> Issue Fields -> Screens* (e.g. Sub-task screen)
2. Create a Screen Scheme via *Administration -> Issue Fields -> Screen Schemes* (e.g. Sub-task Screen Scheme)
3. Configure this newly created screen scheme to have a 'sub-task screen' when creating issue
4. Create a Issue Type Screen Scheme via *Administration -> Issue Fields -> Issue Type Screen Schemes*. Configure this newly created Issue Type Screen Scheme to have a 'Sub-task Screen Scheme' for 'sub-task' issue type
6. Associate this Issue Type Screen Scheme with the project

For more information on Issue Type Screen Scheme, please refer to this documentation:

- http://www.atlassian.com/software/jira/docs/latest/issuetype_screenschemes.html

Searching JIRA Knowledge Base

How to convert types using Jelly

Scenario

When programming in Jelly, you can get in a situation as illustrated by the following code snippet:

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib" xmlns:j="jelly:core">

<jira:CreateProjectRole name="QARole" description="QA role">
${jelly.role.id} ${jelly.role.name} ${jelly.role.description}
</jira:CreateProjectRole>

<j:set var="qaroleid" value="${jelly.role.id}"/>

<jira:AddPermission schemeId="0" permissions="Edit" type="projectrole" projectroleid=
"${qaroleid}"/>

</JiraJelly>
```

The goal here is to create a new Project Role and then set the appropriate Permissions to it. However, as the `projectroleid` and `qaroleid` variables are have not the same type, you should get an error like this:

```
Could not run script.
Extra Information: [hide]
Error: <JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib"
xmlns:j="jelly:core">10010 QARole QA role
Exception: org.apache.commons.jelly.JellyTagException: null:10:0: Cannot assign value of type
'java.lang.Long' to property 'projectroleid' of type 'java.lang.String'
java.io.PrintWriter@334cee
```

So `qaroleid`, which received a `java.lang.Long` value from the `jelly.role.id` context variable, should be converted to a `java.lang.String` type so it can be informed as `projectroleid` attribute when setting a Permission.

Problem

How can you convert types in Jelly?

Solution

You can use the `invoke` Jelly tag to call the method `toString` on the `jelly.role.id` context variable and store this value the in the `qaroleid` variable. So you should replace this line...

```
<j:set var="qaroleid" value="${jelly.role.id}"/>
```

... with this...

```
<j:invoke on="${jelly.role.id}" method="toString" var="qaroleid"/>
```

... and the script will work sucessfully.

You can also (just as an example matter) use the `new` Jelly tag to convert the `java.lang.String` value of the `qaroleid` variable into a `java.lang.Float` value and store it in the `qafloatvar` variable, as shown here:

```
<j:invoke on="${jelly.role.id}" method="toString" var="qaroleid"/>

<j:new var="qafloatvar" className="java.lang.Float">
<j:arg type="java.lang.String" value="${qaroleid}" />
</j:new>
```

You may find useful to look at these pages for more information on Jelly tags:

- Jelly - Tag Reference
- Jelly Scripting Hints

How to create a download link to a file

Symptoms

You can perform the following steps to create a download link to a file in JIRA:

1. Use 'Text Field' custom field type to represent the link to a file (Administration -> Issue Fields -> Custom Fields)
2. Enable the 'Wiki Style Renderer' for this 'Text Field' custom field (Administration -> Issue Fields -> Field Configurations)
3. In the Issue Screen, try to add a link in the 'Text Field' custom field. For example:

```
[file:///V:/Jira test/example.xls]
```

The file can be opened by right clicking the link and select on the 'Save Target As...'. Do note that this only works on Internet Explorer but can also be enabled in FireFox. Please see:

- Linking to local file under Firefox

For more information on renderers, please refer to:

- Configuring Renderers
- http://jira.atlassian.com/secure/WikiRendererHelpAction.jspa?section=links

How to disable the Resolve issue screen while resolving issues

If you don't wan JIRA to to show a screen when resolving an issue, then disassociate the 'Issue Resolved Screen' from the transition.

The transition view should not be associated with any screen.

How to display a different format for the Number customfield

If you do not like how the Number custom field is displaying in Jira (for example, if you do not want it to display the commas), you can modify a velocity file to configure this. In particular, look at the file `WEB-INF/classes/templates/plugins/fields/view/view-number.vm`. As mentioned here:

> *At the moment to change the way the numbers are printed the easiest thing to do is to edit: WEB-INF/classes/templates/plugins/fields/view/view-number.vm*

> *under the JIRA web application and replace:*
> *$!numberTool.format($value)*
> *with:*
> *$value.longValue()*
>
> *and restart.*

How to ensure the Road Map tab is visible

If the Road Map project tab is not visible for a particular project, it may be for one of the following reasons:

- The Road Map Panel (roadmap-panel) module is disabled
- The "Fix Version/s" field is hidden via at the Field Configuration

So, if you are not seeing this tab, but you want to, ensure:

- The Road Map module is enabled under *Administration -> System -> Plugins -> Project Panels Plugin*
- The "Fix Version/s" field is not hidden under *Administration -> Issue Fields -> Field Configurations*

How to Get Unicode 'non-ASCII' Characters in HTTPS URL to Appear Correctly

In most cases URLs running over HTTP work fine, but not when using HTTPS (i.e. over SSL). This usually results in Unicode (non-ASCII) characters in an HTTPS URL appear incorrect in the URL, and the served page contains numerous errors

This occurs when the `useBodyEncodingForURI="true"` flag is not defined in the HTTPS connector definition in `conf/server.xml` of the Apache Tomcat application server running JIRA. This flag is set as such by default in 'recommended' distribution installations of JIRA.

However, in JIRA WAR setups, this might not be the case. Hence, ensure that the `useBodyEncodingForURI="true"` flag is included in the following element of the `conf/server.xml` file of your Apache Tomcat installation running JIRA:

```
<Connector port="8443" maxHttpHeaderSize="8192"
               maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" disableUploadTimeout="true"
               acceptCount="100" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS" useBodyEncodingForURI="true" />
```

After specifying the `useBodyEncodingForURI="true"` in all connector definitions (i.e. both the HTTP and the HTTPS connectors), as described in the 'Modifying Tomcat server.xml' section of the Installing JIRA on Tomcat 6.0 and 5.5 documentation.

How to have long component version names display properly in the Issue Navigator

If you create a component or version name which is very long, it is not possibly to view it in the issue navigator. If you're finding this happening in your Jira instance, you can set the width of the component or version list to auto and wrap it in a element with fixed width in this file:

- WEB-INF/classes/templates/jira/issue/searchers/edit/project-constants-searcher-edit.vm

Here is a code sample:

```
...
<div style="width: 180px; overflow-x: scroll; border: 1px #ddddff solid;">
<select .. style="width: auto;">
...
</select>
</div>
...
```

How to Remove 'NONE' from a Select List Custom Field

There are some instances where the 'NONE' in every select list needs to be removed. By default in JIRA you cannot do this from the web interface. See JRA-7687 for a discussion on this feature request.

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

This workaround will apply to all Select List Custom fields in the instance.

To remove the field, edit `<atlassian-jira/WEB-INF/classes/templates/plugins/fields/edit/edit-select.vm`. Delete the lines:

```
...
  #if (!$fieldLayoutItem || $fieldLayoutItem.required == false
    <option value="-1">$i18n.getText("common.words.none")</option>
  #else
    <option value="">$i18n.getText("common.words.none")</option>
  #end
...
```

The following code should remain:

```
...
<select name="$customField.id" id="$customField.id">  #foreach ($option in $configs.options)
    <option value="$textutils.htmlEncode($option.value)"
  #if ($value && $value == $option.value)selected#end
    >$option.value</option>
  #end
</select>
...
```

Make sure to back up the velocity file before changing it. Keep in mind the notes from Modifying JIRA Templates and JSPs.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

How to Rename the 'Priority' Field in the Issue Navigator

In order to rename the "Priorities" text in the issue navigator, you may need to edit the following properties file in the language pack:

> *com/atlassian/jira/web/action/issue/IssueNavigator_en.properties*

The following property value in the Issuenavigator_en.properties:

> *navigator.filter.matchingpriorities*

The language pack is a JAR file located in `<jira-install>/atlassian-jira/WEB-INF/lib`. For more information about Customizing Text in JIRA, please refer to Customizing Text.

How to re-order statuses

There is currently no way to change Statuses order in JIRA, the only workaround is to manipulate JIRA database, please follow JRA-5198 and vote on this feature.

This FAQ is to document down the workaround for JRA-5198, all credits goes to the contributors in JRA-5198

Direct Database Manipulation is outside the scope of Atlassian support - this document is for informational purposes only

Warning

Please Backup Database before implement the below workaround

The Steps

1. Shutdown JIRA
2. Backup Database
3. List down the current Statuses order using query below:

```
SELECT pname,SEQUENCE FROM issuestatus order by SEQUENCE;
```

4. You will get result like below:

pname	SEQUENCE
Open	1
In Progress	2
Reopened	3
Resolved	4
Closed	5

5. Update the current status sequence numbers to 10, 20, 30 instead of 1,2,3 so it's easier to modify later.

```
update issuestatus set SEQUENCE =10 where SEQUENCE=1;
```

6. Repeat step above for other statuses, you will get result like below:

pname	SEQUENCE
Open	10
In Progress	20
Reopened	30
Resolved	40
Closed	50

7. Finally, Update the table with the sequence you want, for example change Status "Reopened" show after status "Resolved":

```
update issuestatus set SEQUENCE =45 where pname="Reopened";
```

8. Restart JIRA to take effect.

How to re-order the list of issue operation in an issue

To re-order the issue operation list, the 'order' value at the following file needs to be edited:

```
/atlassian-jira/WEB-INF/classes/system-issueoperations-plugin.xml
```

For example, change the 'order' value for 'Comment on this issue' operation from 50 to 10. By doing so, it will place this operation at the top of the issue operation list:

```
<issue-operation key="comment-issue"
i18n-name-key="admin.issue.operations.plugin.comment.issue.name" name="Comment on this issue"
class="com.atlassian.jira.issue.operations.CommentIssueOperation" state='enabled'>
        <resource type="velocity" name="view"
location="templates/plugins/operations/commentissue.vm" />
        <order>10</order>
</issue-operation>
```

Please restart JIRA after the modification. Do note that this modification will be applied to all the projects in JIRA.

How to resize Free Text Field customfield

You can customise the size of customfield (Free Text Field) at
`%JIRA_HOME%/atlassian-jira/WEB-INF/classes/templates/plugins/fields/edit/edit-textarea.vm`

How to resize the 'Components' and 'Affects Versions' fields in the Issue Navigator

When the values of components or versions are too long, the full name will be truncated within Issue Navigator. Hence, the user is not able to view the full name of the component/version.

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

The workaround is navigate to and edit the following file:

`<JIRA_HOME>/atlassian-jira/WEB-INF/classes/templates/jira/issue/searchers/edit/project-constants-searche`

Change width: 180px to be something that is more appropriate for your JIRA instance.

```
{panel}
{color:}...{color}
{color:}<div style="width: 180px; overflow-x: scroll; border: 1px #ddddff solid;">{color}
<select ..  {color:blue}style="width: auto;"{color}>
...
</select>
{color:blue}</div>{color}
...
{panel}
```

A restart of JIRA is required for the change to take effect. Please make sure to test this in a development instance prior to implementing in your production instance.

Make sure to back up the velocity file before changing it. Keep in mind the notes from [JIRA:Modifying JIRA Templates and JSPs].

RELATED PAGES

No content found for label(s) jira-custom-velocity.

How to Restrict the Subversion Commits Tab to Selected Projects or Users

Symptoms

It may be the case that a project or subset of users in JIRA may need the subversion commits tab and another project or subset of users does not. It is not immediately obvious how to configure such access to the tab.

Cause

The tab is displayed based on the permissions associated with the user viewing issue, project, et cetera.

Resolution

To restrict the tab to selected projects or users, simply configure the appropriate project/group of users to either have or lack the View Version Control permission.

How to search by number range in the Issue Navigator

When searching for some issues based on the "Number Field" custom field type, there may be a requirement to search issues for a range of numbers based on the custom field.

The "Number Field" custom field type can be configured to use the "Number range searcher". Hence, the issues can be searched within a range of values for that customfield. The search template can be changed by clicking on the "Edit" operation via Administration -> Issue Fields -> Custom Fields. For more information, please refer to the custom field documentation

Please note that a re-index needs to be performed via Administration -> System -> Indexing after changing the custom field searcher.

How to show a transition only when the Assignee is different from the Current User

This solution is particularly useful when your workflow has a restriction on assigning issues. For example, certain users can only assign an issue to themselves by executing a transition, and you want the "Assign to Me" action to be visible only if the Current User is not yet the Assignee of the issue.

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian **can not guarantee providing any support for it**. Please be aware that this material is provided for your information only and using it is done so at your own risk.

Let us consider the assigning scenario. To show the "Assign to Me" transition only when the Assignee is different from the Current User, execute the following steps:

1. Download and install the Jira Scripting Suite plugin. See the Installation&Upgrade Guide.
2. Create a draft of your workflow so you can change it. See Configuring Workflow - Editing an active workflow.
3. Go to the "Assign to Me" transition and add a "Jython Condition". In the "Add Parameters To Condition" screen, paste the following lines of code and click "Add".

```
import com.atlassian.jira.ComponentManager
curr = ComponentManager.getInstance().getJiraAuthenticationContext().getUser()
assig = issue.getAssignee()
result = (curr != assig)
```

4. Publish the changes to your workflow.

This was tested in JIRA 4.0.2.

See also How to Allow Users to Assign Issues Only in a Specific Transition.

Importing data

To import issue data from CSV (Comma-Separated Value), Bugzilla, FogBugz or Mantis, please see the documentation:

- CSV
- Bugzilla
- FogBugz
- Mantis

For other types of import, please contact us as we may have done it before. See also JIRA's Jelly support — Jelly is a scriptable interface to JIRA that is useful for importing data.

Importing user from LDAP

JIRA's LDAP integration currently requires users to have accounts both in LDAP and in JIRA. For instance, if a user is added to LDAP, they will have no access to JIRA until someone creates them a JIRA username (and assigns it to groups).

The attached tool searches LDAP for user accounts, and generates a JIRA Jelly script which will create a JIRA user account for each LDAP account. Typically one would use this tool when first installing JIRA, to bulk-create JIRA users matching each LDAP account.

How to use

Download the current binary distribution. Alternatively, if you are Java-literate and keen, all current distributions contains source distributions. You can also get the source from Subversion at http://svn.atlassian.com/svn/public/contrib/jira/jira-ldap-userimporter/trunk.

Create a file, `ldap.properties`, to specify your LDAP server's details. If you are unsure of these, first test with an LDAP browser (there are many LDAP browsers available on the internet, you can try using this LDAP browser or search for an alternative one). Here is a `ldap.properties` configured for use against a local OpenLDAP directory:

```
# Configuration file for JIRA's LDAP user importer

# URL of your LDAP server, Eg:
java.naming.provider.url=ldap://192.168.0.74

# Username and password of account that has privileges to loop through all users, eg:
java.naming.security.principal=cn=admin,dc=atlassian,dc=com
java.naming.security.credentials=secret

# LDAP node below which we should search, eg:
searchbase=ou=People,dc=atlassian,dc=com

# LDAP query run below 'searchbase' identifying user nodes, eg:
query=(objectclass=*)

# Name of record in nodes which should become the username in JIRA, eg:
username_attr=uid

# Record that contains the user's full name. When commented out, defaults to username_attr
value. Eg:
fullname_attr=cn

# Record that specifies the user's email address. When commented out, username_attr value with
email_suffix appended will be used
#email_attr=
email_suffix=@atlassian.com


# Generally you don't want to touch this
java.naming.factory.initial=com.sun.jndi.ldap.LdapCtxFactory
```

Once you have created `ldap.properties`, run `java -jar jira-ldap-userimporter-1.1.jar`. If you have the ldap.properties details correct, this command will result in XML text being printed to the console. Eg:

```
~/jira-ldap-userimporter-1.1$ java -jar jira-ldap-userimporter-1.1.jar
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:CreateUser username="nobody" password="nobody" confirm="nobody" fullname="nobody"
email="nobody@atlassian.com"/>
  <jira:CreateUser username="jturner" password="jturner" confirm="jturner" fullname="Jeff
Turner" email="jturner@atlassian.com"/>
  <jira:CreateUser username="anonymous" password="anonymous" confirm="anonymous"
fullname="anonymous" email="anonymous@atlassian.com"/>
  <jira:CreateUser username="devuser" password="devuser" confirm="devuser" fullname="devuser"
email="devuser@atlassian.com"/>
</JiraJelly>
```

This text can now be redirected to a file, and fed to the Jelly Runner (see the Jelly docs).
However, first make sure that **LDAP password checking is disabled** (ie. there is no LDAPCredentialsProvider section in osuser.xml), otherwise the Jelly script will fail, claiming these users already exist.

Additional Options

Check Bob Swift's JIRA Command Line Interface for another great way to import users. See the addUser commands on the page; it includes importing from a file.

Feedback? Problem

Please raise a Support Request.

Exceeding your user limit on import?

If you are getting a `LimitExceededException`, you may find these instructions from one of our customers helpful.

Thanks to Ricardo Sueiras

Version History

Version	Comments
1.1	Username is converted to lowercase automatically
1.0	First Released

Neat JIRA LDAP tricks

Gianugo has an interesting blog up about how to take the pain out of migrating users from LDAP

JIRA as a Support System

This document shows how to set JIRA up as a support system:

- JIRA Advantages
- How Atlassian Uses JIRA for Support
- Feature Setup Instructions
  - Permissions
    - 1. Different Projects
    - 2. Issue Level Security
  - Work Queues
  - Email Integration
  - Support Knowledge Base
  - Log Phone Calls To Tickets
  - SLAs
  - Escalate Tickets
  - Change Status After Comment
- Example Scenario
  - Further Support Discussion
  - Related Best-Practice Discussions

JIRA Advantages

- By using one system for support and bug / feature tracking, you can link support issues to the bugs that they reference.
- JIRA is a very simple system to install and use - there is very little training required for support staff, or end users
- JIRA's configurable workflow adapts to your existing support processes

How Atlassian Uses JIRA for Support

Check out the specific ways Atlassian has configured support.atlassian.com at How Atlassian Uses JIRA for Support.

Feature Setup Instructions

Note that some terminology is different between the two systems - for example a support system typically uses the word 'ticket' where an issue tracking system may use the word 'issue'.

Permissions

A support system has different needs for permissions than a bug tracking system. Typically as an end user you can only see issues that you, or your company has raised. The ways of doing this are:

1. Different Projects

At a very simple level, if you are supporting a very limited number of clients, you can set up a different project for each of your clients, with a different permission scheme for each project.

You can set up the permissions so that only the reporter of an issue, and the support staff, can see the issue (i.e. give the 'Browse Projects' permission to the Reporter and appropriate internal groups). This means that each user can only see their own issues, and is very suited to an internal help desk system, or any other support system with a large number of end users.

2. Issue Level Security

You can set up different security levels for each customer. This is similar to having different projects, but allows the support team to manage the issues in just one project.

Work Queues

Often in support systems, the priority of an issue is not as important as the order in which the issues are raised. There may be a Service Level Agreement in place, which specifies that an issue must be responded to within a certain time.

The JIRA toolkit will show you whether the last commented was from a JIRA Administrator, or whether it was from a customer. This allows issues to be prioritised by the order in which they need a response.

Read about How Atlasian Visualizes our Support Queue.

Email Integration

JIRA can easily be set up to handle incoming email, and create new issues, or comment on existing issues. It also sends mail notifications to users when the issue has been updated.

When setup this way, the client can create and comment on an issue, without having access to JIRA.

For more information, see the documentation on Setting up email integration in JIRA — particularly the CreateOrCommentHandler.

Support Knowledge Base

Please see Adding Knowledge Base Functionality To JIRA.

Log Phone Calls To Tickets

Please see Logging Phone Calls In JIRA.

SLAs

Most SLAs are very specific to a particular organisation, so it is difficult to ship a completely out-of-the-box solution with JIRA that will meet everyone's needs. However JIRA has 2 approaches that can be used separately or jointly to meet SLAs:

1. The most powerful approach is to write a Jelly script (sample available) which invokes a saved search (filter), and loops over the issues, adding a comment, transitioning them to a new state (e.g. "Requires Response"), or otherwise letting people know that action needs to be taken. This Jelly script would then be run periodically by a Jelly runner service. Atlassian uses this approach on https://support.atlassian.com, to automatically close issues that have not been replied to in X days. We have a filter returning issues in status "Waiting for Customer", updated from any time to 1 week ago (i.e. not touched in the last week), and these are transitioned to "Inactive", which triggers an email letting the customer know.
2. Create a search filter that finds all issues that meet a certain criteria. Save this filter and subscribe to it, either by email (through

JIRA) or by subscribing to the filter's RSS feed in an RSS reader. This way JIRA will notify subscribers what issues are 'outstanding'. For more information on creating and saving filters and subscriptions please see this page. There is also a short video on Simple SLA with Filters.

3. If a Jelly script cannot do what you want, or JIRA's searching capabilities are not sufficient to match issues you want, you could write a custom service that locates issues that meet a certain criteria and then does something with matching issues. For example, a service could reassigns the issues to another team member (e.g. project's lead), increments priority, sends notifications, etc. For more information on JIRA services please see this page.

There are also a number of plugins such as Issue Time Tracking Report and Vertygo that provide support for SLAs.

Escalate Tickets

For an example of code that uses JIRA's API to escalate issues please see: Simple Escalation.

Change Status After Comment

A user adding a comment via the JIRA UI can be prompted to change the issue status. The source is not yet available as this is currently a work in progress but please visit Adaptavist for updates.

Example Scenario

Here is an example scenario for a support environment within an organisation and suggestions on how to setup JIRA to fit this environment.

1. End-users: company workers place phone calls to the 'hot-line' team.
2. Hot-line: answer the end-users and open a ticket for every issue
3. 1st level Help Desk: analyse hot-line tickets, and close them if they are able to respond themselves. Otherwise they dispatch the ticket to one of three 2nd level help desk teams.
    a. Technical 2nd level help desk
    b. Functional 2nd level help desk
    c. Logistic 2nd level help desk

The best way to setup JIRA for the above environment is to create a separate JIRA project for each of the four support groups (one 1st level support team and three 2nd level support teams). It would also be useful to create a separate permission scheme for each project so that permissions can be managed for each project separately. For more information on permissions please see:
Managing Project Permissions

The hot-line team will create a new issue in the 1st level support team's dedicated project (referred to as 'hot-line' project from here on) for every call they receive. The way the hot-line project should be setup depends on whether the actual end users need to see JIRA issues. If yes, ensure that every member of this hot-line team has Modify Reporter permission so that they can set the 'reporter' of the issue as the actual end caller.

It is also possible to create a custom field of type User which can be used to track who (which member of the hot-line team) actually created the issue. The hot-line team member will have to populate this field with their username. For more information on custom fields please see:
Adding a Custom Field

You can then give the Browse Project permission in the hot-line project's permission scheme to the 'Reporter' role (please see the permission documentation referenced above for more details) and 2 user group . One user group will represent represents the hot-line team and the other the 1st level support team. This way, the end users can see issue created on their behalf, but not issue's created for other users. The hot-line group members and the 1st level support team will be able to see all issues in the project.

If the actual end users do not need to see the issues in JIRA it is probably better to not give the Modify Reporter permission to anyone for the hot-line project. The reporter field of the issue will then automatically default to the logged in user (i.e. the hot-line group member who is creating the issue). A custom field of type User can still be created and used to record on whose behalf the issue was created. The field will have to be populated manually during issue creation. This custom field can also be made 'required' so that it will have to be populated during issue creation.

The user group representing 1st level support team should be given the resolve and close issue permissions so that they can resolve/close issues once they are dealt with.

I also recommend setting the 'Assignable User' permission in the hot-line permission scheme to the user group representing the 1st level support team, so that issues can be assigned to them. The 'Assign Issue' permission can be given to the hot-line group so that its members can assign issues to specific 1st level support team members.

Alternatively, the 'Assign Issue' permission can be given to only the 'Project Lead'. The default assignee of the hot-line project can be set to 'Project Lead' or 'Unassigned' (if unassigned issues are enabled. Then the hot-line project's lead can go through all the issues assigned to him/herself or all Unassigned issues and assign them appropriately.

If the 1st level support team members cannot resolve an issue they should create a new issue in one of the other three projects (the technical support project, the functional support project, logistics support group project) to
indicate that the issue has been passed to the 2nd level support. For this to occur the 1st level support team must be given the 'Create Issue' permission in the permission schemes used by these projects.

The issues created in the 2nd level support projects should be linked to the issue in the hot-line project
using Issue Links:

- Configuring Issue Linking

Each of the 3 support projects can be setup as required by each team, in terms of their permissions, notifications, workflows, etc.

If all internal users are stored in a LDAP directory, please take note of JIRA's LDAP integration:

- Connecting to an LDAP Directory

JIRA's customisable workflow can also be very useful:

- Configuring Workflow

The workflow can be customised for each project, and can be used to better reflect the business process of each support team in JIRA. For example, if issues can only have 2 stages (Open and Closed) then it is far better to create and use a custom workflow rather than use the JIRA's default workflow.

Using JIRA's flexible plugin system it is also possible to extend JIRA's functionality in regards to workflow. One place where this can become useful, is when closing issues in the hot-line project that have linked issues in one or more of the 2nd level support projects. It is possible to write a custom Workflow Condition that will look at all the linked issues and only allow an issue to be Closed when the linked issues are also closed. This will ensure, that the issues in the hot-line project
are only closed when the linked issues are handled by the respective 2nd level support team. For more information on creating custom workflow elements (e.g. Workflow Conditions) please see:
How to create Custom Workflow Elements for JIRA 3

If one of the support teams also has an existing support system in place that they would like to continue using, it should be possible to integrate that system with JIRA. JIRA has a number of extension points that can be used to communicate (and hence integrate) with external systems:

- Extending JIRA

By default, JIRA related issue links do not affect workflow, so users can close issues even if other open issues are listed as blocking it. You can enforce the rule that all blocking issues must be resolved before you can resolve the parent issue using the custom 'blockingLinksClosed Condition' workflow plugin.

Further Support Discussion

- How Atlassian Uses JIRA For Support
- Example Helpdesk/Support System Project Configuration

Related Best-Practice Discussions

No content found for label(s) best-practices.

This document is a work in progress. Feel free to add any comments at the bottom.

Jelly Escalation

Below are the two Jelly scripts used by Atlassian's support system to automatically close issues after a certain period. These Jelly scripts are then run with the built in Jelly Service.

Make an issue inactive

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib"
xmlns:core="jelly:core" xmlns:log="jelly:log" >
<jira:Login username="atlassiansupport" password="[your password]">
    <log:warn>Running Inactivate issues service</log:warn>
    <!-- Properties for the script -->
    <core:set var="comment">This issue has not been updated for 5 business days.

If you have an update, please use "Add Comments For Atlassian" action to let us know. If you
need more time to gather information please let us know and we will 'freeze' this issue. If you
have no other questions, please Close this issue.

If no update is received in the next 5 business days, this issue will be automatically closed.

Thank you,

  The Atlassian Support Team</core:set>
    <core:set var="workflowStep" value="Mark Inactive" />
    <core:set var="workflowUser" value="atlassiansupport" />
    <core:set var="filter7Days" value="11505" />

    <!-- Run the SearchRequestFilter -->
    <jira:RunSearchRequest filterid="${filter7Days}" var="issues" />

    <core:forEach var="issue" items="${issues}">
        <log:warn>Inactivating issue ${issue.key}</log:warn>
        <jira:TransitionWorkflow key="${issue.key}" user="${workflowUser}"
workflowAction="${workflowStep}" comment="${comment}"/>
    </core:forEach>
</jira:Login>
</JiraJelly>
```

Close an issue

```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.enterprise.JiraTagLib"
xmlns:core="jelly:core" xmlns:log="jelly:log" >
<jira:Login username="atlassiansupport" password="[your password]">
    <log:info>Running Close issues service</log:info>
    <!-- Properties for the script -->
    <core:set var="comment">This issue has not been updated for 10 business days and will be
Closed.

If this issue has not been completed please reopen this issue and we will do our best to help
you as soon as we can.

Thank you,

  The Atlassian Support Team</core:set>
    <core:set var="workflowStep" value="711" />
    <core:set var="workflowUser" value="atlassiansupport" />
    <core:set var="filter7Days" value="11509" />

    <!-- Run the SearchRequestFilter -->
    <jira:RunSearchRequest filterid="${filter7Days}" var="issues" />

    <!-- Iterate over the issues -->
    <core:forEach var="issue" items="${issues}">
        <log:warn>Closing inactive issue ${issue.key}</log:warn>
        <jira:TransitionWorkflow key="${issue.key}" user="${workflowUser}"
workflowAction="${workflowStep}" comment="${comment}" resolution="Customer Timeout"/>
    </core:forEach>
</jira:Login>
</JiraJelly>
```

For more helpings of Jelly, see Jelly Examples.

Simple Escalation

Here is a piece of code that performs simple escalation. The code finds all issues that have been in the 'Open' status for longer than 24 hours and increases the priority of these issues (if there is a higher priority). This code could be used in a JIRA service so that it is performed

periodically.

Please note that the code assumes that all the issues use the default workflow. Hence it also assumes that the the step id it should search for in the OSCurrentstep table is 1. If your issues are using a different workflow you will need to see what status id to search for.

```
EntityCondition con = new EntityExpr(
        new EntityExpr("stepId", EntityOperator.EQUALS, new Long(1)),
        EntityOperator.AND,
        new EntityExpr("startDate", EntityOperator.LESS_THAN_EQUAL_TO, new
Timestamp(System.currentTimeMillis() - 24*3600*1000))
);

List steps = CoreFactory.getGenericDelegator().findByCondition("OSCurrentStep", con, null,
null);
for (Iterator iterator = steps.iterator(); iterator.hasNext();)
{
    GenericValue stepGV = (GenericValue) iterator.next();
    IssueManager issueManager = ComponentManager.getInstance().getIssueManager();
    GenericValue issueGV = issueManager.getIssueByWorkflow(stepGV.getLong("entryId"));

    // Increase priority
    ConstantsManager constantsManager = ComponentManager.getInstance().getConstantsManager();
    GenericValue priority =
constantsManager.getPriority(issueGV.getString(IssueFieldConstants.PRIORITY));
    Collection priorities = constantsManager.getPriorities();
    GenericValue higherPriority = null;
    for (Iterator iterator1 = priorities.iterator(); iterator1.hasNext();)
    {
        GenericValue priorityGV = (GenericValue) iterator1.next();
        if (priorityGV.getString("id").equals(priority.getString("id")))
        {
            if (higherPriority != null)
            {
                // Update issue
                issueGV.set(IssueFieldConstants.PRIORITY, higherPriority.getString("id"));
                // Save issue to database and fire an event
                GenericValue originalIssue = issueManager.getIssue(issueGV.getLong("id"));
                User updater = UserUtils.getUser("admin");
                IssueUpdateBean issueUpdateBean = new IssueUpdateBean(issueGV, originalIssue,
IssueEventType.ISSUE_UPDATED, updater);
                IssueUpdater issueUpdater = ComponentManager.getInstance().getIssueUpdater();
                issueUpdater.doUpdate(issueUpdateBean, true);

            }

            break;
        }
        else
        {
            higherPriority = priorityGV;
        }
    }
}
```

The above code will make change items of updated issues appear as if they have been performed by the "admin" user. You may wish to create a special user for this task.

Letting customers only create issues

This page describes a minor JIRA modification which redirects users to an arbitrary page after creating issues (and potentially other operations). It is mainly of interest to JIRA **Professional** and **Standard** users.

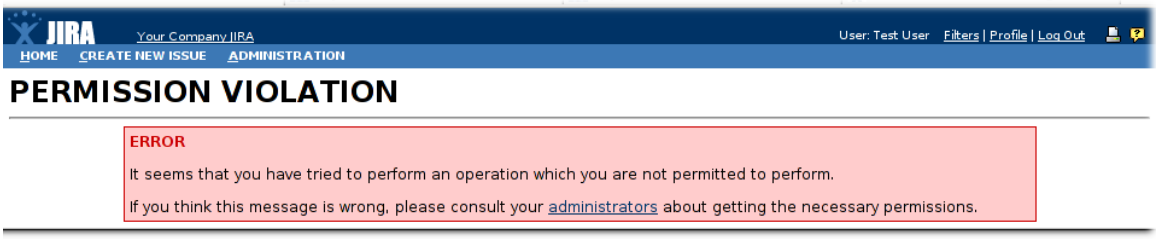
Scenario

When JIRA is used in a public environment, it is often useful for customers to be able to raise issues directly, but not see other customers' issues.

You can also grant the **Reporter** (and your company groups) the **Browse Issue** permission. Customers can then view issues they have raised.

In JIRA Professional and Standard, **Reporter** isn't available, and permissions can only be granted/denied per group. We want the **Create Issue** permission granted to everyone, but **Browse Projects** denied:

Users will see a permission error after creating an issue - not very customer-friendly!



Redirecting to a custom page

What we want is the ability to redirect the user to a nice "Thanks for raising an issue" page. We might want to direct to a different page depending on which groups the user is in. This can be done as follows:

Modify actions.xml

Open *atlassian-jira/WEB-INF/classes/actions.xml* in your JIRA Installation Directory. If you are using the JIRA WAR distribution, first copy *webapp/WEB-INF/classes/actions.xml* to *edit-webapp/WEB-INF/classes* in your JIRA Installation Directory and edit *actions.xml* there.

Locate the section:

```
<action name="issue.ViewIssue" alias="ViewIssue">
<view name="success">/secure/views/issue/viewissue.jsp</view>
<view name="rss">/secure/views/issue/viewissue-rss.jsp</view>
<view name="issuenotfound">/secure/views/issuenotfound.jsp</view>
<view name="permissionviolation">/secure/views/permissionviolation.jsp</view>

<command name="moveIssueLink" alias="MoveIssueLink">
<view name="error">/secure/views/issue/viewissue.jsp</view>
</command>
</action>
```

Modify the **permissionviolation** page to `/redirectusers.jsp`:

```
<view name="permissionviolation">/redirectusers.jsp</view>
```

Create a redirect JSP

Now create *atlassian-jira/redirectusers.jsp* (in your JIRA Installation Directory (or for JIRA WAR distributions, the *edit-webapp/redirectusers.jsp* of the JIRA Installation Directory), containing something like this:

```
<%@ page import="com.opensymphony.user.User"%>
<%
User user = com.opensymphony.user.UserManager.getInstance().getUser(request.getRemoteUser());
if (user.inGroup("customerA-users")) {
response.sendRedirect("http://localhost/thankyou.jsp?user="+user);
} else {
response.sendReirect("http://localhost/thankyou.jsp");
}
%>
```

Your logic group(s) to check for and redirect URLs will be different. If you don't want to create a custom page, you can redirect to *request.getContextPath()+"/secure/Dashboard.jspa"*

Deploy

Simply restart JIRA (or if you are using the WAR distribution, run *build.bat* or *build.sh* to regenerate the .war file and redeploy this in your application server).

Linking to local file under Firefox

There is a new KB article related to this topic which contains updated information. Please review that if you have questions about linking to file:// URLs from within JIRA:

KB Article: Can't Link to Local Files from within JIRA

Wiki markup allows you to links to files on the network / server with the format:

```
[file:///c:/temp/foo.txt]
```

This works fine under Internet Explorer, but Firefox and Mozilla block links to local files for security purposes. If you are happy with the risk of linking to local content, you can override the security policy and also enable linking in Firefox

The instructions for this can be found at http://kb.mozillazine.org/Links_to_local_pages_don't_work and you may also want to check out the other network preferences.

Please note that you need to use full URL syntax for your link (from http://kb.mozillazine.org)

> *You also need to use proper URI syntax for local file references. It is not proper to enter an operating-system-specific path, such as c:\subdir\file.ext without converting it to a URI, which in this case would be* `file:///c:/subdir/file.ext`*. In general, a file path is converted to a URI by adding the scheme identifier file:, then three forward slashes (representing an empty authority or host segment), then the path with all backslashes converted to forward slashes.*

Login problems

I have manually reset a user's password, but the user still cannot login

Check (in Admin -> Global Settings -> Global Permissions) that the user belongs to a group that has the **JIRA Users** permission.

The user cannot get past the login page. After clicking the "Log In" button, the login page just refreshes.

This usually occurs when JIRA cannot set a browser cookie. Ensure that cookies are allowed in the user's browser settings.

If you are using IE6, check that your server name does not have an underscore ("_") in it, as IE6 has a problem with this (see JRA-1624).

Mail error - Unable to relay

I'm getting exceptions like "SMTPAddressFailedException: 550 5.7.1 Unable to relay for XXX@XXX". What does this mean?

The "Unable to relay" error means that your mail server doesn't allow relaying for the e-mail address that you are using for your SMTP server. (see http://www.chilkatsoft.com/faq/Smtp550.html). Please try getting your mail server admin to enable relaying for your e-mail address or use another address that has relaying enabled.

You can get more help on changing the e-mail addressed used by JIRA here.

Making JIRA login case insensitive for JIRA 3.13.x

As JIRA will depend on database whether to be case sensitive or insensitive, JIRA login for case sensitive database (eg. Postgres) will be case sensitive as well. To make the login page case insensitive, there are two files that need to be modified:

- `<jira-install>\atlassian-jira\WEB-INF\classes\templates\jira\dashboard\macros.vm` which control the loginform at the dashboard page (eg. http://localhost:8085/secure/Dashboard.jspa). Modify the file as below

```
<tr>
            <td valign="middle" align="center" colspan="2">
                <input id="login" type="button" onClick="CheckForm();"
value="$i18n.getText('common.concepts.login')" tabindex="4" />
            </td>
        </tr>

  <script type="text/javascript">
   function CheckForm() {

    var Username = document.loginform.os_username.value;

    document.loginform.os_username.value = Username.toLowerCase();

    document.loginform.submit();
   }
  </script>

        #if ($allowPasswordReset == true)
        <tr>
            <td valign="middle" align="right" width="25%"> </td>
            <td valign="top"><font size="1"><a
href="${baseurl}/secure/ForgotPassword!default.jspa">$i18n.getText('common.concepts.forgotpas
</tr>
        #end
```

- `<jira-install>\atlassian-jira\includes\loginform.jsp` which control the loginform in the middle (eg. http://localhost:8085/login.jsp?os_destination=%2Fbrowse%2FTST-1). Modify the file as below

```
<tr>
   <td valign="middle" align="center" colspan="2">
    <input id="login" type="button" onClick="CheckForm();" value="<webwork:text
name="'common.concepts.login'"/>" tabindex="4">
   </td>
  </tr>

  <script type="text/javascript">
   function CheckForm() {

    var Username = document.loginform.os_username.value;

    document.loginform.os_username.value = Username.toLowerCase();

    document.loginform.submit();
   }
  </script>

  <%
   if
(!ManagerFactory.getApplicationProperties().getOption(APKeys.JIRA_OPTION_USER_EXTERNALMGT)
&&

!ManagerFactory.getApplicationProperties().getOption(APKeys.JIRA_OPTION_USER_PASSWORD_EXTERNA
{
  %>
  <tr>
   <td valign="middle" align="right" width="25%"> </td>
   <td valign="top"><font size="1"><a href="<%= request.getContextPath()
%>/secure/ForgotPassword!default.jspa"><webwork:text
name="'common.concepts.forgotpassword'"/></a></font></td>
  </tr>
  <% } %>
```

Restart JIRA after the modification. If it does not take effect, delete the work directory and restart JIRA.

Outward Link Description and Inward Link Description

When creating a new Issue Link Type, you need to specify an **Outward Link Description** (e.g. "duplicates") and an **Inward Link Description** (e.g. "is duplicated by").

What do these mean?

When a JIRA user links two issues together,

- the **Outward Link Description** applies to the issue from within which they clicked "Link this issue to another issue".
- the **Inward Link Description** applies to the issue that they choose to link to.

Parsing utf-7 emails

Some users report having problem parsing `unicode-1-1-utf-7` (aka `utf-7`) emails. JIRA breaks with a stacktrace like:

```
2007-01-31 12:54:59,176 JiraQuartzScheduler_Worker-2 ERROR
[service.util.handler.CreateIssueHandler] Could not create issue!
java.io.UnsupportedEncodingException: unicode-1-1-utf-7
at sun.nio.cs.StreamDecoder.forInputStreamReader(StreamDecoder.java:52)
at java.io.InputStreamReader.<init>(InputStreamReader.java:83)
at com.sun.mail.handlers.text_plain.getContent(text_plain.java:64)
at javax.activation.DataSourceDataContentHandler.getContent(DataHandler.java:774)
at javax.activation.DataHandler.getContent(DataHandler.java:521)
at javax.mail.internet.MimeBodyPart.getContent(MimeBodyPart.java:603)
at
com.atlassian.jira.service.util.handler.AbstractMessageHandler.handleMultipart(AbstractMessageHandl
com.atlassian.jira.service.util.handler.AbstractMessageHandler.createAttachmentsForMessage(Abstract
com.atlassian.jira.service.util.handler.CreateIssueHandler.handleMessage(CreateIssueHandler.java:20
com.atlassian.jira.service.util.handler.CreateOrCommentHandler.handleMessage(CreateOrCommentHandler
com.atlassian.jira.service.services.mail.MailFetcherService.run(MailFetcherService.java:190)
at com.atlassian.jira.service.JiraServiceContainerImpl.run(JiraServiceContainerImpl.java:67)
at com.atlassian.jira.service.ServiceRunner.execute(ServiceRunner.java:48)
at org.quartz.core.JobRunShell.run(JobRunShell.java:191)
at org.quartz.simpl.SimpleThreadPool$WorkerThread.run(SimpleThreadPool.java:516)
```

The solution

Hopefully one day, Sun will include support for this encoding natively (see this Sun bug report, but in the meanwhile you can install a library to get this working. Installation is quite simple:

1. Download the jutf7 jar from http://sourceforge.net/projects/jutf7
2. Copy the jar to your $JAVA_HOME/jre/lib/ext directory. **No other directory will do** - it has to be in this (lowest)classloader to be picked up.
3. Restart JIRA (or Confluence, or whatever is parsing the emails).

Project-specific email templates

Using email notifications, can separate templates be setup for projects or events?

Unfortunately templates are currently global. We anticipate adding this feature to JIRA in future.

QuickSearch guesses the issue key prefix (sometimes)

The Quick Search box (at the top-right of your JIRA screen) can sometimes find issues when you type just the number (e.g. '53'). Other times, you need to type the prefix too (e.g. 'JRA-53').

This is due to the concept of a 'selected project' - a bit of JIRA magic if you like. Basically, if you have recently done something in a project, that project becomes your 'selected project'. JIRA tries to 'guess' which issue you are looking for, given the 'selected project'.

But if you've just logged into JIRA, and not yet gone to an issue or a project, you will need to type the complete issue key (including the prefix).

Receiving a Daily Summary of Updated Issues

Some people may prefer to receive a daily summary of updated issues, rather than continual notifications each time an issue is updated. To do this, you will need to:

- 1. Set up your search criteria
- 2. Save your search as a 'Filter'
- 3. Subscribe to your Filter
- 4. (Optional) Stop the continual notifications

1. Set up your search criteria

For example, to find all issues that have been updated in the past 24 hours, use the following Advanced Search query:

```
updated >= "-24h"
```

Or, to find all issues in the "ACME" project that have been updated in the past 24 hours, use the following Advanced Search query:

```
project = "ACME" and updated >= "-24h"
```

2. Save your search as a 'Filter'

Click the '**Save**' link in the '**Operations**' column. Type a name for your new filter (e.g. "Joe's Daily Updates"), then click the '**Save**' button.

- For further details, please see Saving Searches ('Issue Filters').

3. Subscribe to your Filter

Once you haved saved your new filter, click the '**Subscriptions**' link in the '**Operations**' column. Click '**Add subscription**', adjust the default settings if you need to, then click the '**Subscribe**' button.

- For further details, please see Receiving Search Results via Email.

4. (Optional) Stop the continual notifications

If you don't want to receive continual updates each time an issue is updated, your name will need to be removed from the appropriate Notification Schemes.

Recieving Notification for Select Issues or Updates

To receive notifications of updates on issues meeting a set of criteria (For example, watching the output of a particular user), create an issue filter meeting said criteria and either subscribe to it by mail or add the RSS feed to a newsreader software.

For example, to search a list of issues with the following criteria:

```
Assignee: username
Updated from: -1d (updated within the last 24 hours)
```

The filter subscription will periodically send a notification reporting the issues assigned to username which have been updated within the last 24 hours. For more information, please refer to:

- Saving Searches ('Issue Filters')
- Receiving Search Results via Email
- Using the Issue Navigator

Removing Commas for Values Held in Number Field Custom Field Type

JIRA adds commas to numeric value stored in Number field, like 1,234. For further reference see JIRAKB:JRA-7582.

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

This workaround will apply to all Security drop-down lists in the instance.

If you don't wan commas to be added, edit `/WEB-INF/classes/templates/plugins/fields/view/view-number.vm`. Replace the following line:

```
$!numberTool.format($value)
```

with:

```
$value.longValue()
```

Make sure to back up the velocity file before changing it. Keep in mind the notes from Modifying JIRA Templates and JSPs.

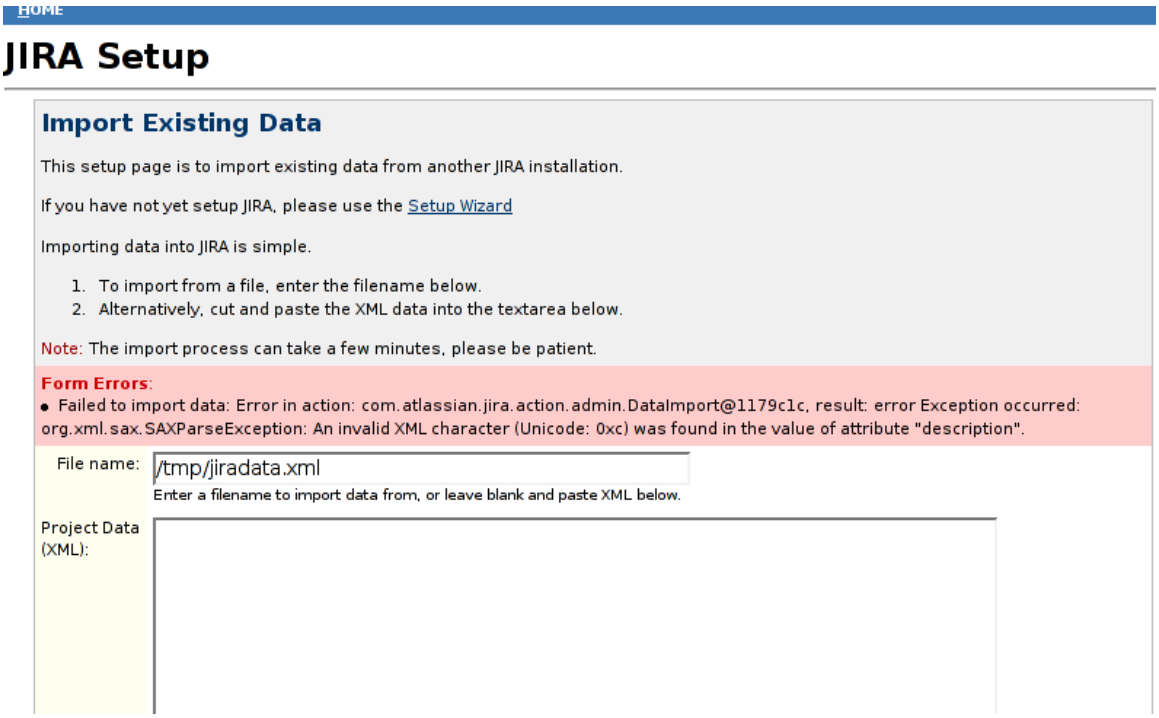
RELATED PAGES

No content found for label(s) jira-custom-velocity.

Removing invalid characters from XML backups

JIRA 3.1 and above should not suffer from this problem. Invalid characters are automatically stripped from imported data

In older versions of JIRA it was possible to cut & paste text containing control characters into JIRA issue fields. This causes problems, because JIRA's backup format is XML, and XML does not allow for the storage of most control characters . When XML containing control characters is imported into JIRA, the import fails with an error:



To fix this, one needs to remove the control characters from the JIRA backup file. This can be done as follows:

- Download atlassian-xml-cleaner-0.1.jar
- Open a command prompt and locate the XML or ZIP backup file on your computer, here assumed to be called jiradata.xml
- Run:

```
java -jar atlassian-xml-cleaner-0.1.jar jiradata.xml > jiradata-clean.xml
```

This will write a copy of jiradata.xml to jiradata-clean.xml, with invalid characters removed. You should now be able to import jiradata-clean.xml without problems.

Removing NONE from the Issue Security Drop-Down List

There are some instances where the 'NONE' in the Issue Security Drop-Down list must be removed. By default in JIRA you cannot do this from the web interface. See JRA-5332 for a discussion on this feature request.

Customisations to Velocity templates or other JIRA files are not included in the scope of Atlassian Support.

Deploying Velocity Templates without a Restart

In a development instance, you can play with picking up velocity file changes without a restart.
From `<jira-install>/atlassian-jira/WEB-INF/classes/velocity.properties`:

1. Change class.resource.loader.cache from true to false
2. Uncomment (remove the # sign from) #velocimacro.library.autoreload=true

Keep in mind that the next time you upgrade JIRA – or need a new installation for any reason – you will have to manually copy any changes you have made to the JSPs or tempates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your customization into them.

This workaround will apply to all Security drop-down lists in the instance.

To remove the field, edit `<atlassian-jira/WEB-INF/classes/templates/jira/issue/field/securitylevel-edit.vm`. Delete the lines:

```
<option value="$!noneLevelId"
        #if ($noneLevelId && $security && $security == $noneLevelId )selected#end
        >$i18n.getText('common.words.none')</option>
```

Make sure to back up the velocity file before changing it. Keep in mind the notes from Modifying JIRA Templates and JSPs.

RELATED PAGES

No content found for label(s) jira-custom-velocity.

Re-order workflow transactions

If you need to re-order the workflow actions of a workflow step, for example:

From:

- action1
- action 2
- action 3

To:

- action 2
- action 3
- action 1

...You cannot do this from the web interface, but you can do it.

- Download the workflow as an XML file
- Open up the XML file and change the order of the `<common-actions>` and `<actions>` for every `<step>`
- Upload the XML and view the changes

There are some details here in our documentation.

Resolved issues appearing in Open issues filters

Symptoms

If you find that supposedly "Resolved" issues are appearing in an open issues filter, your customized workflow may not be properly configured.

JIRA regards an "open" issue to be one without a resolution. With a standard JIRA workflow, this means all statuses except Resolved and Closed. When Resolving or Closing an issue, you are presented with a transition screen containing the Resolution field, which you must set to complete that transition. See the default workflow as an example.

It is possible to reconfigure JIRA such that Resolved issues, for example, do not have a resolution. This can happen in two ways:

- The user creates a custom workflow, and doesn't prompt the user for a Resolution on the resolve screen.
- The user hides the Resolution field in the field configuration, so it never appears to users.

Resolution

The long-term solution is to fix the workflow to present the Resolution at every transition into a non-open status. If the Resolution field is hidden, it should be made visible.

Existing issues in Resolved or Closed that have no resolution can be fixed by reopening and reclosing with a resolution.

Here is the process, assuming issues in Resolved and Closed statuses without a resolution set:

1. Do a search for issues in status "Closed", with Resolution "unresolved", which will show affected issues.
2. On the right, you'll see the text "Bulk Change: all X issue(s)". Open that link twice, eg. the same page in two tabs or two browser windows.
3. In one page:
   o Click the checkbox to select all issues, and click "Next >>".
   o For Operation, choose "Transition Issues", and then choose "Reopen Issue"
   o Uncheck "Send mail for this update"
   o Click confirm.
4. Now in the second page (displaying that original set of issues):
   o Click the checkbox to select all issues, and click "Next >>".
   o Operation, choose "Transition Issues", and then choose "Close Issue"
   o Select a resolution (eg. "Fixed").
   o Uncheck "Send mail for this update"
   o Click confirm.

By doing this you have reopened and closed the issues, setting a resolution. The Closed issues should now no longer appear on your dashboard.

Repeat the same process, but selecting "Resolved" issues this time (and transitioning through Reopen and Resolve).

Restricting the Visibility of Worklog on an Issue

To restrict the visibility of worklog done on an issue, adjust the 'Log Viewable By' field to specify which users can view the log work information in an issue. For more information, please refer to Logging Work on an Issue.

Retrieving the JIRA Administrator

On this page:

- Scenario A: I don't know which user has the *JIRA Administrators* or *JIRA System Administrators* global permission
  - If there are no JIRA Administrators
- Scenario B: I know which user has the *JIRA Administrators* or *JIRA System Administrators* global permission, but I have forgotten the password
  - 1. Send it via email
  - 2. Set the password directly in the database

Scenario A: I don't know which user has the *JIRA Administrators* or *JIRA System Administrators* global permission

You first need to find out which group(s) have been granted the global permission.

The *JIRA System Administrators* global permission was added to JIRA in version 3.12. Anyone granted the *JIRA System Administrators* global permission can perform **all** administration tasks in JIRA, whereas anyone granted the *JIRA Administrators* global permission can perform most but not all administration tasks. Prior to version 3.12, anyone granted the *JIRA Administrators* global permission can perform **all** administration tasks.

To find out which group(s) have been granted the *JIRA Administrators* global permission, run the following database query:

```
select perm_parameter from schemepermissions where PERMISSION=0;
```

To find out which group(s) have been granted the *JIRA System Administrators* global permission, run the following database query:

```
select perm_parameter from schemepermissions where PERMISSION=44;
```

Now that you know which group(s) have the global permission, run the following database query to find out which users are in that group (replace "jira-administrators" with the group returned by the above query):

```
select child_name from cwd_membership where parent_name='jira-administrators';
```

If you don't know the password for the user(s) returned by this query, move on to Scenario B.

If there are no JIRA Administrators

If you're using Crowd or an external user management system, there may be no users with administrator permissions.
Find the groups in the external management system that you want to grant the administrator permissions and do the following:

1. Shutdown JIRA.
2. Use SQL to assign the appropriate group to the administrator permissions similar to this:

```
update schemepermissions set perm_parameter='jira-system-administrators' where
permission=44;
update schemepermissions set perm_parameter='jira-administrators' where permission=0;
update schemepermissions set perm_parameter='jira-users' where permission=1;
```

3. Restart JIRA.

If you are still not able to log in with the user, please make sure that the new user is present in the '*userbase*' table and is member of the group '*jira-users*' in the '*membershipbase*' table.

Scenario B: I know which user has the *JIRA Administrators* or *JIRA System Administrators* global permission, but I have forgotten the password

Note that this will only work for users in the Internal directory. The following methods will not work with external user directories (eg in an LDAP server), since authentication is performed externally. You can find which directory a user belongs to by comparing the `directory_id` in `cwd_user` to those in `cwd_directory`.

Here are two different ways you can solve this problem:

1. Send it via email

This is the recommended approach

If you have configured JIRA to send email, just click on the **Forgot Password** link on the login page, enter your username and click the **Send it to me** button. You will receive an email which will help you reset your password.

2. Set the password directly in the database

This is a last resort only.

You can also update the password hash stored for a user in your database. Run the following command to set the user called XXXX's password to the word *sphere*.

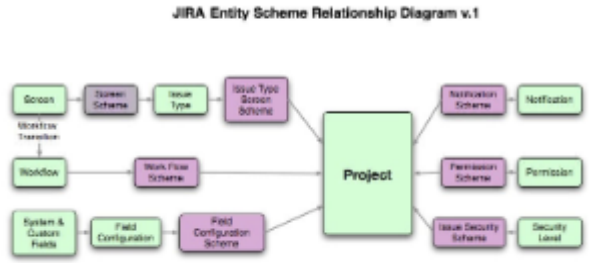
```
update cwd_user set
credential='uQieO/1CGMUIXXftw3ynrsaYLShI+GTcPS4LdUGWbIusFvHPfUzD7CZvms6yMMvA8I7FViHVEqr6Mj4pCLKAFQ=
where user_name='XXXX';
```

Then restart your JIRA instance.

Scheme Entity Relations Map

This diagram illustrates the relationships between various JIRA entities and schemes.

JIRA Scheme Entity Relations Map (click to view larger image)



You may also wish to view the following user-contributed diagram:

JIRA Scheme Entity Relations Map - user-contributed (click to view larger image)

Sending JIRA Data to Support

To replicate reported problems, Atlassian support staff may ask you for a copy of your JIRA data.

Automatic Support Request (Preferred)

To send an anonymous backup automatically,

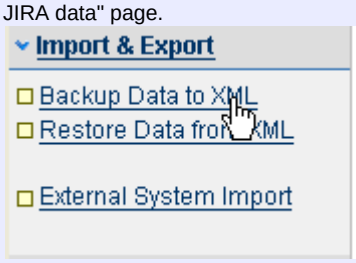
1. From **Administration**, select **Support Request**.
2. Make sure to select the existing ticket number and to include an XML backup, as you fill out the form.

As of JIRA 4.1.1, it is no longer possible to send data via the **Administraion** -> **Support Request** page. Please see below for instructions on providing a manual XML Backup.

Manual XML Backup (Recommended For Email Filters or Large Backups)

To create an anonymous a backup locally,

1. Login as a user with global administrator access.
2. Bring up the administration page by clicking either on the "Administration" link on the top bar or the title of the Administration box on the dashboard:

3. On the panel on the left, under the title "Import & Export", click "Backup Data to XML". This will bring up the "Backup JIRA data" page.


4. In the form, fill in the File path data entry box with a full path, including filename, that JIRA can write to.
5. Select the "Backup as Zip" checkbox. (if you want to anonymize the data, see below).
6. Click the Backup button, and be patient.
7. JIRA will return, confirming that it has written out the content to the file specified above.
8. Attach the generated file on disk to a support request. As the support site runs over SSL this is more secure than email, and you can remove attachments if you like.

Anonymizing data

For JIRA versions 3.7 through to 4.1, data sent via Administration -> Support Request is anonymised by default, and it is thus the easiest route to sending us anonymised data (but be sure your mail server has a username/password specified, so relaying is allowed).

As of JIRA 4.1.1, it is no longer possible to send data via the Administration -> Support Request page. To do so, please login to Atlassian Support and attach the data to the existing issue.

Support requests are resolved **much** faster if a data export is provided. However, with sometimes this is not an option because the data contains sensitive information.

In JIRA 3.7.x to 4.1, JIRA automatically anonymises data sent to Atlassian from the Administration -> Support Request page. For earlier or later versions, or people who want to anonymise JIRA data from the command-line, we've created a data 'anonymiser', which replaces most text in JIRA XML backups with **x**'s.

The anonymiser can be downloaded from here.

Unzip the package, then open a console and in the `jira_anon` directory run:

```
$ java -jar joost.jar <name of your backup file.xml> anon.stx > <name of the anonymised backup
file to be generated.xml>
```

For example:

```
$ java -jar joost.jar backup.xml anon.stx > anon-backup.xml
```

Then zip the generated backup XML file, and attach it to a support case on https://support.atlassian.com

The anonymiser currently replaces the following text with x's:

- Issue summary, environment, and description
- Comments, work logs, change logs
- Project descriptions
- Descriptions for most elements (notification schemes, permission schemes, resolutions)
- Attachment file names.
- "Unlimited text" custom fields

Check anon-backup.xml to ensure it's clean enough for your needs before you send to us.

Problems?

Invalid XML Characters

If, when you run the anonymiser, you get an error indicating that there are invalid XML characters in the XML backup of your database, run our utility to remove invalid XML characters first before anonymising.

Out of Memory / Heap Space Errors

If creating your anon-backup.xml partway through, you are likely facing a memory limitation with running the 'java' command with the default settings. To allow the command more memory for the command, simply add arguments after the 'java' command, like so:

```
$ java -Xms512m -Xmx512m -jar joost.jar backup.xml anon.stx > anon-backup.xml
```

Note: you may need to adjust the memory allocation beyond '512m' if the process continues to fail.

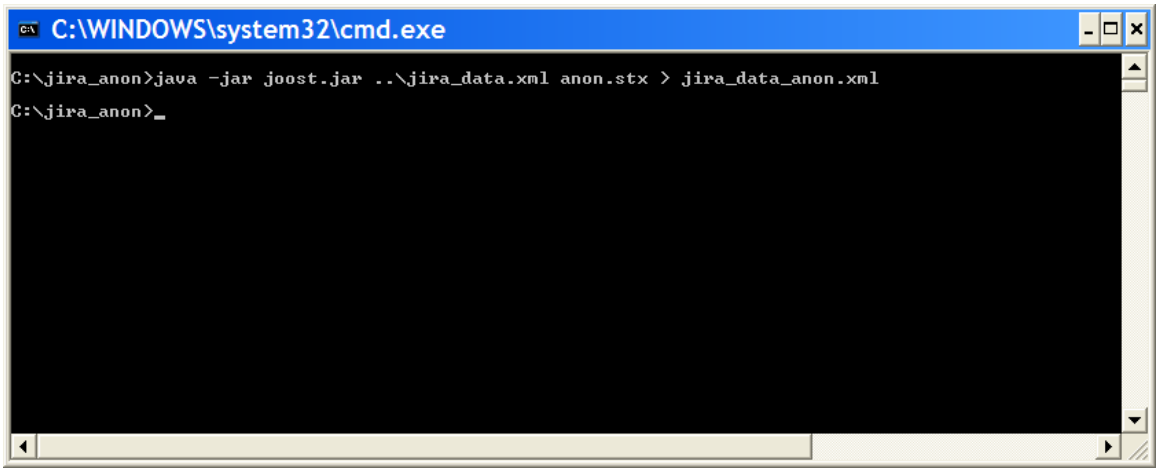
Java Version

You will need Java 1.4 or above to run this. You can check your Java version by running `java -version`, eg:

```
$ java -version
java version "1.5.0_07"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_07-b03)
Java HotSpot(TM) Client VM (build 1.5.0_07-b03, mixed mode, sharing)
```

If you find yourself using JDK 1.3 or earlier, check your path (`echo %PATH%` on Windows, `echo $PATH` on Unix) and ensure that the right version of Java is at the beginning. See the docs for more info on setting up Java.

The screenshot below is a simple example of how it is run in the command prompt of Windows XP:



Setting Additional Fields for Issues Created from Email

To set the issue's assignee from e-mail, set the ccassignee name parameter in the comment handler for the POP/IMAP service used to create issues. Refer to Creating Issues and Comments from Email for more information.

The JIRA Advanced Mail Handler allows users to define fields for the issues created email such as reporter, issue type, priority, summary, description and more.

The priority can be set on a scale 1-5 by setting the X-Priority on the email itself. See your mail server's documentation for tips on how to set the X-Priority. A related discussion appears in JRA-7316.

Setting a Default Value in the Description Field

This page describes how to set a Default Value for the Description field in JIRA. There's a feature request for this at JRA-4812. Please watch and vote on the issue if this is important to you.

There are workarounds to add this functionality. The modification does not persist if you upgrade JIRA so you'll have to re-apply these steps again in the future. Here are directions to set a default value for your Description field in JIRA:

1. Locate and backup the file: WEB-INF/classes/templates/jira/issue/field/description-edit.vm
2. Open that file:

```
#controlHeader ($action $field.id $i18n.getText($field.nameKey) $fieldLayoutItem.required
$displayParameters.get('noHeader'))

## setup some additional parameters
$!rendererParams.put("rows", "12")
$!rendererParams.put("wrap", "virtual")

## let the renderer display the edit component
$rendererDescriptor.getEditVM($!description, $!issue.key, $!fieldLayoutItem.rendererType,
$!field.id, $!field.name, $rendererParams, false)

#controlFooter ($action $fieldLayoutItem.getFieldDescription()
$displayParameters.get('noHeader'))
```

3. Add a section like this:

```
#if($description == '')
#set ($description = 'Put stuff here:')
#end
```

So, ultimately it should look something like:

```
#controlHeader ($action $field.id $i18n.getText($field.nameKey) $fieldLayoutItem.required
$displayParameters.get('noHeader'))

## setup some additional parameters
$!rendererParams.put("rows", "12")
$!rendererParams.put("wrap", "virtual")


#if($description == '')
#set ($description = 'Put stuff here:')
#end

## let the renderer display the edit component
$rendererDescriptor.getEditVM($!description, $!issue.key, $!fieldLayoutItem.rendererType,
$!field.id, $!field.name, $rendererParams, false)

#controlFooter ($action $fieldLayoutItem.getFieldDescription()
$displayParameters.get('noHeader'))
```

4. Restart your JIRA instance.

Adding multi-line values

If you wish to display in the description a default value of:

```
Step 1

Step 2

Step 3
```

you'll need to tweak the above instructions a bit:

```
#set ($description = "Step 1\
\
Step 2\
\
Step 3")
#set ($description = $description.replace('\',' '))
```

Setting Priority field value based on customfield value

Please note that adding Javascript to custom fields is a customisation and not maintained as a supported part of JIRA.

Check the comments below. This page has been flagged as outdated.

As the users are neglecting the description of the Priority field, a more detail custom field is created to represent the Priority field. Depending on the radio button custom field selected, the Priority field value is set.

1. Create a customfield name "Severity" at `Browse >> Administration >> Issue Fields >> Custom Fields`
2. Configure the radio button custom field to have a field options
3. Check the customfield ID in the Customfield table from the database by using the following SQL query:

```
SELECT id FROM customfield WHERE cfname="Severity";
```

4. Modify the following javascript code so that **<id>**will be replaced with the id of the custom field found from the first step:

```
<script type="text/javascript" charset="utf-8" id="priorityCustomFieldScript">
    function setIssuePriorityAndSubmit(e)
    {
        // set the value of the priority field here:
 if(document.getElementById("customfield_<id>_1").checked)
 {
  document.getElementById("priority").selectedIndex = 0;
 }
 else if(document.getElementById("customfield_<id>_2").checked)
 {
  document.getElementById("priority").selectedIndex = 1;
 }
 else if(document.getElementById("customfield_<id>_3").checked)
 {
  document.getElementById("priority").selectedIndex = 2;
 }
 else if(document.getElementById("customfield_<id>_4").checked)
 {
  document.getElementById("priority").selectedIndex = 3;
 }
 else if(document.getElementById("customfield_<id>_5").checked)
 {
  document.getElementById("priority").selectedIndex = 4;
 }
    }

    function hidePriorityAndAddSeverityHook()
    {
 var row = document.getElementById("priorityFieldArea");
 row.style.display = 'none';

 AJS.$("#customfield_<id>_1").parents('form').submit(setIssuePriorityAndSubmit);
    }

    var currentLocation = window.location.href;
    if (currentLocation.indexOf('CreateIssue') > -1 ||
currentLocation.indexOf('EditIssue') > -1) {
        AJS.$(document).ready(hidePriorityAndAddSeverityHook);
    }
</script>
```

5. Paste the javascript into the description of the "Severity" customfield at `Browse >> Administration >> Issue Fields >> Custom Fields`.

There is no need to hide or remove the 'Priority' field from the screen. The javascript code will hide the Priority field by itself.

Showing Extended Timestamp in the Created Column of the Issue Navigator

This document describes how to modify the *Created* date field in the Issue Navigator to include the time. By default, the column view of the field hard-codes the rendering format to the locale specific "Day" format.

Procedure for JIRA 4.2 and Before

Edit the file `JIRA_INSTALL_DIR/atlassian-jira/WEB-INF/classes/templates/jira/issue/field/created-columnview.vm` to always use the extended *formatDMYHMS* rather than *formatDMY*.
For example, the following macro fragment should be changed:

Original - short data format

```
#if ($created)
    #if (${displayParams.excel_view})
        $outlookDateManager.getOutlookDate($authcontext.getLocale()).formatDMYHMS($created)
    #else
        $outlookDateManager.getOutlookDate($authcontext.getLocale()).formatDMY($created)
    #end
#else
     
#end
```

Edited - full time-stamp

```
#if ($created)
    $outlookDateManager.getOutlookDate($authcontext.getLocale()).formatDMYHMS($created)
#else
     
#end
```

For JIRA 4.3 and Later

Edit the file `JIRA_INSTALL_DIR/atlassian-jira/WEB-INF/classes/templates/jira/issue/field/date-columnview.vm`:

```
#if ($value) <span title="${title}"><time datetime="${iso8601}">${title}</time></span> #else
  #end
```

Single Sign-on

Single Sign-on Information

Tracking the Time Taken for Each Workflow Transition

There are 2 plugins available in JIRA Extension Page which might be able to fulfill the requirement:

- JIRA Charts can report Average Time in Status
- The JIRA Suite Utilities has a Transitions Summary Tab Panel.

Troubleshooting Issue Creation Via Email

1. Is the message reaching the e-mail account?

Have your mail server administrator confirm that e-mail sent to the account JIRA is using is successfully reaching the account's Inbox.

2. Is the Create Or Comment Handler service configured correctly in JIRA?

Please review this guide to confirm this:

Creating Issues and Comments from Email

3. Are permissions set properly?

Does the user submitting the issue have Create Issue permissions in the Permission Scheme? If you are having troubles adding comments,

make sure your Issue Security Scheme is not restricting the user's access to the issue.

4. Still not working? Enable debug logging in JIRA:

First, we need to change the com.atlassian package from the *WARN* logging level to *DEBUG*. This can be done from the following menu:

Administration -> System -> Logging & Profiling -> Click Edit next to the com.atlassian package

5. Send two e-mails to the email address that JIRA is checking for new issues and comments. Wait 5 minutes and then submit a support request that includes the JIRA logs.

This can be done from the following menu:

Administration -> System -> Support Request

Remember to check the *Attach JIRA logs* box! Also, please note the e-mail address being used for testing and copy/paste the JIRA service settings for this Create Or Comment handler:

Administration -> System -> Services

Example of Service Info

handler: Create Or Comment Handler
popserver: POP server - JSP
handler.params: project=JSP, issuetype=1, createusers=true, stripquotes=true, bulk=forward
usessl: No SSL
forwardEmail: jira-support@atlassian.com

6. Change com.atlassian back from *DEBUG* to *WARN*.

Administration -> System -> Logging & Profiling -> Click Edit next to the com.atlassian package

WARNING: Leaving com.atlassian in debug mode will result in VERY large log files!

Note. If you want to log on a protocol level (IMAP, POP3 or SMTP), please refer to Logging email protocol details.

Using JIRA to Manage reusable modules

Many software products use external modules that are shared with other software products. The external projects are often managed separately, and have their own versions and lifecycles. So the question of how to "map" this scenario to JIRA often arises.

Currently, the best way to solve this in JIRA would be to create a separate JIRA project for each module and application. Then create issues in each JIRA project as needed and use issue linking to link related issues. Using Issue Links, issues can be easily linked across projects.

JIRA also has a clone issue function which can be used to copy an issue. The cloned issue can be then moved to another project. This should save the trouble of manually duplicating issues.

To get an idea of where each product and each module is "up to", JIRA's dashboard can be used. For example, one could place a portlet for each JIRA project that shows all open scheduled issues. This way the dashboard will provide an overview of all outstanding work for each project.

If all relevant issues for external modules have an issue in the product's JIRA project the standard reports and project summary panels (e.g. Change Log and Release Notes) should have all the information they need to be useful. Otherwise, JIRA can be extended by creating a custom project tab panel and/or a report that can look at more than one JIRA project and produce desired summaries. If you decide to write a custom report this tutorial should be useful.

In future we hope to better support this style of project organization, eg. through shareable subprojects (JRA-1072). Please vote/add your thoughts to the issue to increase its popularity. Also, please refer to this document which explains the way Atlassian shcedules new features.

Using validators to make custom fields required on transition screens

Use the 'Fields Required' workflow validator that is packaged in the JIRA Suite Utilities.

Please note the following caveats regarding validation of data by the 'Fields Required' workflow validator at the time of issue creation:

- fields that you set up as "required fields" are not flagged as such in the form to the end-user
- such fields can be cleared at a later time, which is not what you may have intended
- plugins such as GreenHopper will not detect the requirement as implemented by the workflow validator, so may fail later during usage

The reason 3rd party tools are needed is because JIRA's interpretation of "required" from a project's Field Configuration on some custom field means that the field is now required across all screens available to that project, regardless if the screen doesn't actually display that particular field. 3rd party tools, like the JIRA Suite Utilities' 'Fields Required' validator, are effectively a more granular means to control fields at the step or screen level at a project, instead of at the project level by the Project's Field Configuration.

We already have users & groups defined elsewhere - can JIRA make use of these?

Yes. If you have users and groups defined elsewhere then you can either use an existing OSUser provider (such as LDAP or JDBC) or write your own if you have custom needs.

Where are the application server logs?

Please always provide us with both:

1. atlassian-jira.log (see Logging and Profiling)
2. the application server log file, as it can contain useful error information that is not in `atlassian-jira.log`

JIRA Log Location

The logs are written to the `log` subdirectory of your JIRA Home Directory (or elsewhere if you have configured a different location). You can view the location of the `atlassian-jira.log` in the **'File Paths'** section of the System Information page.

Application Server Logs

Finding the application server log file is application server-specific and in some cases operating system-specific. Here is a decision tree:

- **If you are on Windows**
  - ..and JIRA was installed as a **Windows Service**:
    - ..then the logs are in the `logs\stdout_*.log` file in your JIRA Installation Directory and `c:\WINDOWS\system32\atlassian-jira.log` (WINDOWS may be replaced by WINNT), or for JIRA WAR, in your Tomcat installation directory.
  - ..and JIRA was started via `start-jira.bat`:
    - ..then some logs are effectively being lost (to the popup DOS window, where it cannot be recaptured). Some logs do go to the `atlassian-jira.log` file in the current directory (wherever you ran `startup.bat` from) but this might not work if your current directory isn't writeable (eg. c:\WINNT\system32, the default). Even if you see an `atlassian-jira.log` it may be an old one, created from a previous startup. **If you value your sanity** (and ours) please install JIRA as a service, even if only to get all of the right logs appearing in a consistent place.
- **If you are on Linux or Solaris**
  - ..then the logs are in `logs/catalina.out` in your JIRA Installation Directory, or for JIRA WAR, under your Tomcat installation directory.

Why doesn't JIRA have a Severity field like Bugzilla?

Originally, JIRA did have both a Priority and a Severity field. The Severity field was removed for a number of reasons, but principally because it was confusing to business users. To a software developer, it seems obvious that the severity of the bug ("The system crashes completely") is unrelated to the priority of it ("There is a one in a million chance of this occurring"). However, JIRA succeeds so well because business users can actually use it. If you present a business user with these two fields, they are instantly confusing (which is why the Severity field was removed).

In order to re-implement Severity, you can create a select-list custom field, order it (with field layouts), put it on your filters (with column layouts) and indeed search and filter it (in the Navigator).

For details, please see Custom Fields.

Workflows Guidebook

Download a PDF guide with step-by-step examples of common custom JIRA workflows. Guide created for the Mastering JIRA Workflows talk at Atlassian Summit 2010.

You need flash player installed to preview ppt and pdf files

XML format for import & export files

Is the XML format for the import/export files (which also contains the configuration) documented?

Not as such - it is an XML version of the underlying entity model, pulled out of the database. As a result it is always changing with new fields and entities being added. The entity model itself is defined in **WEB-INF/classes/entitymodel.xml**

Installation Notes

Search the Installation Notes:

FAQs

- Configuring IIS with Tomcat
- Database Notes — These pages contain notes on configuring JIRA with various databases.
  - Incorrect database type specified
  - Restarting JIRA from the Setup Wizard
  - Database limitations on number of projects
  - JIRA and HSQL
    - Running SQL commands in a HSQL database — On rare occasions, one may wish to run raw SQL queries on a JIRA or Confluence database. This page describes how to obtain a SQL console for hsqldb databases, which are built into JIRA and Confluence for evaluation purposes.
  - JIRA and MS SQL Server 2005
    - Connecting to named instances in SQL Server
    - Error caused by SET NOCOUNT in MS SQL Server
    - MS SQL Server 2000 Startup errors
    - Setting Up a SQL Server 2005 database for JIRA
  - JIRA and MS SQL Server 2008
  - JIRA and MySQL
    - Configuring MySQL 5.1 to store non-ASCII characters
    - JIRA Cannot Connect to MySQL with Named Pipes Enabled
    - JIRA Cannot Create Issues when Using MySQL with Binary Logging
    - MySQL Administrator and Data Truncation Errors
    - MySQL Data Access Exception - Errcode - 17 occurs with JIRA
    - Setting Up a MySQL Database on Linux for JIRA
  - JIRA and Oracle
    - Configuring Datasource for Oracle 10g JDBC drivers
    - Restoring data using I-Net (Oranxo) Driver for Oracle
    - Store Workflow on Disk with Oracle 8 — A workaround for the problem of > 4000 character workflows in Oracle 8 is to store these on disk, instead of in the database.
  - JIRA and PostgreSQL
    - Setting up a PostgreSQL Database on Linux for JIRA
- How to Set Up SMTP Relay in Exchange 2007
- Installation Troubleshooting Guide
- Installing a LDAP server on Debian Linux for use with JIRA
- Installing Java on Ubuntu or Debian
- Installing JIRA on Mac OS X
  - Configure JIRA as service on Mac OS X
- Is Clustering or Load Balancing JIRA Possible
- java.lang.NoClassDefFoundError
- JVM and Appserver configuration info
- LicenseFactory error after upgrading JIRA
- Logging request headers
- Running multiple instances of JIRA on one machine
- Solaris ClassNotFoundException
- Windows cannot find -Xms128m

Configuring IIS with Tomcat

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian **can not guarantee providing any support for it**. Please be aware that this material is provided for your information only and using it is done so at your own risk.

It is possible to set this up rather painlessly and the main documentation that we provide covers most use cases. But sometimes there are a few IIS configurations that differ from the default.

You are not using the "Default Website" in IIS

If you are in this boat, you will need to mimic what the Jakarta ISAPI installer created for you in the default website.

There will need to be two virtual directories:

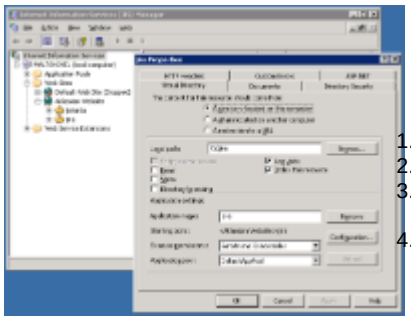
One named 'jira'

Follow these steps to setup the 'jira' virtual directory

1. Right-click on the website, go to New > Virtual Directory ...
2. The alias needs to be called **jira**
3. The path can point to any location, a temp directory, or perhaps your jira install location
4. Check the "Execute (Such as ISAPI application of CGI)", then next and you are now finished.

Now in the properties for the 'jira' virtual directory confirm:

1. The Execute Permissions is set to "Scripts and Executables" in the "Execute Permissions" section.



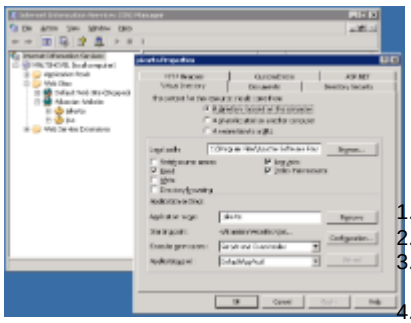
One named 'jakarta'

Follow these steps to setup the 'jakarta' virtual directory

1. Right-click on the website, go to New > Virtual Directory ...
2. The alias needs to be called **jakarta**
3. The path needs to point to the bin directory of the Jakarta Isapi Redirector, ie: C:\Program Files\Apache Software Foundation\Jakarta Isapi Redirector\bin
4. Check the "Execute (Such as ISAPI application of CGI)", then next and you are now finished.

Now in the properties for the 'jakarta' virtual directory confirm:

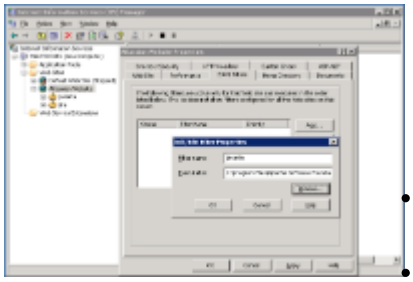
1. "Script source access" is checked
2. "Read" access is checked
3. The Execute Permissions is set to "Scripts and Executables"
4. The "Local Path" points to the bin directory of the Jakarta Isapi Redirector



You will also need to make sure that the non-default website has the Jakarta Redirector installed. This can be done by right-clicking on the non-default website, clicking 'properties' and then clicking on the "ISAPI Filters" tab.

From here you will need too:

1. Click the "Add..." button
2. Enter a filter name: **jakarta**
3. Browse to the "isapi_redirect.dll" file located here: C:\Program Files\Apache Software Foundation\Jakarta Isapi Redirector\bin
4. Click OK, Apply and then OK.



The final step is to restart the IIS Server, this can be done by opening the services.msc and clicking restart on "World Wide Web Publishing"

Gotcha's

- If you are using IIS 6.0 did you remember to add the *Jakarta Isapi Redirector* to the **Web Service Extension's** and set the extension status to *allow* ?
- Also for IIS 6.0 did you remember to add the *Jakarta Isapi Redirector* to the **ISAPI Filters** for the website?

- Is Tomcat listening on port 8009? Try the following from the command prompt to make sure:

```
netstat -na | findstr 8009
```

- Have you given JIRA a context in Tomcat's **server.xml**?

```
<Context path="/jira" docBase="${catalina.home}/atlassian-jira" reloadable="false">
```

And does it match the virtual directory and value in your **uriworkermap.properties** file?

```
/jira/*=wlb
```

Database Notes

These pages contain notes on configuring JIRA with various databases. They are supplementary to the JIRA documentation. If you've ever thought "I wish I'd known that when I started", please help others by adding a note to the relevant database page.

Recommended Databases

Please read the Supported Platforms document to see the list of databases that we recommend for use with your JIRA installation.

- Incorrect database type specified
- Restarting JIRA from the Setup Wizard
- Database limitations on number of projects
- JIRA and HSQL
  - Running SQL commands in a HSQL database — On rare occasions, one may wish to run raw SQL queries on a JIRA or Confluence database. This page describes how to obtain a SQL console for hsqldb databases, which are built into JIRA and Confluence for evaluation purposes.
- JIRA and MS SQL Server 2005
  - Connecting to named instances in SQL Server
  - Error caused by SET NOCOUNT in MS SQL Server
  - MS SQL Server 2000 Startup errors
  - Setting Up a SQL Server 2005 database for JIRA
- JIRA and MS SQL Server 2008
- JIRA and MySQL
  - Configuring MySQL 5.1 to store non-ASCII characters
  - JIRA Cannot Connect to MySQL with Named Pipes Enabled
  - JIRA Cannot Create Issues when Using MySQL with Binary Logging
  - MySQL Administrator and Data Truncation Errors
  - MySQL Data Access Exception - Errcode - 17 occurs with JIRA
  - Setting Up a MySQL Database on Linux for JIRA
- JIRA and Oracle
  - Configuring Datasource for Oracle 10g JDBC drivers
  - Restoring data using I-Net (Oranxo) Driver for Oracle
  - Store Workflow on Disk with Oracle 8 — A workaround for the problem of > 4000 character workflows in Oracle 8 is to store these on disk, instead of in the database.
- JIRA and PostgreSQL
  - Setting up a PostgreSQL Database on Linux for JIRA

Incorrect database type specified

Background

JIRA needs to know what kind of database it will be using, in order to generate database tables of the correct data types, and to generate correctly formatted SQL. The database type is specified in the `dbconfig.xml` file at the root of your JIRA Home Directory. For example:

```
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
 <name>defaultDS</name>
 <delegator-name>default</delegator-name>
 <database-type>hsql</database-type>
 <schema-name>PUBLIC</schema-name>
 <jdbc-datasource>
   <url>jdbc:hsqldb:/path/to/jira/database/jiradb</url>
   <driver-class>org.hsqldb.jdbcDriver</driver-class>
   <username>sa</username>
   <password></password>
   <pool-size>15</pool-size>
   <min-evictable-idle-time-millis>4000</min-evictable-idle-time-millis>
   <time-between-eviction-runs-millis>5000</time-between-eviction-runs-millis>
 </jdbc-datasource>
</jira-database-config>
```

In this example, JIRA expects to use HSQLDB (JIRA's internal database used for evaluation purposes).

If you've got it wrong ...

If you forgot to edit the `dbconfig.xml` file (see the documentation), then follow these steps to recover:

1. Fix the type in dbconfig.xml

Refer to the relevant documentation for your database in the Connecting JIRA to a Database section.

2. Fix the database

Is this the first time you have run JIRA?

If so, the **database has been created incorrectly**. Specifically, table columns have been created with incorrect data types, and you will see warnings like these in the logs:

```
2005-02-10 12:24:33,307 [core.entity.jdbc.DatabaseUtil] SQL Exception while executing the
following:
CREATE TABLE jiraaction (ID BIGINT NOT NULL, issueid BIGINT, AUTHOR VARCHAR, actiontype
VARCHAR, actionlevel
  VARCHAR, actionbody VARCHAR, CREATED TIMESTAMP, actionnum BIGINT, CONSTRAINT PK_jiraaction
PRIMARY KEY (ID))
Error was: java.sql.SQLException: You have an error in your SQL syntax.  Check the manual that
corresponds
  to your MySQL server version for the right syntax to use near ' actiontype VARCHAR,
actionlevel VARCHAR, actionbody VARCHAR, C
2005-02-10 12:24:33,308 [core.entity.jdbc.DatabaseUtil] Entity "ChangeGroup" has no table in
the database
2005-02-10 12:24:33,309 [core.entity.jdbc.DatabaseUtil] Could not create table "changegroup"
2005-02-10 12:24:33,309 [core.entity.jdbc.DatabaseUtil] SQL Exception while executing the
following:
CREATE TABLE changegroup (ID BIGINT NOT NULL, issueid BIGINT, AUTHOR VARCHAR, CREATED
TIMESTAMP, CONSTRAINT
  PK_changegroup PRIMARY KEY (ID))
Error was: java.sql.SQLException: You have an error in your SQL syntax.  Check the manual that
corresponds to
  your MySQL server version for the right syntax to use near ' CREATED TIMESTAMP, CONSTRAINT
PK_changegroup PRIMARY KEY (ID))
2005-02-10 12:24:33,310 [core.entity.jdbc.DatabaseUtil] Entity "ChangeItem" has no table in the
database
2005-02-10 12:24:33,310 [core.entity.jdbc.DatabaseUtil] Could not create table "changeitem"
```

The solution is to drop (delete) and recreate the database. When next restarted with the correct data types, JIRA will recreate the tables correctly.

Upgrading JIRA?

This situation is potentially problematic, because the newer version of JIRA may have added tables or columns with incorrect data types to your existing database schema.

The safest solution is to start a new database, and import an XML backup made before the upgrade.

If for some reason, you cannot import an XML backup (eg. your upgraded instance has been in production for a few days and contains new data), it is generally possible to patch the database by hand with SQL '`alter table`' statements. Please review the log files for information on what types JIRA expects, and what is actually present. JIRA will print this information every time it starts up. If in doubt, attach the logs and other relevant information to a support request on our support system.

Other situations

If this is not the first time JIRA has loaded, and you are not upgrading, you probably do not need to fix the data. After fixing the `dbconfig.xml` file, restart and check the logs for errors. If there are none, the database is fine.

Need help?

Please create a support request and attach the startup logs, your current `dbconfig.xml` file, and any other information relevant.

Restarting JIRA from the Setup Wizard

If you ever want to restart JIRA from the Setup Wizard again:

1. Stop JIRA by running either `bin\shutdown.bat` (for Windows) or `bin/shutdown.sh` (for Linux/Solaris) in your JIRA Installation Directory (or the Apache Tomcat installation directory running the JIRA WAR distribution).
   If JIRA is running as a JIRA or Tomcat service, stop the relevant service.
2. Make a backup of your JIRA Home Directory and JIRA Installation Directory.
3. Delete the `dbconfig.xml` file in your JIRA Home Directory.
4. Restart JIRA and access your JIRA server from a browser.
   JIRA will detect that no database configuration is present and will take you through the JIRA Setup Wizard again.

Database limitations on number of projects

Limitations on project totals imposed by databases:

Database	Limitation
Oracle	1000

SQL Server	No hard limit as a query can have any number of parameters, as long as it remains within the maximum batch size — which defaults to 65,536 * Network Packet Size (256MB). In practice, however, 2100 is the limit as this is the maximum number of parameters allowed by SQL Server.
MySQL	No hard limit; maximum query size has to be less than max_allowed_packet, which defaults to 1GB, but can be any value in the range 1024-1073741824 Bytes.
PostgreSQL	No hard limits; SQL query size is limited to the OS maximum file size.
HSQLDB	No limits specified, but in practice limited by heap memory.

JIRA and HSQL

This page has general notes on connecting JIRA to HSQL. It supplements the official HSQL installation documentation.

- Running SQL commands in a HSQL database

Running SQL commands in a HSQL database

On rare occasions, one may wish to run raw SQL queries on a JIRA or Confluence database. This page describes how to obtain a SQL console for hsqldb databases, which are built into JIRA and Confluence for evaluation purposes.

Locate the hsqldb directory

hsql stores its database in text files on the filesystem. Typically these will be in a `database` subdirectory of your JIRA Home Directory:

```
[jira-home-directory ~]$ ls -l database/
total 108
-rw-r--r--    1 jturner  jturner         0 Jul 28 09:12 jiradb.data
-rw-r--r--    1 jturner  jturner       343 Jul 28 09:12 jiradb.properties
-rw-r--r--    1 jturner  jturner     72272 Jul 28 10:02 jiradb.script
[jira-home-directory ~]$
```

Locate the hsqldb jar

The hsqldb binary is usually located in the `lib` sub-directory of your JIRA Installation Directory:

```
[jira-installation-directory ~]$ ls lib/hsql*
lib/hsqldb-1.8.0.5.jar
```

Shut down JIRA/Confluence

If you haven't already, shut down any apps using the database.

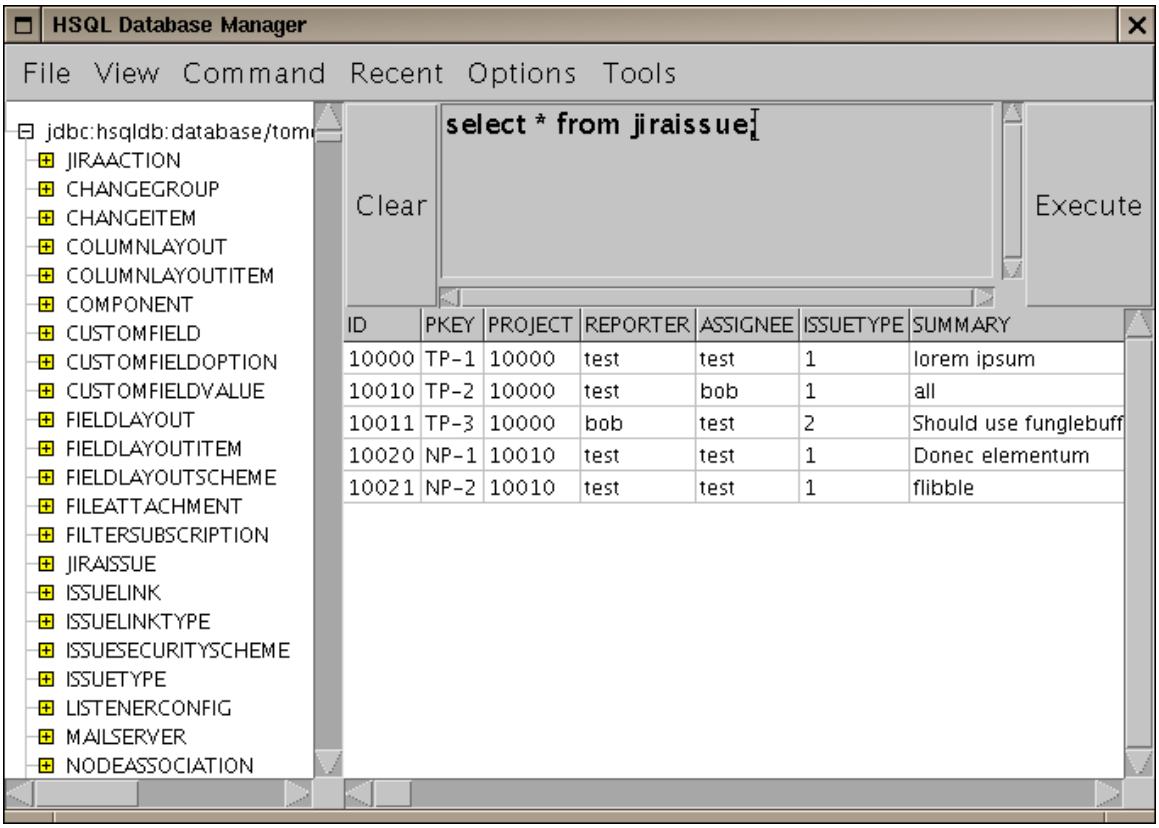
Run the console

Run the following command from the directory that contains the `database` directory (JIRA 4.0+):

```
java -cp lib/hsqldb-1.8.0.5.jar org.hsqldb.util.DatabaseManager -user sa -url jdbc:hsqldb:database/jiradb
```

In versions of JIRA before JIRA 4.1 the jar file was in `common/lib`

The hsqldb console should load, listing tables in the database in the left panel. You can run SQL commands in the top panel:



Once you have finished running SQL queries, shut down the console before starting JIRA/Confluence.

JIRA and MS SQL Server 2005

This page has general notes on connecting JIRA to SQL Server 2005. It supplements the official SQL Server 2005 installation documentation .

- Connecting to named instances in SQL Server
- Error caused by SET NOCOUNT in MS SQL Server
- MS SQL Server 2000 Startup errors
- Setting Up a SQL Server 2005 database for JIRA

Connecting to named instances in SQL Server

When using named instances you will need to specify the URL slightly differently in the connection properties.

First off, try:

```
<url>jdbc:jtds:<server_type>://<server>[:<port>][/<database>];instance=<instance_name></url>
```

This is specified at the JTDS FAQ

If this doesn't work, try dropping the instance name, and changing the port to the port used by the named instance:

```
<url>jdbc:jtds:<server_type>://<server>[:<instance_port>][/<database>]</url>
```

Note. This port is different to the normal SQL Server port as each instance listens on a different port.

Error caused by SET NOCOUNT in MS SQL Server

It is necessary to ensure that the **SET NOCOUNT** option is not set in the SQL Server configuration. For further details on how to verify these settings, please refer to the JIRA Installation documentation.

If this option is set, it can result in the following errors that can be found in the log file:

```
2006-05-03 15:51:24,331 INFO [atlassian.jira.upgrade.ConsistencyCheckImpl] Could not find Issue
Cache Listener, adding.
```

```
2006-05-03 15:51:26,088 WARN [ofbiz.core.entity.SequenceUtil]
[SequenceUtil.SequenceBank.fillBank] first select failed: trying to add row, result set was
empty for sequence: ListenerConfig

2006-05-03 15:51:26,093 WARN [ofbiz.core.entity.SequenceUtil]
[SequenceUtil.SequenceBank.fillBank] first select failed: trying to add row, result set was
empty for sequence: ListenerConfig

2006-05-03 15:51:26,094 ERROR [ofbiz.core.entity.SequenceUtil]
[SequenceUtil.SequenceBank.getNextSeqId] Fill bank failed, returning null

2006-05-03 15:51:26,112 ERROR [atlassian.jira.upgrade.ConsistencyCheckImpl] Error adding
listener: org.ofbiz.core.entity.GenericEntityException: while
inserting:
[GenericEntity:ListenerConfig][clazz,com.atlassian.jira.event.listeners.cache.IssueCacheListener][n
Cache Listener][id,null] (SQL
exception while executing the following:INSERT INTO listenerconfig (ID, CLAZZ, listenername)
VALUES (?, ?, ?) (Cannot insert the value NULL into column
'ID', table 'Jira36Test.JiraUser.listenerconfig'; column does not allow nulls. INSERT fails.))

org.ofbiz.core.entity.GenericEntityException: while inserting:
[GenericEntity:ListenerConfig][clazz,com.atlassian.jira.event.listeners.cache.IssueCacheListener][n
Cache Listener][id,null] (SQL Exception while
executing the following:INSERT INTO listenerconfig (ID, CLAZZ, listenername) VALUES (?, ?, ?)
(Cannot insert the value NULL into column 'ID', table 'Jira36Test.JiraUser.listenerconfig';
column does not allow nulls. INSERT fails.))

    at org.ofbiz.core.entity.GenericDAO.singleInsert(GenericDAO.java:123)
    at org.ofbiz.core.entity.GenericDAO.insert(GenericDAO.java:88)
    at org.ofbiz.core.entity.GenericHelperDAO.create(GenericHelperDAO.java:63)
    at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:470)
    at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:450)
    at org.ofbiz.core.entity.GenericValue.create(GenericValue.java:77)
    at com.atlassian.core.ofbiz.util.EntityUtils.createValue(EntityUtils.java:61)
    at com.atlassian.jira.action.admin.ListenerCreate.execute(ListenerCreate.java:22)
    at webwork.dispatcher.GenericDispatcher.executeAction(GenericDispatcher.java:132)
    at
com.atlassian.core.action.DefaultActionDispatcher.execute(DefaultActionDispatcher.java:34)
    at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.ensureSingleListener(ConsistencyCheckImpl.java:669)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkCacheListener(ConsistencyCheckImpl.java:563)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkDataConsistency(ConsistencyCheckImpl.java:306)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkConsistency(ConsistencyCheckImpl.java:295)
    at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.initialise(ConsistencyCheckImpl.java:164)
    at
com.atlassian.jira.upgrade.ConsistencyLauncher.contextInitialized(ConsistencyLauncher.java:27)
    at org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3692)
    at org.apache.catalina.core.StandardContext.start(StandardContext.java:4127)
    at org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:759)
    at org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:739)
    at org.apache.catalina.core.StandardHost.addChild(StandardHost.java:524)
    at org.apache.catalina.startup.HostConfig.deployDescriptor(HostConfig.java:603)
    at org.apache.catalina.startup.HostConfig.deployApps(HostConfig.java:493)
    at org.apache.catalina.startup.HostConfig.check(HostConfig.java:1195)
    at sun.reflect.GeneratedMethodAccessor341.invoke(Unknown Source)

org.ofbiz.core.entity.GenericDataSourceException: SQL Exception while executing the
following:INSERT INTO listenerconfig (ID, CLAZZ, listenername)
VALUES (?, ?, ?) (Cannot insert the value NULL into column 'ID', table
'Jira36Test.JiraUser.listenerconfig'; column does not allow nulls. INSERT fails.)
    at org.ofbiz.core.entity.jdbc.SQLProcessor.executeUpdate(SQLProcessor.java:375)
    at org.ofbiz.core.entity.GenericDAO.singleInsert(GenericDAO.java:115)
    at org.ofbiz.core.entity.GenericDAO.insert(GenericDAO.java:88)
    at org.ofbiz.core.entity.GenericHelperDAO.create(GenericHelperDAO.java:63)
    at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:470)
    at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:450)
    at org.ofbiz.core.entity.GenericValue.create(GenericValue.java:77)
    at com.atlassian.core.ofbiz.util.EntityUtils.createValue(EntityUtils.java:61)
    at com.atlassian.jira.action.admin.ListenerCreate.execute(ListenerCreate.java:22)
    at webwork.dispatcher.GenericDispatcher.executeAction(GenericDispatcher.java:132)
```

```
    at
com.atlassian.core.action.DefaultActionDispatcher.execute(DefaultActionDispatcher.java:34)
    at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.ensureSingleListener(ConsistencyCheckImpl.java:669)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkCacheListener(ConsistencyCheckImpl.java:563)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkDataConsistency(ConsistencyCheckImpl.java:306)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkConsistency(ConsistencyCheckImpl.java:295)
    at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.initialise(ConsistencyCheckImpl.java:164)
    at
com.atlassian.jira.upgrade.ConsistencyLauncher.contextInitialized(ConsistencyLauncher.java:27)
    at org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3692)
   at org.apache.catalina.core.StandardContext.start(StandardContext.java:4127)
    at org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:759)
    at org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:739)
    at org.apache.catalina.core.StandardHost.addChild(StandardHost.java:524)
    at org.apache.catalina.startup.HostConfig.deployDescriptor(HostConfig.java:603)
    at org.apache.catalina.startup.HostConfig.deployApps(HostConfig.java:493)
    at org.apache.catalina.startup.HostConfig.check(HostConfig.java:1195)
    at sun.reflect.GeneratedMethodAccessor341.invoke(Unknown Source)

java.sql.SQLException: Cannot insert the value NULL into column 'ID', table
'Jira36Test.JiraUser.listenerconfig'; column does not allow nulls. INSERT fails.
    at net.sourceforge.jtds.jdbc.SQLDiagnostic.addDiagnostic(SQLDiagnostic.java:365)
    at net.sourceforge.jtds.jdbc.TdsCore.tdsErrorToken(TdsCore.java:2781)
    at net.sourceforge.jtds.jdbc.TdsCore.nextToken(TdsCore.java:2224)
    at net.sourceforge.jtds.jdbc.TdsCore.getMoreResults(TdsCore.java:628)
    at net.sourceforge.jtds.jdbc.JtdsStatement.processResults(JtdsStatement.java:525)
    at net.sourceforge.jtds.jdbc.JtdsStatement.executeSQL(JtdsStatement.java:487)
    at
net.sourceforge.jtds.jdbc.JtdsPreparedStatement.executeUpdate(JtdsPreparedStatement.java:421)
    at
org.apache.tomcat.dbcp.dbcp.DelegatingPreparedStatement.executeUpdate(DelegatingPreparedStatement.ja
at org.ofbiz.core.entity.jdbc.SQLProcessor.executeUpdate(SQLProcessor.java:373)
    at org.ofbiz.core.entity.GenericDAO.singleInsert(GenericDAO.java:115)
    at org.ofbiz.core.entity.GenericDAO.insert(GenericDAO.java:88)
    at org.ofbiz.core.entity.GenericHelperDAO.create(GenericHelperDAO.java:63)
    at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:470)
    at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:450)
    at org.ofbiz.core.entity.GenericValue.create(GenericValue.java:77)
    at com.atlassian.core.ofbiz.util.EntityUtils.createValue(EntityUtils.java:61)
    at com.atlassian.jira.action.admin.ListenerCreate.execute(ListenerCreate.java:22)
    at webwork.dispatcher.GenericDispatcher.executeAction(GenericDispatcher.java:132)
    at
com.atlassian.core.action.DefaultActionDispatcher.execute(DefaultActionDispatcher.java:34)
    at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.ensureSingleListener(ConsistencyCheckImpl.java:669)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkCacheListener(ConsistencyCheckImpl.java:563)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkDataConsistency(ConsistencyCheckImpl.java:306)
at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.checkConsistency(ConsistencyCheckImpl.java:295)
    at
com.atlassian.jira.upgrade.ConsistencyCheckImpl.initialise(ConsistencyCheckImpl.java:164)
    at
com.atlassian.jira.upgrade.ConsistencyLauncher.contextInitialized(ConsistencyLauncher.java:27)
    at org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3692)
    at org.apache.catalina.core.StandardContext.start(StandardContext.java:4127)
    at org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:759)
    at org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:739)
    at org.apache.catalina.core.StandardHost.addChild(StandardHost.java:524)
    at org.apache.catalina.startup.HostConfig.deployDescriptor(HostConfig.java:603)
```

```
    at org.apache.catalina.startup.HostConfig.deployApps(HostConfig.java:493)
    at org.apache.catalina.startup.HostConfig.check(HostConfig.java:1195)
    at sun.reflect.GeneratedMethodAccessor341.invoke(Unknown Source)
```

---

add your notes

MS SQL Server 2000 Startup errors

Please refer to our Knowledge Base article on the topic.

A user reports getting errors like these every time JIRA starts up:

```
[core.entity.jdbc.DatabaseUtil] Entity "Action" has no table in the database
2003-11-06 09:33:45,265 ERROR [core.entity.jdbc.DatabaseUtil] Could not create table
"jiraaction"
2003-11-06 09:33:45,265 ERROR [core.entity.jdbc.DatabaseUtil] SQL Exception while executing the
following:
CREATE TABLE jiraaction (ID NUMERIC NOT NULL, issueid NUMERIC, AUTHOR VARCHAR(255), actiontype
VARCHAR(255), actionlevel VARCHAR(255), actionbody TEXT, CREATED DATETIME, actionnum NUMERIC,
CONSTRAINT PK_jiraaction PRIMARY KEY (ID))
Error was: java.sql.SQLException: There is already an object named 'jiraaction' in the
database.
2003-11-06 09:33:45,265 WARN [core.entity.jdbc.DatabaseUtil] Entity "ChangeGroup" has no table
in the database
2003-11-06 09:33:45,265 ERROR [core.entity.jdbc.DatabaseUtil] Could not create table
"changegroup"
```

JIRA functions correctly otherwise.

A solution is suggested in this jira-user post:

> *Hi Jason,*
>
> *I have had the same errors when at startup that you are seeing.*
>
> *The problem on my server was that when the user in my database ('JIRA')*
> *created tables they were created as dbo.<tablename> and not*
> *JIRA.<tablename>*
> *so when JIRA attempts to verify a table JIRA.<tablename> exists it fails.*
> *Then it tries to create <tablename>, but it already exists. All the*
> *created*
> *tables are owned by 'dbo' and not 'JIRA'.*
>
> *I am running on Microsoft SQL Server so my fix may not fit exactly but this*
> *is what I had to do:*
> *Create the 'JIRA' user as a regular user of the JIRA database. Add the*
> *JIRA user to the db_owner (database owner) role. (DO NOT change the database*
> *owner to 'JIRA', just add the role!)*
>
> *Of course, you will have to drop your existing database first.*
>
> *Cheers,*
> *Bradley.*

We have also had reports from other users that there are also alternatives to this solution. The `ddl_admin`, `db_datareader` and `db_datawriter` roles could be used instead of `db_owner` role for the `jirauser` account.

Setting Up a SQL Server 2005 database for JIRA

On this page:

- Overview
- Before you start
  - 1. Enable network connectivity for SQL Server
  - 2. Configure SQL Server with the appropriate Authentication Mode
  - 3. Disable the 'SET NOCOUNT' option in SQL Server
- Setting up the JIRA database
  - 1. Create a new database

- 2. Create a new database user
- 3. Create a JIRA database schema

Overview

This page supplements the documentation for Connecting JIRA to SQL Server 2005. It provides detailed instructions on setting up your JIRA database for a **straightforward** integration of JIRA with SQL Server 2005. Unfortunately we do not provide support for advanced database configuration, such as hardening or performance tuning. If you require a more complex solution, refer to MS SQL 2005 Documentation and, i necessary, consult with someone in your organisation who is knowledgeable in the configuration of SQL Server 2005.

Before you start

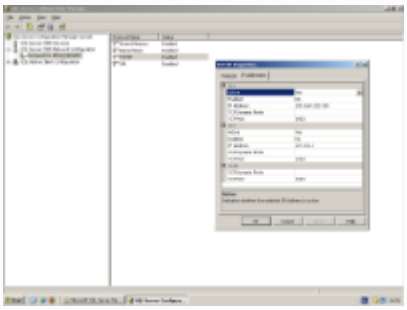
1. Enable network connectivity for SQL Server

Ensure that your instance of SQL Server allows TCP/IP connection and is listing on the default port. Please note that network connectivity is **disabled by default** in some versions of SQL Server (e.g. SQL Server 2005 Express edition). Hence, you will have to enable it, as describec below:

To enable TCP/IP for SQL Server,

1. Open the '**SQL Server Configuration Manager**'.
2. Expand '**SQL Server 2005 Network Configuration**' in the console pane.
3. Click '**Protocols for <instance name>**'.
4. The details pane will display (see screenshot below). Right-click '**TCP/IP**' and click '**Enable**'.
5. Click '**SQL Server 2005 Services**' in the console pane.
6. The details pane will display. Right-click '**SQL Server (<instance name>)**' and click '**Restart**' to stop and restart the SQL Server service.

Screenshot: Enabling TCP/IP for SQL Server 2005



2. Configure SQL Server with the appropriate Authentication Mode

Ensure that SQL Server is operating in the appropriate authentication mode. By default, SQL Server operates in '**Windows Authentication Mode**'. However, if your user is not associated with a trusted SQL connection, i.e. '`Microsoft SQL Server, Error: 18452`' is received during JIRA startup, you will need to change the authentication mode to '**Mixed Authentication Mode**'.

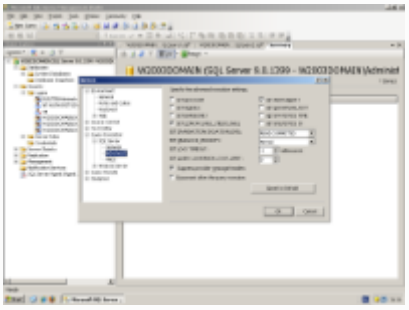
Read the Microsoft documentation on authentication modes for instructions on changing the authentication mode.

3. Disable the 'SET NOCOUNT' option in SQL Server

To disable the 'SET NOCOUNT' option in SQL Server,

1. Open the 'SQL Server Management Studio'
2. Navigate to **"Tools' -> 'Options' -> 'Query Execution' -> 'SQL Server' -> 'Advanced'**. The advanced settings for SQL Server will display.
3. Ensure that the '**SET NOCOUNT**' option is **not selected**, as per the screenshot below:

Screenshot: Disabling 'SET NOCOUNT' for SQL Server



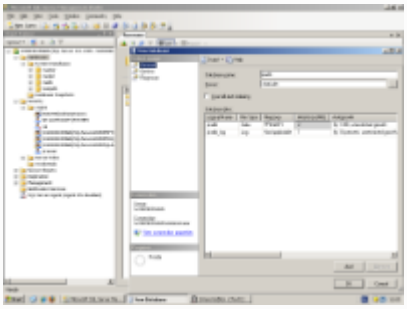
Setting up the JIRA database

To set up your JIRA database for SQL Server 2005,

1. Create a new database

    1. Open the 'SQL Server Management Studio'.
    2. Connect to the SQL Server that you want to integrate JIRA with. By default this will be '**localhost**'.
    3. Navigate to '**<your server name>**' -> '**Databases**' in the left menu of the '**SQL Server Management Studio**'.
    4. Right-click '**Databases**' under the server name of your SQL Server and select the '**New Database...**' option from the dropdown menu that appears.
    5. The '**New Database**' window will display. Select the '**General**' option in the left menu.
    6. The '**General**' page will display (see screenshot below). Enter **jiradb** in the '**Database name**' field.
    7. Select the '**Options'** option in the left menu. Check the collation type, the collation type has to be case insensitive e.g.: 'SQL_Latin1_General_CP437_CI_AI' is case insensitive. If it is using your server default, check the collation type of your server.

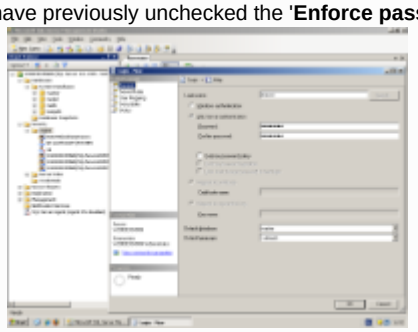
    Screenshot: Create jiradb database

    8. Click the '**OK**' button to create the database.

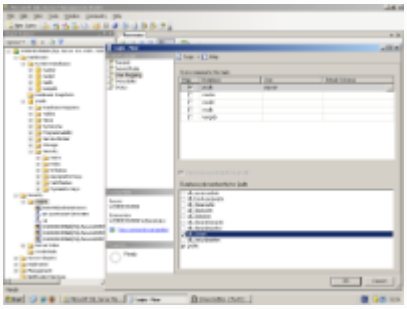
2. Create a new database user

1. Navigate to **'<your server name>' -> 'Security' -> 'Logins'** in the left menu of the 'SQL Server Management Studio'.
2. _Right-click the '**Logins**' folder and select '**New Login**'.
3. The '**Login - New**' window will display. Select the '**General**' option in the left menu.
4. Enter the database user details into the window that displays (see screenshot below), as follows:
    a. Enter '**jirauser**' in the '**Login name**' field.
    b. Select '**SQL Server authentication**'.
    c. Enter '**jirauser**' as the password, and enter '**jirauser**' again in the '**Confirm password**' field.
    d. If you wish to enforce a password policy, check the '**Enforce password policy**' checkbox. However, please be aware that you may need to modify the previously entered password ('jirauser') to meet your password policy rules (e.g. your password policy may require numeric characters in all passwords).
    e. Ensure that the '**Enforce password expiration**' checkbox is unchecked. It will be automatically unchecked and disabled, if you have previously unchecked the '**Enforce password policy**' checkbox.
    f. Ensure that the '**User must change password at next login**' checkbox is unchecked. It will be automatically unchecked and disabled, if you have previously unchecked the '**Enforce password policy**' checkbox.



Screenshot: Create jirauser user

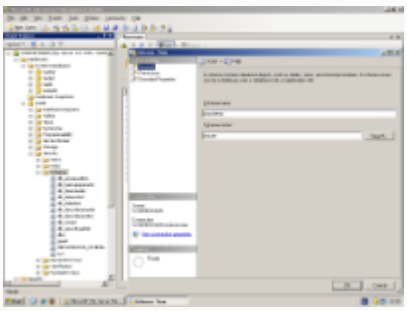
5. Select the '**User Mapping**' option in the left menu.
6. The User Mapping fields for jiradb will display (see screenshot below). Tick the '**jiradb**' checkbox.
7. The '**Database role membership for:jiradb**' panel will display in the bottom panel of the window. Tick the '**db_owner**' checkbox.
8. Click the '**OK**' button to save your changes.



Screenshot: Create user mapping for jirauser

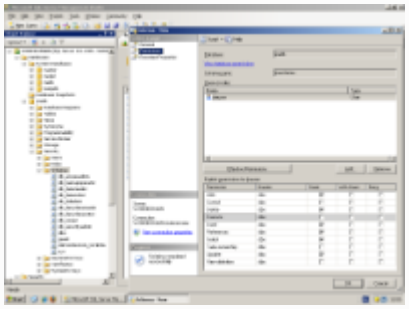
3. Create a JIRA database schema

1. Navigate to **'<your server name>' -> 'Databases' -> 'jiradb' -> 'Security' -> 'Schemas'** in the left menu of the 'SQL Server Management Studio'.
2. Right-click the '**Schemas**' folder and select '**New Schema**'
3. The 'Schema - New' window will display. Select the '**General**' option in the left menu.
4. The '**General**' page will display (see screenshot below). Fill in the fields, as follows:
    - Enter **jiraschema** in the '**Schema name**' field.
    - Enter **jirauser** in the '**Schema owner**' field.



Screenshot: Create JIRA database schema

5. Select the '**Permissions**' option in the left menu.
6. The '**Permissions**' page will display (see screenshot below). Click the '**Add...**' button.
7. Enter '**jirauser**' in the '**Enter the object names to select (examples):**' field on the pop-up window that displays. Click '**OK**' to save your update and close the pop-up window.
8. Specify the schema permissions in the '**Explicit permission for jirauser**' table on the '**Permissions**' page, as follows:
    - **Alter** — check the '**Grant**' checkbox.
    - **Delete** — check the '**Grant**' checkbox.
    - **Insert** — check the '**Grant**' checkbox.
    - **References** — check the '**Grant**' checkbox.
    - **Select** — check the '**Grant**' checkbox.
    - **Update** — check the '**Grant**' checkbox.
9. Click the '**OK**' button to save your changes.



Screenshot: Create Permissions for JIRA Schema

Congratulations, you have set up a JIRA database for SQL Server 2005. Please refer back to the Connecting JIRA to SQL Server 2005 page to continue integrating SQL Server 2005 with JIRA.

JIRA and MS SQL Server 2008

This page has general notes on connecting JIRA to SQL Server 2008. It supplements the official SQL Server 2008 installation documentation .

JIRA and MySQL

This page contains additional notes, tips, tricks and caveats on connecting JIRA to MySQL, which supplements the official MySQL installation documentation. Many of these notes are contributed by users, based on their specific experiences in connecting JIRA to MySQL.

- Configuring MySQL 5.1 to store non-ASCII characters
- JIRA Cannot Connect to MySQL with Named Pipes Enabled
- JIRA Cannot Create Issues when Using MySQL with Binary Logging
- MySQL Administrator and Data Truncation Errors
- MySQL Data Access Exception - Errcode - 17 occurs with JIRA
- Setting Up a MySQL Database on Linux for JIRA

Configuring MySQL 5.1 to store non-ASCII characters

To set up a MySQL 5.1 database with JIRA to work with non-ASCII (non-English) characters, please do the following:

1. Create a new MySQL database using the following command:
   ```
   create database jiradb default character set utf8;
   ```
2. Grant all the required permissions to the JIRA user for the database as described here.
3. Change JIRA's JDBC URL (in the `dbconfig.xml` file in your JIRA Home Directory) to use the new database and be:

```
jdbc:mysql://<your_server>:<port>/jiradb?autoReconnect=true&useUnicode=true&characterEncoding=UTF8
```

Please note the '&' XML escape for the ampersands in the url above is needed since it is specified in an xml file.

4. Start JIRA and complete the setup process.

Please ensure that you create a new database using the correct character set and ensure that JIRA creates all its tables on startup without problems. This should allow you to work with all characters supported by Unicode, which covers most characters out there.

Please ensure that you are using the latest MySQL JDBC driver (see Connecting JIRA to MySQL for information on the JDBC driver).

Also please ensure you are using the UTF-8 character encoding in JIRA (Administration -> Global Settings -> General Configuration).

JIRA Cannot Connect to MySQL with Named Pipes Enabled

JIRA can't connect to the database with Named Pipes enabled

```
I've tried a number of things, and it looks like named pipes is the problem. This is a problem
with MySQL, not with JIRA. Essentially I've had to install MySQL with two key things:

    * Go through the Standard Installation route for MySQL, not the Detailed Installation route
    * Enable TCP/IP connections in the MySQL Config Wizard afterwards

After doing this, JIRA now appears to connect to the MySQL and can see the new database
```

Details of what I did to recover MySQL after installing it using named pipes:

MySQL Installation and Config:

- Install MySQL server 5.0 release 5.0.24a (see http://dev.mysql.com/doc/refman/5.0/en/windows-installation.html and http://dev.mysql.com/tech-resources/articles/securing_mysql_windows.html)
- Select typical install
- Configure MySQL with the Configuration Wizard
  o Detailed Configuration
  o Server Machine
  o Multifunctional Database
  o Chose C:\<Installation path> for the InnoDB tablespace
  o Decision Support DSS/OLAP
  o Disable TCP/IP networking for security and Enable Strict Mode
  o UTF-8 character set
  o Install as Windows Service
  o Include MySQL /bin directory on path - allows mysql* commands to be run directly
  o Choose root password: ********** Do not allow access from remote machines
- Execute configuration
  o Config OK - my.ini
  o Service started - mySQL
  o Security setting FAILED - error 2017. Can't open named pipe to host: .pipe:mysql(2)
- Create a my.cnf with (client) host=localhost in it, as per http://mysqld.active-venture.com/Windows_vs_Unix.html. Also edit my.ini to have the same line under (client). Don't know if this will work. Named pipes may be a problem.
- Hmm.. There's a suggestion that the Detailed Configuration method just doesn't work (http://forums.mysql.com/read.php?11,80814,93616). If I can't get JIRA to connect to MySQL it might be necessary to re-install the whole thing...
- Create MySQL database and user for JIRA to use. In a command shell run:
  o mysql -u root
  o CREATE DATABASE jiradb CHARACTER SET 'utf8';
  o show databases;
  o CREATE USER 'jirauser'@'localhost' IDENTIFIED BY '****'; (where **** is jirauser's password)
  o GRANT ALL PRIVILEGES ON jiradb.* TO 'jirauser'@'localhost';
  o quit;
  o mysql -u jirauser -p
  o <enter password>
  o show databases;
  o jiradb is listed as one of the databases
  o quit;

JIRA configuration to use MySQL:

- Download JDBC driver mysql-connector-java-3.1.12.zip
- Copy the mysql-connector-java-3.1.12-bin.jar file from this zip to C:\Jira\atlassian-jira-professional-3.6.5-standalone\common\lib
- Edit the conf/server.xml file
  o username and password for the jirauser account set up above
  o driverClassName="com.mysql.jdbc.Driver"
  o url="jdbc:mysql://localhost/jiradb?autoReconnect=true&useUnicode=true&characterEncoding=UTF8"
  o delete the minEvictableIdleTimeMillis and timeBetweenEvictionRunsMillis parameters
- Edit the entityengine.xml file and change the field-type-name to mysql

Re-start JIRA to use MySQL database

- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\shutdown
  o Tomcat web-server shuts down
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup
  o Get error on connection: Unable to establish connection with the database. I suspect this is because the database wasn't set up correctly above and can't open named pipes. This is probably the issue with WinNT-based systems not being able to support named pipes (without modification).
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\shutdown

Re-installing MySQL without named pipes

- Go to Control Panel->Add/Remove Programs and remove MySQL
- Delete C:\Program Files\MySQL
- Reinstall as above (typical installation)
- Configure and select "Standard Installation"
  o Install as Windows Service
  o Add \bin to path
  o Choose root password **********
- Execute configuration.
  o Success!
- Run MySQL Config Wizard. Choose options as above.
- Execute configuration
  o Success!
- Set up MySQL database and jira user as above.
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup
  o FAILED! Tomcat starts and shuts down immediately. Looking at the logs, it seems that the jirauser account has a & in the password, which stuffs the XML.
- log back in to MySQL as root and run:
  o DROP USER 'jirauser'@'localhost';
  o CREATE USER 'jirauser'@'localhost' IDENTIFIED BY '********'; (makeing sure password has no & in it)
  o GRANT ALL PRIVILEGES ON jiradb.* TO 'jirauser'@'localhost';
  o quit;
- Edit conf/server.xml to the new password
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup
  o No good. Still won't connect.
- Try re-running MySQL config and this time enable TCP/IP connection over port 3306 (so that we don't have to use named pipes)
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup
  o Tomcat server starts! Hurray. We appear to connect in some way, although there are lots of exceptions. Maybe these are due to the first start?
- Point web browser at http://localhost:8080/
  o JIRA config screen appears - good
- Go through the JIRA setup and initial configuration steps as above

---

http://confluence.atlassian.com/pages/editpage.action?pageId=133186

JIRA Cannot Create Issues when Using MySQL with Binary Logging

If you use JIRA with MySQL and attempt to create a JIRA issue, JIRA may generate an error similar to the following:

```
Error creating issue: Could not create workflow instance: root cause: while inserting:
[GenericEntity:OSWorkflowEntry][id,null][name,jira][state,0]
(SQL Exception while executing the following:INSERT INTO OS_WFENTRY (ID, NAME, INITIALIZED,
STATE) VALUES (?, ?, ?, ?)
(Binary logging not possible. Message: Transaction level 'READ-COMMITTED' in InnoDB is not safe
for binlog mode 'STATEMENT'))
```

OR

```
Binary logging not possible. Message: Transaction level 'READ-COMMITTED' in InnoDB is not safe
for binlog mode 'STATEMENT'
```

You may encounter this problem if your JIRA MySQL database configuration:

- Makes use of the InnoDB database storage engine (which is recommended)
  AND
- Uses MySQL's default binary logging format

JIRA uses the 'READ-COMMITTED' transaction isolation level with MySQL, which currently only supports row-based binary logging. For more information about this, please refer to MySQL issue no. 40360.

To overcome this problem, you must configure MySQL's binary logging format to use 'row-based' binary logging.

To do this:

1. Shutdown JIRA and your MySQL service if necessary.
2. Open the MySQL configuration file (`my.cnf`) in a text editor.
   On UNIX-based systems, this file may be located in the `/etc` directory.
3. Locate the `binlog_format` property in this file in the `[mysqld]` section and ensure that its value is `row`, such that you end up with:

   ```
   binlog_format=row
   ```

   This is only needed (and valid) for MySQL versions 5.1.5 and later.
4. Save your changes to this file and restart your MySQL service and JIRA.

MySQL Administrator and Data Truncation Errors

Hi All,

Due to the release of the 3.7 branch requiring an empty database on startup (see here), a lot of our customers have had issues importing their data into the new install due to encoding inconsistencies between their existing databases and the new ones they've created for 3.7.x.

Errors that users are seeing are Data Truncation errors that look like:

```
org.ofbiz.core.entity.GenericDataSourceException: SQL Exception while executing the
following:INSERT INTO jiraaction (ID, issueid, AUTHOR, actiontype,
actionlevel, rolelevel, actionbody, CREATED, actionnum) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)
(Data truncation: Data too long for column 'actionbody' at row 1)\
 at org.ofbiz.core.entity.jdbc.SQLProcessor.executeUpdate(SQLProcessor.java:375)
```

We've been told that users using 'MYSQL Administrator' to create their databases lack the ability to specify what encoding type they wish their database to use. Here is the comment we received from a customer:

> *I solved the problem. I used to create the database using the MySQL Adminstrator with the effect, that I could not define the character-set to use for the database. So I tried to create the database manually using the command-line tool and bang, the import of the data worked fine.*

I hope this helps and saves you all some installation hiccups.

-Michelle

MySQL Data Access Exception - Errcode - 17 occurs with JIRA

A user reports of getting this error caused by a MySQL Bug:

```
com.atlassian.jira.exception.DataAccessException: java.sql.SQLException: Can't create/write to
file 'C:\temp2#sql_eb4_0.MYI' (Errcode: 17)
at com.atlassian.jira.upgrade.util.UpgradeUtils.getExactColumnName(UpgradeUtils.java:222)
at
com.atlassian.jira.appconsistency.db.Build178SchemaCheck.isColumnInTable(Build178SchemaCheck.java:8
com.atlassian.jira.appconsistency.db.Build178SchemaCheck.check(Build178SchemaCheck.java:71)
at com.atlassian.jira.appconsistency.db.Build178SchemaCheck.isOk(Build178SchemaCheck.java:38)
at com.atlassian.jira.appconsistency.db.DatabaseChecker.checkDatabase(DatabaseChecker.java:108)
at
com.atlassian.jira.appconsistency.db.DatabaseCompatibilityEnforcer.contextInitialized(DatabaseCompa
org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3692)
```

where the error code means:

```
C:\>perror 17
OS error code 17: File exists
```

The Workaround:
Disabling their virus checker seemed to resolve the issue of JIRA not coming up. Users should therefore not run "on-access" checking on their JIRA servers.

Bug Details:
The bug is described in more detail on the following link MySQL Bug Forum.

Setting Up a MySQL Database on Linux for JIRA

The latest official documentation on configuring JIRA with MySQL can be found in the Connecting JIRA to MySQL guide.

This is a step-by-step supplement guide for setting up your MySQL database for JIRA. Although this guide assumes that your MySQL database server is running on Linux, the various procedures described below can be adapted (or may be applicable) to other similar operating systems.

Enable MySQL TCP/IP networking

Some Linux distributions (eg. Debian) disable MySQL's TCP/IP networking as a security precaution. You can test that MySQL is listening on the default port (3306) as follows:

```
jturner@teacup:~$ netstat -na | grep 3306
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:48211         127.0.0.1:3306          TIME_WAIT
tcp6       1      0 ::ffff:127.0.0.1:34785  ::ffff:127.0.0.1:3306   CLOSE_WAIT
```

Or if `netstat` isn't available:

```
jturner@teacup:~$ telnet localhost 3306
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
D
5.0.13-rc-Debian_1-lo!X{$:;V#H!ju (press ctrl-] here)
telnet> quit
Connection closed.
```

On Debian, you can enable MySQL TCP connections by editing `/etc/my.cnf`, commenting out the 'skip-networking' flag, and restarting mysqld.

Create MySQL database and user

Create a MySQL user called 'jirauser' and database called 'jiradb':

```
jturner@teacup:~$ mysql --user=root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 559 to server version: 5.0.13-rc-Debian_1-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database jiradb character set utf8;
Query OK, 1 row affected (0.02 sec)

mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,ALTER,INDEX on
  jiradb.* TO 'jirauser'@'localhost' IDENTIFIED BY 'mypassword';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye
```

The 'IDENTIFIED BY' phrase sets the password for the user (in this case, 'mypassword'). Your hostname may be different; you will find out in the next steps.

Now verify that user 'jirauser' can connect:

```
jturner@teacup:~$ mysql \--user=jirauser \--password=mypassword \--database=jiradb
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 565 to server version: 5.0.13-rc-Debian_1-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

If you get errors like:

```
Access denied for user 'jirauser'@'localhost' (using password: YES)
```

You will need to adjust the 'host' field for the JIRA user record:

```
jturner@teacup:~$ mysql --user=root -p mysql
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 655 to server version: 5.0.13-rc-Debian_1-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> select user, host from user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| debian-sys-maint | localhost |
| jirauser         | localhost |
| root             | localhost |
| root             | teacup    |
+------------------+-----------+
4 rows in set (0.00 sec)

mysql> update user set host='localhost.localdomain' where user='jirauser';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> flush privileges;
Query OK, 0 rows affected (0.03 sec)
```

See also Atlassian's MySQL Tips.

If problems persist, see the MySQL Causes of Access Denied Errors page.

For more general information, see Adding New User Accounts to MySQL.

Start JIRA

Check for errors in the log files.

Again, if you see an 'Access denied' error:

```
Access denied for user 'jirauser'@'localhost.localdomain' (using password: YES)
```

Then you need to adjust your /etc/hosts so that 'localhost' comes before 'localhost.localdomain', and restart MySQL. This is a MySQL bug fixed in 5.0.11.

Run the Setup Wizard

Point a browser at http://localhost:8080/, and set up JIRA as described in the Setup Wizard.

Troubleshooting

Q: I get the following error message in MySQL, "`Attempted reconnect 3 times. Giving up.`" What should I do?
A: *MySQL error message*

```
jdbc:mysql://localhost/test?autoReconnect=true connection error:
Server connection failure during transaction.
Attempted reconnect 3 times. Giving up.
```

To troubleshoot your MySQL connection, please follow the steps below:

1. Enter the following command to connect to MySQL:

   ```
   mysql -p -u [dbuser] -h 127.0.0.1 [dbname]
   ```

   For example,

   ```
   mysql -p -u mydbuser -h 127.0.0.1 test
   ```

2. If you cannot connect to MySQL after entering your password, login to your mysql with the root account:

   ```
   mysql -p -u root
   ```

   And enter following command:

   ```
   mysql> GRANT ALL PRIVILEGES ON <dbname>.* to <user>@127.0.0.1 identified by '<password>';
   mysql> FLUSH PRIVILEGES;
   ```

   where,
   `<dbname>` is your database name,
   `<user>` is your database user name,
   `<password>` is your database password.
   Do not forget the last command: '`FLUSH PRIVILEGES`'

3. If you still cannot connect, please check that your MySQL is listening on the default port of `3306` and bind in your IP, `127.0.0.1` by running either of the following commands:

   ```
   netstat -a |grep mysql
   ```

   or,

   ```
   netstat -a |grep 3306
   ```

   If MySQL is listening, you should see the following message:

   ```
   tcp 0 0 *:mysql *:* LISTEN
   ```

   Alternatively, you also could check if your MySQL is listening on the default port by running this command:

   ```
   telnet 127.0.0.1 3306
   ```

JIRA and Oracle

This page has general notes on connecting JIRA to Oracle. It supplements the official Oracle installation documentation.

- Configuring Datasource for Oracle 10g JDBC drivers
- Restoring data using I-Net (Oranxo) Driver for Oracle
- Store Workflow on Disk with Oracle 8

Configuring Datasource for Oracle 10g JDBC drivers

When using JIRA with Oracle, the Oracle 10g JDBC driver needs to have the **SetBigStringTryClob** property set to true to store text of unlimited size in the database. If this property is not set, you will have problems modifying JIRA workflows and storing large (over 32k) text strings.

The **SetBigStringTryClob** property needs to be set in the application server, where the database connection is defined (the 'datasource' definition). The definition depends on the application server that you are using. Please refer to one of the sections below that is applicable to your application server to determine what to add to the datasource definition.

The same thing applies to I-Net's JDBC driver, except the property is called **streamstolob**.

Refer to the Connecting JIRA to Oracle documentation for details on how to specify this property in JIRA's `dbconfig.xml` file.

Resolving Connection Closure Issues

If you experiencing problems with connections closing, you may be able to resolve them by configuring the Commons Database Connection Pool (DBCP) to detect broken connections and create replacement connections. Please read Surviving Connection Closures for instructions on how to do this.

Restoring data using I-Net (Oranxo) Driver for Oracle

When restoring data into an Oracle 9 database using the I-Net Oranxo Driver a user was seeing this error message in their logs:

```
org.ofbiz.core.entity.GenericDataSourceException: SQL Exception while executing the
following:INSERT INTO jiraaction (ID, issueid, AUTHOR, actiontype, actionlevel, actionbody,
CREATED, actionnum) VALUES (?, ?, ?, ?, ?, ?, ?, ?) ([OraDriver] #7 Unexpected end of
inputstream in header.)
 at org.ofbiz.core.entity.jdbc.SQLProcessor.executeUpdate(SQLProcessor.java:375)
 at org.ofbiz.core.entity.GenericDAO.singleInsert(GenericDAO.java:115)
 at org.ofbiz.core.entity.GenericDAO.insert(GenericDAO.java:88)
 at org.ofbiz.core.entity.GenericHelperDAO.create(GenericHelperDAO.java:63)
 at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:470)
 at org.ofbiz.core.entity.GenericDelegator.create(GenericDelegator.java:450)
 at org.ofbiz.core.entity.GenericValue.create(GenericValue.java:77)
 at com.atlassian.jira.action.admin.ImportParser$1.run(ImportParser.java:191)
 at EDU.oswego.cs.dl.util.concurrent.PooledExecutor$Worker.run(PooledExecutor.java:751)
 at java.lang.Thread.run(Thread.java:595)
com.inet.ora.Ora3SQLException: [OraDriver] #7 Unexpected end of inputstream in header.
 at com.inet.ora.Ora3Factory.createSQLException(Unknown Source)
 at com.inet.ora.Ora3Factory.createSQLException(Unknown Source)
 at com.inet.ora.OraConnection.c(Unknown Source)
 at com.inet.ora.OraConnection.a(Unknown Source)
 at com.inet.ora.OraPreparedStatement.a(Unknown Source)
 at com.inet.ora.OraPreparedStatement.a(Unknown Source)
 at com.inet.ora.OraPreparedStatement.executeUpdate(Unknown Source)
```

Fix

This error was fixed by changing the **Set Clob** entry in JIRA's `dbconfig.xml` in the JIRA Home Directory.

So instead of using:

```
<connection-properties>SetBigStringTryClob=true</connection-properties>
```

for this driver it needs to be replaced with:

```
<connection-properties>streamstolob=true</connection-properties>
```

See the appropriate section in the Oracle JDBC Manual for details on these connection properties and the Connecting JIRA to Oracle for more information about configuring your `dbconfig.xml` file.

Store Workflow on Disk with Oracle 8

Atlassian support for Oracle 8 officially ends with the JIRA 3.6 release. Oracle 8 users are advised to upgrade to avoid the problem described on this page.

Applies to:	Oracle 8 users

Oracle has a 4000 character limitation on its VARCHAR2 field type. This causes problems for JIRA, which uses VARCHAR2 to store comments and 'workflows'. Whenever a comment or workflow exceeds 4000 characters (very easy in the case of workflows), JIRA breaks.

Oracle have a workaround for this problem in their 10g JDBC driver, which can be used with Oracle 9 and 10. Use of this workaround has been incorporated into the documentation.

This doesn't help Oracle 8 users.A workaround for the problem of > 4000 character workflows in Oracle 8 is to store these on disk, instead of in the database.This can be done as follows:

1. Run JIRA (with hsqldb database) to construct the workflow, and then:
2. Export the created workflow as XML, and save this to disk, eg custom-workflow.xml
3. In the JIRA instance that will use Oracle, edit WEB-INF/classes/workflows.xml and add a line:

```
<workflow name="custom" type="resource" location="custom-workflow.xml"/>
```

   Where 'custom' is the workflow name.

4. Copy custom-workflow.xml to WEB-INF/classes/
5. Restart JIRA. The 'custom' workflow should appear in the list of available workflows.

JIRA and PostgreSQL

This page has general notes on connecting JIRA to Postgres. It supplements the official Postgres installation documentation.

- Setting up a PostgreSQL Database on Linux for JIRA

Setting up a PostgreSQL Database on Linux for JIRA

The latest official documentation on configuring JIRA with PostgreSQL can be found in the Connecting JIRA to PostgreSQL guide.

This is a step-by-step supplement guide for setting up your PostgreSQL database for JIRA on Ubuntu. However, the various procedures described below can be adapted (or may be applicable) to other Linux distributions too.

Set Up a PostgreSQL User

PostgreSQL is very easy to set up on Ubuntu:

```
user:~$ sudo apt-get install postgresql-8.2 postgresql-client-8.2
Reading package lists... Done
Building dependency tree
....
 * Starting PostgreSQL 8.2 database server             [ OK ]
```

Now we create a `jira` PostgreSQL user for the user account that runs JIRA to connect as:

```
user:~$ sudo su - postgres
postgres:~$ createuser -P jira
Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) y
Shall the new role be allowed to create more new roles? (y/n) n
CREATE ROLE
```

Set Up a PostgreSQL user

We can now connect as our `jira` user and create a database.

```
postgres:~$ logout
user:~$ sudo su - jira
jira:~$ createdb jiradb
CREATE DATABASE
```

Start JIRA

Run the Setup Wizard

Point a browser at http://localhost:8080/, and set up JIRA as described in the Setup Wizard.

How to Set Up SMTP Relay in Exchange 2007

There are a few known issues setting up a proper SMTP relay for our Exchange 2007 environment. The JIRA install was originally relaying off a different SMTP service before it was moved to Exchange 2007 server. This is occurring because SMTP Relaying not configured in Exchange 2007. The issue that has been encountered thus far are:

Depending on the configuration for the Receive Connector (i.e. SMTP relay) in Exchange 2007 will encounter one of the following problems:

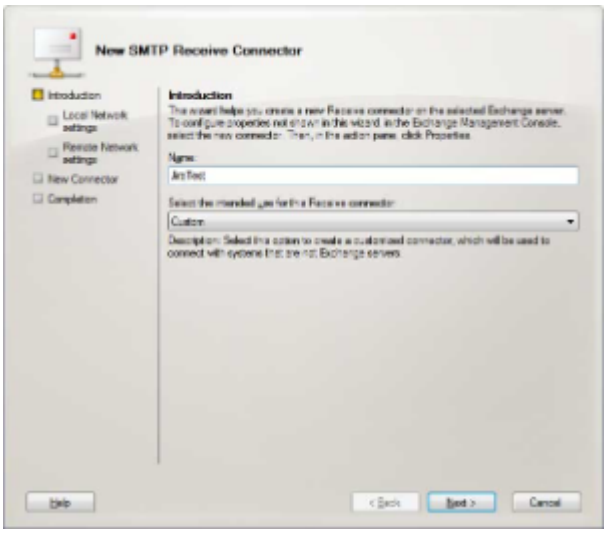
1. Emails would relay to outside domains but the user's name (that created or commented on the task) would be stripped off the `From` address in the email leaving only the "Jira-Replies" address.
2. Email will not relay to outside domains but the user's name would stay intact in the `From` address of the email.

Neither solution is optimal. While it is possible to set up an anonymous SMTP relay in Exchange 2007, for it to work properly there is one step that needs to be completed outside of the MMC. That command is listed below after the step by step instructions below.

Step by step instructions for setting up an SMTP relay in Exchange 2007 for JIRA.
Open up the Exchange MMC and select Hub Transport under Server Configuration on the left side. Split into two horizontal windows, it makes it easier to see a list of configured servers with the Hub Transport role at the top, and Receive Connectors at the bottom. Select whichever Hub Transport server is needed want this relay for and select the "New Receive Connector" in the action area on the right of the MMC. This should open the "New SMTP Receive Connector" wizard.

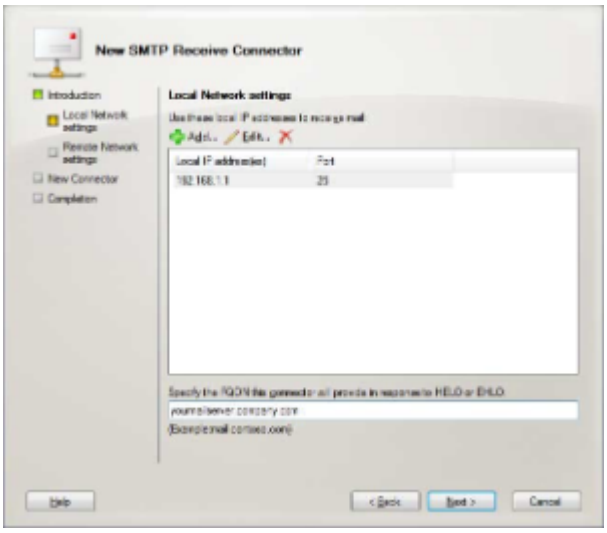
Thanks to Michael Athey for writing, documenting and providing all content for this knowledge base article.

New SMTP Receive Connector wizard



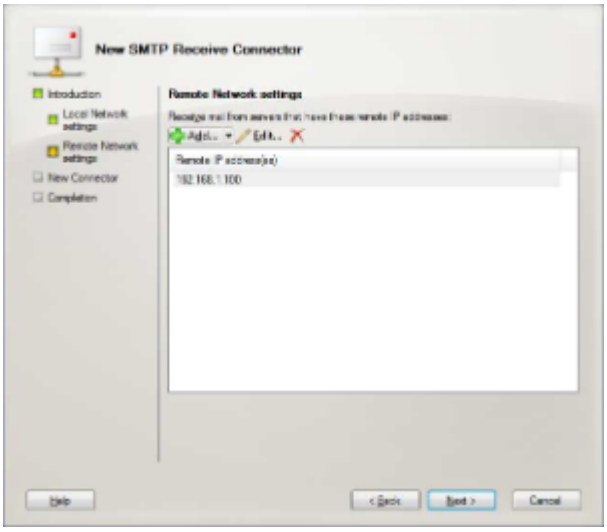
Main windows

1. Give the new relay a name (this example uses JiraTest as the name)
2. Make sure the drop down selection is set to "Custom" for "Select the intended use for this Receive connector"
3. Hit Next



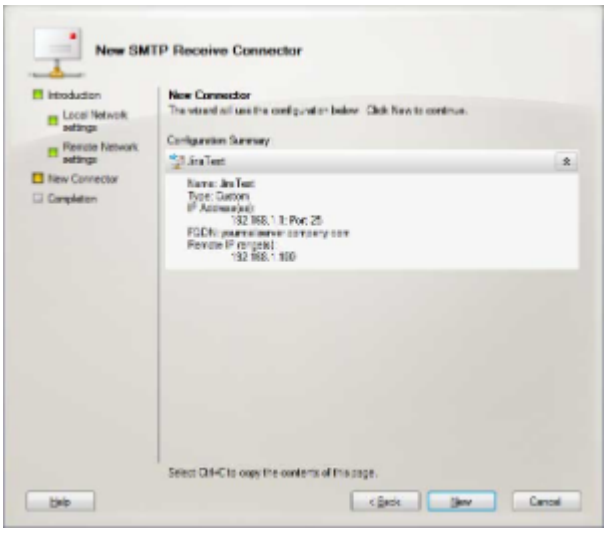
Local Network settings window

1. Select "Add", then choose "Specify an IP address"
2. Enter the IP address of the Exchange Hub Transport server
3. Keep the Port setting at 25
4. Hit OK
5. Delete the default "All available IPv4 addresses"
6. Enter the FQDN for the Exchange Hub Transport server
7. Hit Next



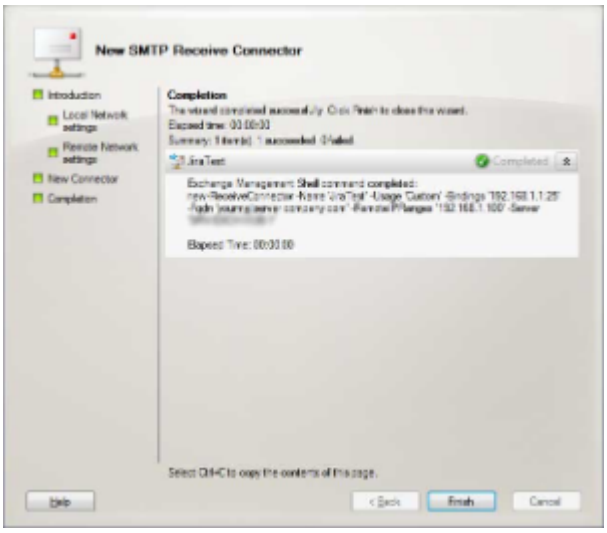
Remote Network settings window

1. Select "Add"
2. Enter the IP address of the JIRA server
3. Hit OK
4. Delete the default 0.0.0.0-255.255.255.255 range
5. Hit Next



New Connector window

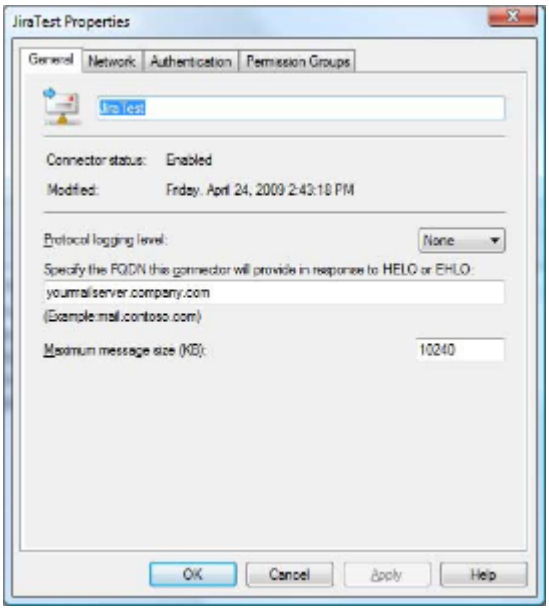
1. It should now be possible to see an overview of the connector being creating
2. Hit New



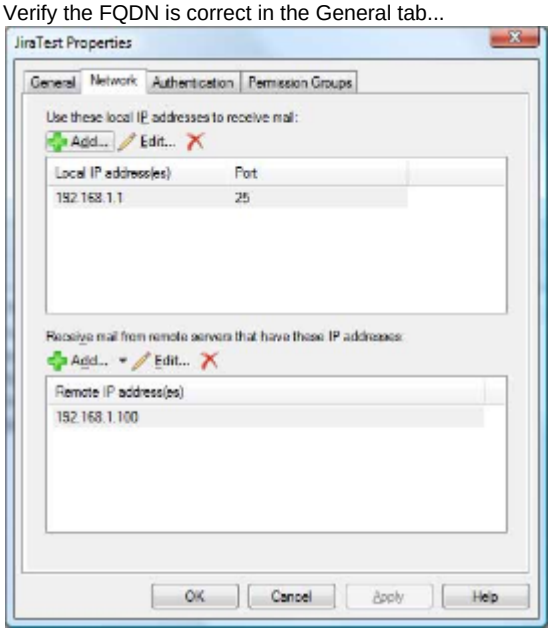
Completion window

1. Hopefully this shows the connector completed successfully
2. Hit Finish

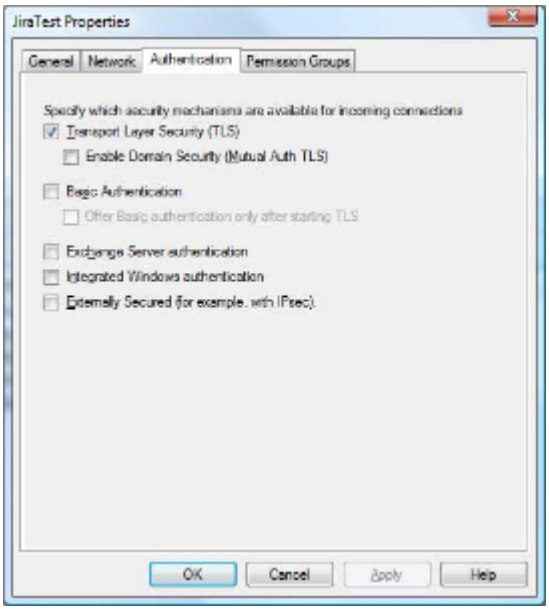
It should now be possible to should now see the connector listed on the bottom half of the screen with any others that may have created previously.
Double click the one that was just created to open its properties.



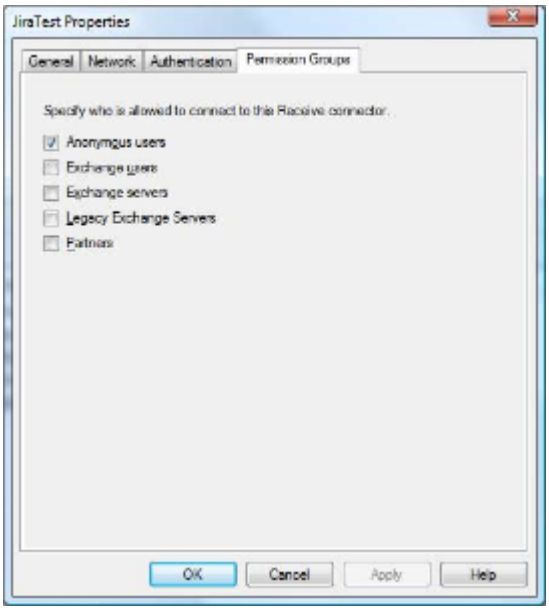
Verify the FQDN is correct in the General tab...



and that the IP addresses are also correct in the Network tab.

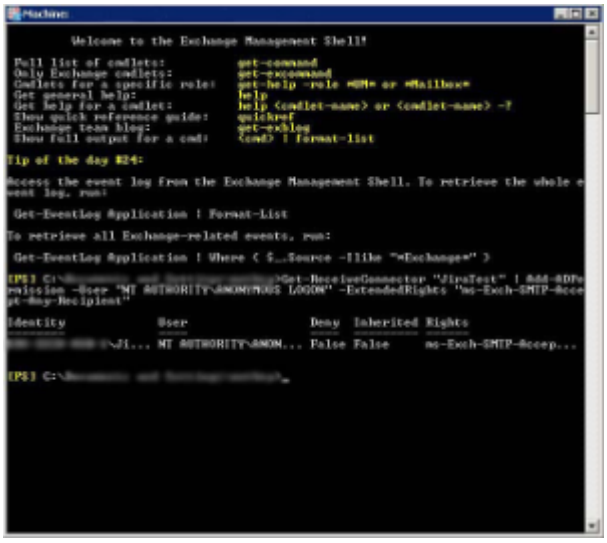


The Authentication tab should only have Transport Layer Security (TLS) selected only.



- Now go to the Permission Groups tab and select Anonymous Users, unselecting any other options, then hit OK.
- Normally that should be it, but it's not. There is one more additional step that which needs to be done in PowerShell.
- Open up the Exchange Management Shell and type the following, where "JiraTest" is the name of the connector that was just created.

```
Get-ReceiveConnector "JiraTest" | Add-ADPermission -User "NT AUTHORITY\ANONYMOUS LOGON"
-ExtendedRights "ms-Exch-SMTP-Accept-Any-Recipient"
```



What this command does is grant the relay permission to the Anonymous group for that connector. When simply selecting Anonymous Users through the GUI all that is assigned is the most common permissions, but it does not grant the relay permission. So now run the command through Management Shell.
After that is completed it is possible to start relaying JIRA's email through the Exchange 2007 setup.

Installation Troubleshooting Guide

This troubleshooting guide lists some of the common installation problems people run into

> If you have a question that is not answered here, please see our support page for information on how to seek help.

Issues

- My JIRA instance starts up with strange errors, what could be wrong?

My JIRA instance starts up with strange errors, what could be wrong?

If you're using the Windows XP, you may have extracted JIRA with the built-in unzip tool. This built-in unzip tool is broken - it silently fails to extract files with long names (see JRA-2153). Other users have also reported problems using WinRAR. Please use another tool like 7-zip or WinZIP to unpack JIRA.

If you're using Solaris, it also suffers from similar problems. You will need to use GNU tar to handle the long filenames.

Other users have reported similar problems using Midnight Commander.

Installing a LDAP server on Debian Linux for use with JIRA

This page contains an example of how to install LDAP on Linux. It's assumed that you are working towards LDAP authentication in JIRA or Confluence.

Install LDAP

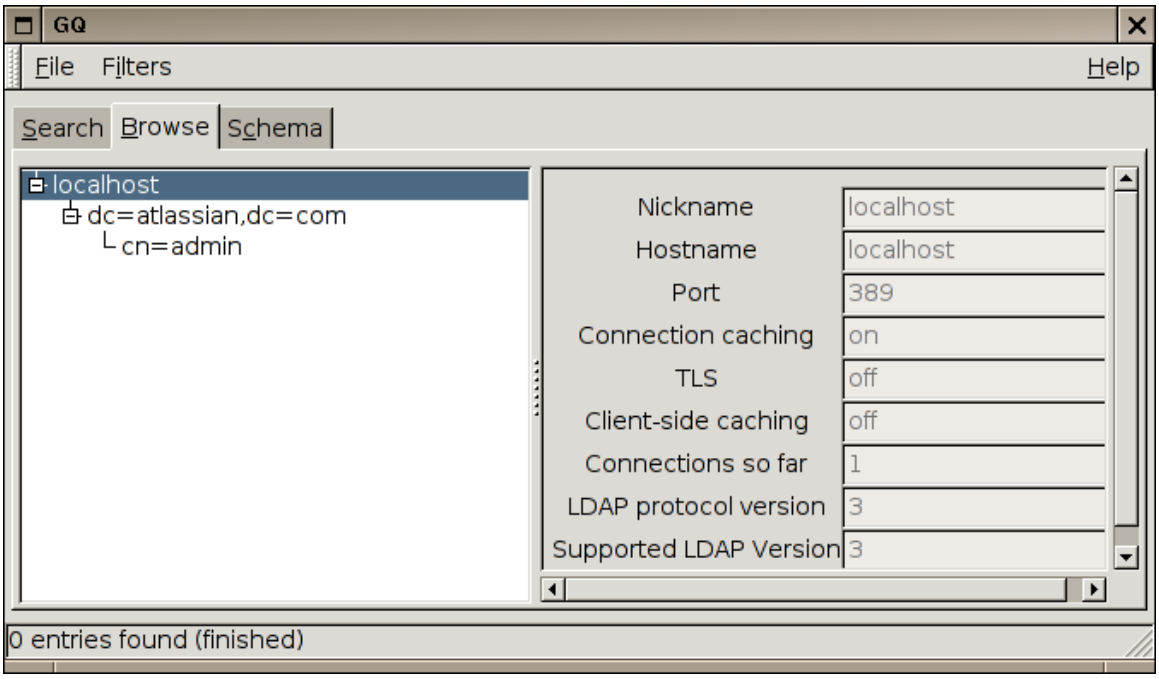
On Debian, an LDAP server can be installed with:

```
apt-get install slapd ldap-utils
```

Entering the following details when prompted (customize for your organization):

Domain name	atlassian.com
Organization name	Atlassian
Admin password	secret
LDAP v2 protocol	no

At this point, you might as well install a graphical LDAP browser, like 'gq'. Connecting anonymously, you'll see there is one entry, `cn=admin,dc=atlassian,dc=com`, created.



Create a schema

Rather than try to devise my own LDAP schema, I used the 'migrationtools' package to create a schema, and import system users from `/etc/passwd`:

```
apt-get install migrationtools
```

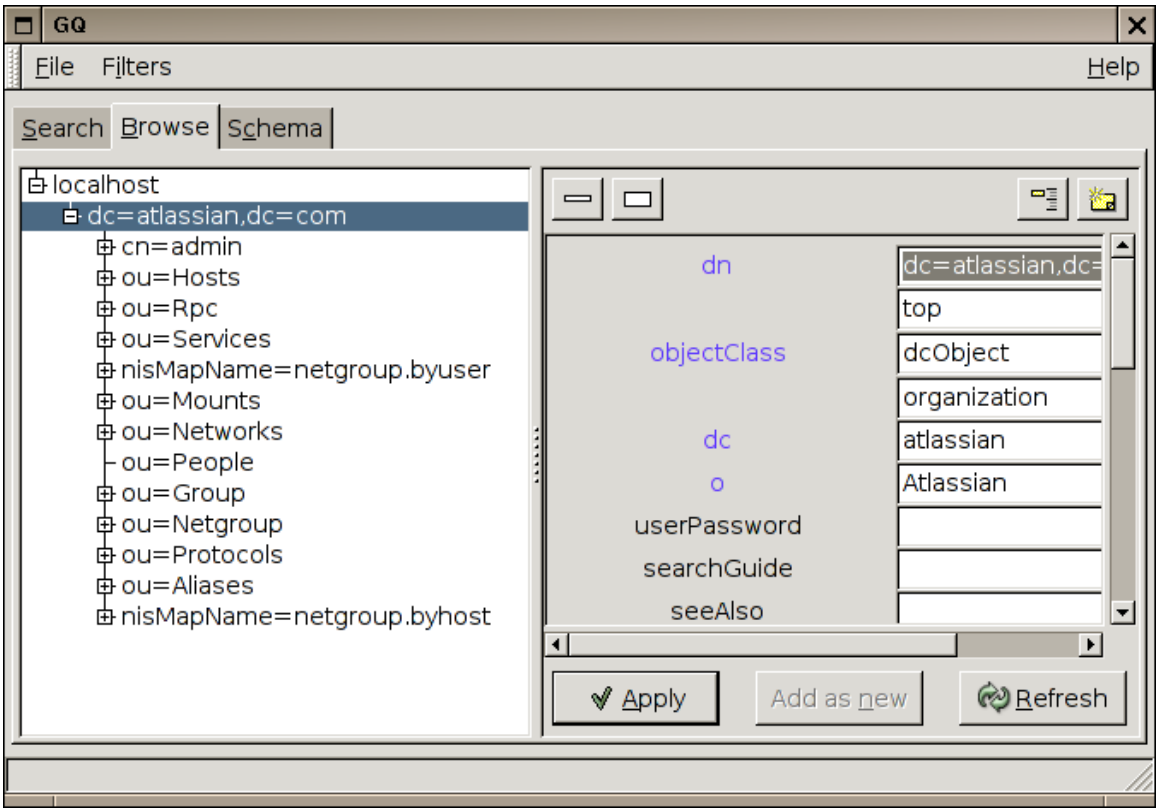
1. Edit `/etc/migrationtools/migrate_common.ph`, and make the following changes:

```
@@ -68,10 +68,10 @@
}

    1. Default DNS domain
       -$DEFAULT_MAIL_DOMAIN = "padl.com";
       +$DEFAULT_MAIL_DOMAIN = "atlassian.com";

    1. Default base
       -$DEFAULT_BASE = "dc=padl,dc=com";
       +$DEFAULT_BASE = "dc=atlassian,dc=com";

    1. Turn this on for inetLocalMailReceipient
    2. sendmail support; add the following to
       @@ -93,8 +93,8 @@
       #$USE_UTF8 = 1;

    1. Uncomment these to avoid Debian managed system users and groups
       -#$IGNORE_UID_BELOW = 1000;
       -#$IGNORE_GID_BELOW = 100;
       +$IGNORE_UID_BELOW = 1000;
       +$IGNORE_GID_BELOW = 100;

    1. And here's the opposite for completeness
       #$IGNORE_UID_ABOVE = 9999;
```

1. Run /usr/share/migrationtools/migrate_all_online

```
teacup:/usr/share/migrationtools# ./migrate_all_online.sh
Enter the X.500 naming context you wish to import into: [dc=padl,dc=com] dc=atlassian,dc=com
Enter the hostname of your LDAP server Connecting to an LDAP Directory: localhost
Enter the manager DN: [cn=admin,dc=atlassian,dc=com]:
Enter the credentials to bind with:
Do you wish to generate a DUAConfigProfile [yes|no]? no

Importing into dc=atlassian,dc=com...

Creating naming context entries...
Migrating aliases...
Migrating groups...
Migrating hosts...
Migrating networks...
Migrating users...
Migrating protocols...
Migrating rpcs...
Migrating services...
Migrating netgroups...
Migrating netgroups (by user)...
Migrating netgroups (by host)...
Importing into LDAP...
adding new entry "ou=Hosts,dc=atlassian,dc=com"

adding new entry "ou=Rpc,dc=atlassian,dc=com"

adding new entry "ou=Services,dc=atlassian,dc=com"

adding new entry "nisMapName=netgroup.byuser,dc=atlassian,dc=com"

adding new entry "ou=Mounts,dc=atlassian,dc=com"

adding new entry "ou=Networks,dc=atlassian,dc=com"

adding new entry "ou=People,dc=atlassian,dc=com"

adding new entry "ou=Group,dc=atlassian,dc=com"

adding new entry "ou=Netgroup,dc=atlassian,dc=com"

adding new entry "ou=Protocols,dc=atlassian,dc=com"

adding new entry "ou=Aliases,dc=atlassian,dc=com"

adding new entry "nisMapName=netgroup.byhost,dc=atlassian,dc=com"

adding new entry "cn=postmaster,ou=Aliases,dc=atlassian,dc=com"
ldapadd: update failed: cn=postmaster,ou=Aliases,dc=atlassian,dc=com
ldap_add: Undefined attribute type (17)
additional info: rfc822MailMember: attribute type undefined

/usr/bin/ldapadd: returned non-zero exit status
```

At this point, you should be able to browse the updated schema in a LDAP browser:



Add users

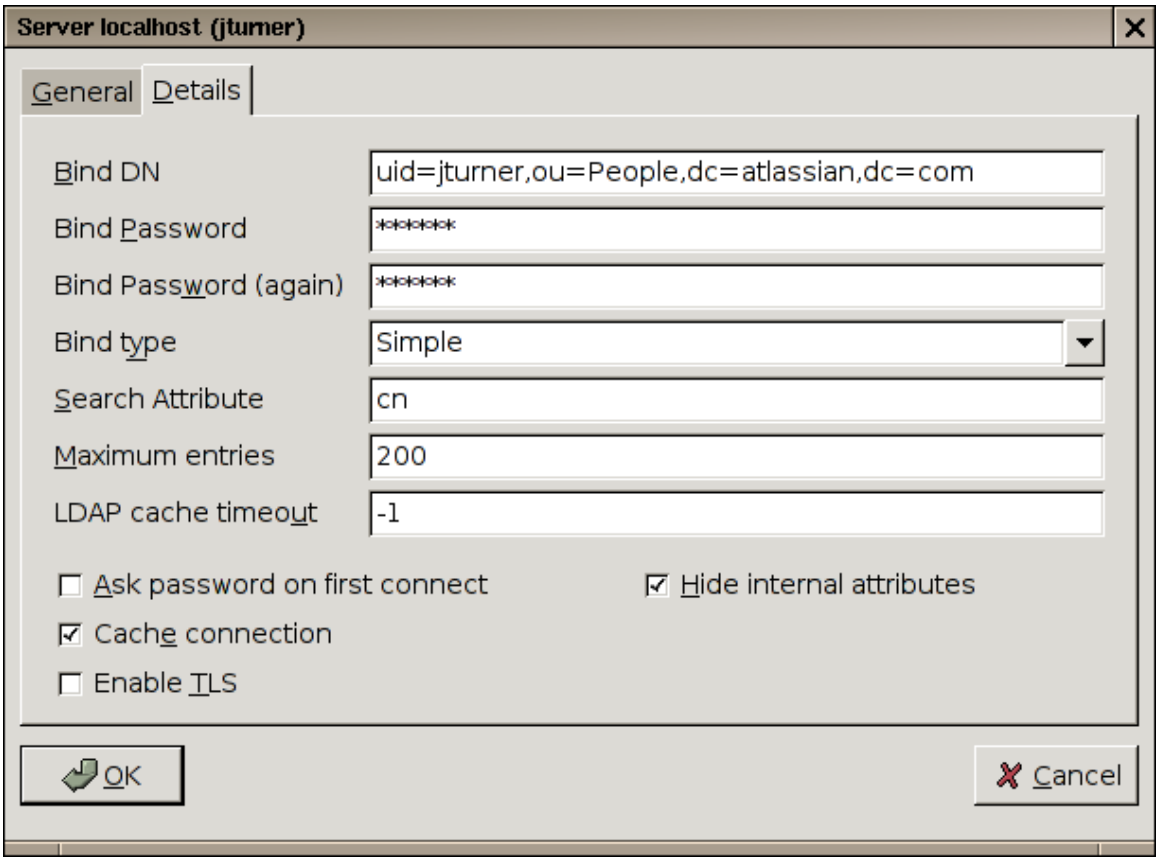
Still in the migrationtools directory, run:

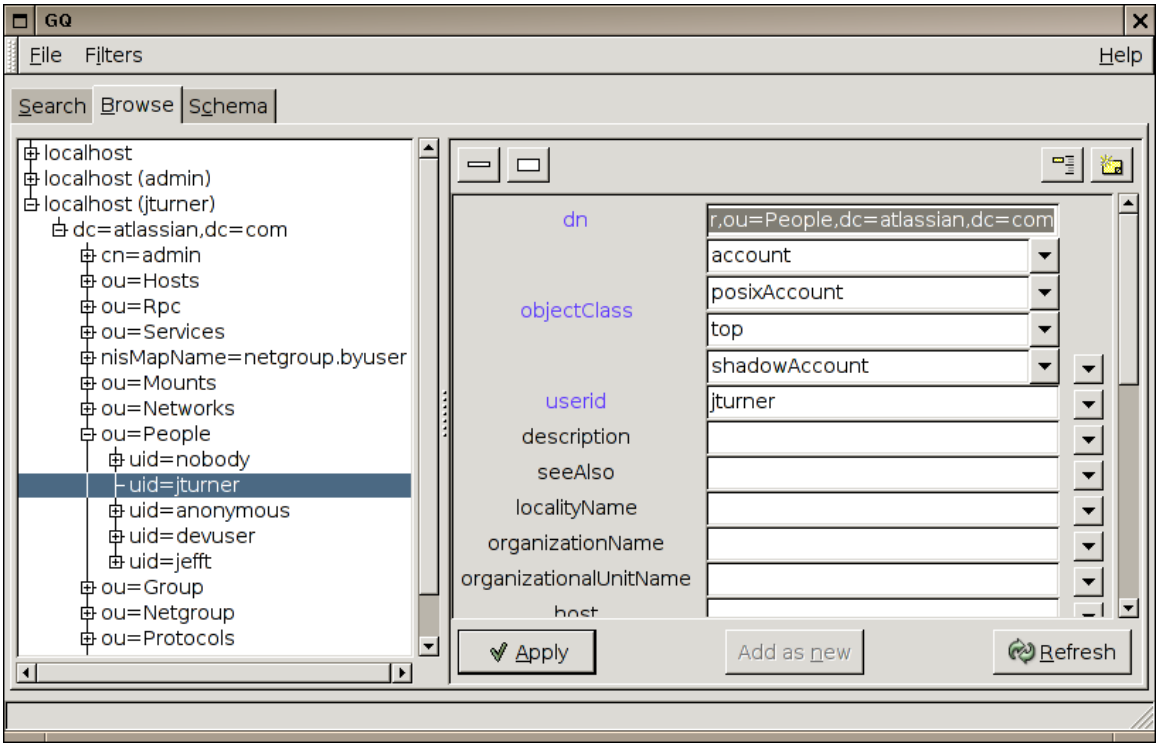
```
teacup:/usr/share/migrationtools# ./migrate_passwd.pl /etc/passwd | ldapadd -x -D "cn=admin,dc=atlassian,dc=com" -W
Enter LDAP Password:
adding new entry "uid=nobody,ou=People,dc=atlassian,dc=com"

adding new entry "uid=jturner,ou=People,dc=atlassian,dc=com"

adding new entry "uid=anonymous,ou=People,dc=atlassian,dc=com"

adding new entry "uid=devuser,ou=People,dc=atlassian,dc=com"

adding new entry "uid=jefft,ou=People,dc=atlassian,dc=com"
```

This creates users, but doesn't set passwords. We must do this manually:

```
teacup:/usr/share/migrationtools# ldappasswd -x -v -S -W -D "cn=admin,dc=atlassian,dc=com"
"uid=jturner,ou=People,dc=atlassian,dc=com"
New password:
Re-enter new password:
Enter LDAP Password:
ldap_initialize( <DEFAULT> )
Result: Success (0)
```

You should now be able to connect anonymously, or as an authenticated user:





Notes

- Some customers have found it helpful to use Likewise Open for LDAP authentication, as it is easy to install and setup.

Installing Java on Ubuntu or Debian

Some linux distributions (notably Debian and Ubuntu) come with a free version of Java called GIJ (from the GCJ project) pre-installed:

```
jturner:~$ java -version
java version "1.4.2"
gij (GNU libgcj) version 4.1.2 (Ubuntu 4.1.2-0ubuntu5)

Copyright (C) 2006 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
jturner:~$
```

Unfortunately GCJ is incomplete, and unable to run Atlassian Java applications without problems.

The solution is to install a supported version of Java. ✅ Visit the Java download page on the Oracle web site to download a supported version of Java for your Linux distribution. Installation instructions are provided on this web site.

After doing this, make sure the correct version of Java is in use by running `java -version`:

```
jturner:~$ java -version
java version "1.6.0"
Java(TM) SE Runtime Environment (build 1.6.0-b105)
Java HotSpot(TM) Server VM (build 1.6.0-b105, mixed mode)
jturner:~$
```

If the GCJ Java is still being used, you will need to explicitly set Linux's default Java platform to a JIRA-supported (e.g. Oracle's) Java platform as the default:

```
jturner:~$ sudo update-alternatives --config java

There are 2 alternatives which provide `java'.

  Selection    Alternative
-----------------------------------------------
*         1    /usr/bin/gij-wrapper-4.1
 +        2    /usr/lib/jvm/java-6-sun/jre/bin/java

Press enter to keep the default[*], or type selection number: 2
Using `/usr/lib/jvm/java-6-sun/jre/bin/java' to provide `java'.
jturner:~$
```

Setting JAVA_HOME

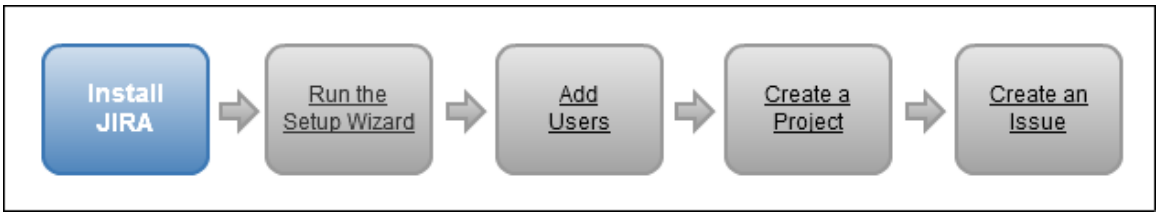
Some programs like Tomcat (bundled with most Atlassian products) need a JAVA_HOME variable set, so they know where Java is installed. This can be set system-wide in `/etc/profile`:

```
jturner:~$ sudo su -
Password:
root:~# cat >> /etc/profile
JAVA_HOME=/usr/lib/jvm/java-6-sun
export JAVA_HOME
root:~#
```

The variable will be set for new terminals:

```
jturner:~$ echo $JAVA_HOME
/usr/lib/jvm/java-6-sun
jturner:~$
```

Installing JIRA on Mac OS X



JIRA running on Mac OS X should only be used for evaluation purposes!

Mac OS X is not a supported operating system for the JIRA server because Oracle JDK and JRE (formerly Sun JDK and JRE), which are the only supported Java platforms for JIRA, are not available for this operating system. Mac OS X is packaged with a JDK optimised for its hardware.

Please note:

- If you encounter problems with your JIRA server running on Mac OS X, they may not be fixed as we do not test JIRA with unsupported Java platforms.
- However, JIRA users can still access your JIRA server through a supported browser on Mac OS X.

To install JIRA on Mac OS X, follow these steps:

- Before you begin
- 1. Download and Install JIRA
- 2. Set JIRA Home
- 3. Create a Dedicated User Account on the Operating System to Run JIRA
- 4. Start JIRA
- 5. Run the Setup Wizard
- Next Steps

Before you begin

Please ensure that you have set JAVA_HOME. (You don't need to install Java as it comes with Mac OS X.)

1. Download and Install JIRA

1. Download the JIRA **tar.gz** file here.
2. Unzip the downloaded file.

2. Set JIRA Home

To specify the location of your JIRA Home Directory:

- Edit the `jira-application.properties` file and set the value of the '`jira.home`' property to the desired location for your JIRA Home Directory.
  See the JIRA Installation Directory page to find where this file is located.
- Use the JIRA Configuration Tool (included with all JIRA distributions except JIRA WAR) to change the location of your JIRA Home Directory.
- Set an environment variable named `JIRA_HOME` in your operating system whose value is the location of your JIRA Home Directory. To do this, do one of the following:
  - Enter the following command at a shell/console prompt before running JIRA:
    - `export JIRA_HOME=/path/to/jira/home`
  - Specify the command above in a script used to start JIRA.

You can specify any location on a disk for your JIRA home directory. Please be sure to specify an absolute path.

Please note that you cannot use the same JIRA home directory for multiple instances of JIRA. We recommend locating your JIRA Home Directory completely independently of the JIRA Installation Directory (i.e. not nesting one within the other) as this will minimise information being lost during major operations (e.g. backing up and restoring instances).

3. Create a Dedicated User Account on the Operating System to Run JIRA

A dedicated user should be created to run JIRA, as JIRA runs as the user it is invoked under and therefore can potentially be abused. For example:

- If your operating system is *nix-based (for example, Linux or Solaris), type the following in a console:
  ```
  $ sudo /usr/sbin/useradd --create-home --comment "Account for running JIRA" --shell /bin/bash jira
  ```
- If your operating system is Windows:
  1. Create the dedicated user account by either:
     - Typing the following at the Windows command line:
       ```
       > net user jira mypassword /add /comment:"Account for running JIRA"
       ```

(This creates a user account with user name 'jira' and password 'mypassword'. You should choose your own password.)
- Opening the Windows 'Computer Management' console to add your 'jira' user with its own password.

2. *(Optional)* Use the Windows 'Computer Management' console to remove the 'jira' user's membership of all unnecessary Windows groups, such as the default 'Users' group.
   If Windows is operating under a Microsoft Active Directory, ask your Active Directory administrator to create your 'jira' account (with no prior privileges).

Ensure that *only the following directories* can be written to by this dedicated user account (e.g. 'jira'):

- The following subdirectories of your JIRA Installation Directory for 'recommended' JIRA distributions (or for JIRA WAR distributions, the installation directory of the Apache Tomcat application running JIRA):
  - `logs`
  - `temp`
  - `work`
- Your JIRA Home Directory.

Do not make the JIRA Installation Directory itself writeable by the dedicated user account.

See also Tomcat security best practices.

4. Start JIRA

Run `bin/startup.sh` to start JIRA.

JIRA will be launched in a black 'Tomcat' window (do not close this window). Wait until the startup messages have finished.

To access JIRA, go to your web browser and type this address: http://localhost:8080.

If JIRA does not appear, you may need to change the port that JIRA runs on.

If something goes wrong, please verify that Java is installed correctly. If the problem persists, please contact us — we're happy to help.

5. Run the Setup Wizard

See Running the Setup Wizard.

Next Steps

- See JIRA 101 to start creating Projects, creating Users, and customising your JIRA instance.
- By default, JIRA uses the standard Tomcat port (i.e. 8080). If you need another application to run on that port, either now or in the future, please see Changing JIRA's TCP Ports.
- As part of its installation process, JIRA automatically installs, configures and connects itself to an HSQLDB database. This is fine for evaluation purposes, however HSQLDB is prone to data corruption. For production installations, we strongly recommend that you connect JIRA to an external database.
- To get the most out of JIRA, please see Optimising Performance.

Configure JIRA as service on Mac OS X

Apple Mac OS X is not a supported operating system for the JIRA server, as JIRA is only tested against Oracle JDK and JRE (formerly Sun JDK and JRE).
Please note: This does not effect your JIRA end users, who can still use Mac OS X with any of the supported browsers.

If you want to run JIRA as a server on OSX, you will need to configure it to load as a userdaemon. OSX has migrated configuration scripts from services such as cron, rc, or init.d to the launchd utility. There are some good introductory and in-depth explanations of it's function on the web. You can find out more about launchd here:

http://developer.apple.com/macosx/launchd.html

It's easier to use Lingon (http://lingon.sourceforge.net/) to define your plist xml definitions for import into launchd; although because launchd does not permit forking of processes you will need to call $TOMCAT_HOME/bin/catalina.sh directly.

Example definition of com.atlassian.jira.plist:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>Label</key>
        <string>com.atlassian.jira</string>
        <key>ProgramArguments</key>
        <array>

<string>/usr/local/apache-tomcat-5.5.23/bin/catalina.sh</string>
                <string>run</string>
        </array>
        <key>RunAtLoad</key>
        <true/>
        <key>ServiceDescription</key>
        <string>JIRA autoloaded as a service</string>
        <key>UserName</key>
        <string>pw</string>
</dict>
</plist>
```

I'd advise adding the <UserName> tag to tell OSX which user to run Tomcat under, eg running Tomcat under root is not recommended. JAVA_HOME will be inherited as an environment variable for whichever user you define in the XML definition - so configure it for that user's .profile in their home directory.

If you wanted to run JIRA as a WAR web-archive, and use OSX's factory install of Tomcat, please see the JIRA WAR installation instructions.

Is Clustering or Load Balancing JIRA Possible

Atlassian recommends working with a partner to provide either of these configurations.

Does JIRA support clustering for scaling?

Not at this time. Currently, splitting JIRA into multiple instances is our recommended means of scaling JIRA — please see Splitting a JIRA instance.

Note: If you wish to run JIRA across multiple nodes you will need a license for each node. You can place your order from our secure online order form .

Does JIRA support high-availability?

If you are interested in clustering for hardware failover, you can configure a distributed database, load balancer front-end and two separate servers. The primary JIRA instance runs on the main server, while the second server contains an unstarted, free development license instance of JIRA. The load balancer directs 100% of requests to the primary as long as it is responsive, but if the primary server goes down, the load balancer starts the backup instance and directs all requests there until manually reset.

There are several problems with this model. Specifically, attachments and indexes will be out of date on the backup server. You may be able to institute a script to update these.

java.lang.NoClassDefFoundError

If you get the following error when starting JIRA:

> *"java.lang.NoClassDefFoundError:com/atlassian/jira/issue/search/parameters/lucene/SingleFieldMultiValueLuceneParameter"*

this means that Windows XP's unzip is broken. See the Installation Guide for how to avoid this.

JVM and Appserver configuration info

LicenseFactory error after upgrading JIRA

If, after upgrading JIRA, you get an error containing 'com/atlassian/jira/license/LicenseFactory', it means that your application server is using old cached JSPs from the previous JIRA version. Please delete the directory where the app server keeps these (the **work/** directory in Tomcat; the **application-deployments/** directory in Orion; etc), and restart.

Logging request headers

If you are having trouble authenticating to JIRA or a web application, it can be useful to log the details of all HTTP request headers that are being sent to the web application. If your application server is Tomcat, you can do this with the **Request Dumper Valve**.

Add the following entry to the <Engine> section of your Tomcat `conf/server.xml` file:

```
<Valve className="org.apache.catalina.valves.RequestDumperValve"/>
```

Then restart JIRA.

You will get **lots** of entries like the following in your `logs/catalina.out` log file:

```
12/11/2007 16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:             header=user-agent=Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.8.1.6)
Gecko/20071008 Ubuntu/7.10 (gutsy) Firefox/2.0.0.6
12/11/2007 16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:
header=accept=text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/
16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:             header=accept-language=en-us,en;q=0.5
12/11/2007 16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:             header=accept-encoding=gzip,deflate
12/11/2007 16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:             header=accept-charset=ISO-8859-1,utf-8;q=0.7,*;q=0.7
12/11/2007 16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:             header=keep-alive=300
12/11/2007 16:27:06 org.apache.catalina.valves.RequestDumperValve invoke
INFO:             header=connection=keep-alive
```

Running multiple instances of JIRA on one machine

It's possible to run multiple JIRA instances on one machine as long as the instance completes the following requirements:

- Not sharing the same listening or shutdown port of any other instance or service.
- Not sharing the same database with another instance.
- When using services within windows, using 2 different windows service definitions.
- Not sharing the same JIRA home folder.
- Not deploying multiple instances using a Single Tomcat Application Container.
- Having separate licenses for each of the instances.

By default, JIRA runs on port **8080** (and hence is available at (http://<`yourserver`>:8080, eg. http://localhost:8080).

If you want to run another JIRA instance but already has a service claiming port **8080**, there will be a conflict, and JIRA will fail to start. You may see errors like this:

```
LifecycleException:  Protocol handler initialization failed: java.net.BindException: Address
already in use:8080
```

This can be fixed by changing JIRA to use another listening port (eg. **8090**) and shutdown port (eg. **8015**). This is done by editing `conf\server.xml` (eg. in Wordpad). The start of the file looks like:

```
<Server port="8005" shutdown="SHUTDOWN">

  <Service name="Catalina">

    <Connector port="8080"
      maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
      enableLookups="false" redirectPort="8443" acceptCount="100" connectionTimeout="20000"
disableUploadTimeout="true" />
```

Here, change the shutdown port from "**8005**" to "**8015**" and change the listening port from "**8080**" to "**8090**" (or some other free port — see below).

Then restart JIRA (bin\shutdown.bat; bin\startup.bat) and point a browser to http://<`yourserver`>:**8090** (eg. http://localhost:8090).

If you are running on a Unix server and bind the ports below 1024 (such as port 80 for example), you will **need to start JIRA as root** in order to successfully bind to the port

Which port number should I choose?

If you are not sure which port number to choose, use a tool such as *netstat* to determine which port numbers are free to use by JIRA. The highest port number that can be used is 65535 because it is the highest number which can be represented by an unsigned 16 bit binary number. The Internet Assigned Numbers Authority (IANA) lists the registration of commonly used port numbers for well-known Internet services, it's advisable to avoid any of those ports.

Solaris ClassNotFoundException

After unpacking the WAR on Solaris, JIRA fails to start with a ClassNotFoundException once deployed. How is this fixed?

On Solaris, the default **tar** utility should be avoided as it cannot handle long filenames. **GNU tar** should be used instead in order to handle long filenames found within the JIRA distribution, it can usually be found at:

```
/usr/sfw/bin/
```

Stacktrace example:

```
2006-11-15 15:43:27,539 ERROR [ContainerBase.[Catalina].[localhost].[/]] Error configuring
application listener of class com.atlassian.jira.upgrade.ConsistencyLauncher
java.lang.ClassNotFoundException: com.atlassian.jira.upgrade.ConsistencyLauncher
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1332)
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1181)
            at
org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3617)
            at org.apache.catalina.core.StandardContext.start(StandardContext.java:4104)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardHost.start(StandardHost.java:718)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardEngine.start(StandardEngine.java:442)
            at org.apache.catalina.core.StandardService.start(StandardService.java:450)
            at org.apache.catalina.core.StandardServer.start(StandardServer.java:683)
            at org.apache.catalina.startup.Catalina.start(Catalina.java:537)
            at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
            at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
            at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
            at java.lang.reflect.Method.invoke(Method.java:324)
            at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:271)
            at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:409)
2006-11-15 15:43:27,582 ERROR [ContainerBase.[Catalina].[localhost].[/]] Error configuring
application listener of class com.atlassian.jira.upgrade.UpgradeLauncher
java.lang.ClassNotFoundException: com.atlassian.jira.upgrade.UpgradeLauncher
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1332)
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1181)
            at
org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3617)
            at org.apache.catalina.core.StandardContext.start(StandardContext.java:4104)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardHost.start(StandardHost.java:718)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardEngine.start(StandardEngine.java:442)
            at org.apache.catalina.core.StandardService.start(StandardService.java:450)
            at org.apache.catalina.core.StandardServer.start(StandardServer.java:683)
            at org.apache.catalina.startup.Catalina.start(Catalina.java:537)
            at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
            at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
            at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
            at java.lang.reflect.Method.invoke(Method.java:324)
            at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:271)
```

```
            at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:409)
2006-11-15 15:43:27,592 ERROR [ContainerBase.[Catalina].[localhost].[/]] Error configuring
application listener of class com.atlassian.jira.scheduler.JiraSchedulerLauncher
java.lang.ClassNotFoundException: com.atlassian.jira.scheduler.JiraSchedulerLauncher
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1332)
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1181)
            at
org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3617)
            at org.apache.catalina.core.StandardContext.start(StandardContext.java:4104)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardHost.start(StandardHost.java:718)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardEngine.start(StandardEngine.java:442)
            at org.apache.catalina.core.StandardService.start(StandardService.java:450)
            at org.apache.catalina.core.StandardServer.start(StandardServer.java:683)
            at org.apache.catalina.startup.Catalina.start(Catalina.java:537)
            at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
            at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
            at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
            at java.lang.reflect.Method.invoke(Method.java:324)
            at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:271)
            at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:409)
2006-11-15 15:43:27,602 ERROR [ContainerBase.[Catalina].[localhost].[/]] Error configuring
application listener of class com.atlassian.jira.soap.axis.JiraAxisHttpListener
java.lang.ClassNotFoundException: com.atlassian.jira.soap.axis.JiraAxisHttpListener
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1332)
            at
org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1181)
            at
org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:3617)
            at org.apache.catalina.core.StandardContext.start(StandardContext.java:4104)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardHost.start(StandardHost.java:718)
            at org.apache.catalina.core.ContainerBase.start(ContainerBase.java:1012)
            at org.apache.catalina.core.StandardEngine.start(StandardEngine.java:442)
            at org.apache.catalina.core.StandardService.start(StandardService.java:450)
            at org.apache.catalina.core.StandardServer.start(StandardServer.java:683)
            at org.apache.catalina.startup.Catalina.start(Catalina.java:537)
            at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
            at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
            at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
            at java.lang.reflect.Method.invoke(Method.java:324)
            at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:271)
            at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:409)
```

```
2006-11-15 15:43:27,606 ERROR [ContainerBase.[Catalina].[localhost].[/]] Skipped installing
application listeners due to previous error(s)
```

Windows cannot find -Xms128m

When running startup.bat I get an error message: "Windows cannot find '-Xms128m'"

This error message means that the Java JDK (Java Development Kit) is not installed or the **JAVA_HOME** environment variable has not been set correctly. Please refer to the Java installation guide.

Contributing to the JIRA Documentation

Would you like to share your JIRA hints, tips and techniques with us and with other JIRA users? We welcome your contributions.

On this page:

- Tweeting your Hints and Tips – Tips via Twitter
- Blogging your Technical Tips and Guides – Tips of the Trade
- Contributing Documentation in Other Languages
- Updating the Documentation Itself
  - Getting Permission to Update the Documentation
  - Our Style Guide
  - How we Manage Community Updates

Tweeting your Hints and Tips – Tips via Twitter

Do you have hints and tips about your JIRA bug tracker to share with the world? Even more, would you like to see your tips appear on a page in the Atlassian documentation? Just tweet with the hash tag "`#JIRATips`" and see your hint appear in our documentation. Then grab a **badge for your blog!** More...

Blogging your Technical Tips and Guides – Tips of the Trade

Have you written a blog post describing a specific configuration of JIRA or a neat trick that you have discovered? Let us know, and we will link to your blog from our documentation. More....

Contributing Documentation in Other Languages

Have you written a guide to JIRA in a language other than English, or translated one of our guides? Let us know, and we will link to your guide from our documentation. More....

Updating the Documentation Itself

Have you found a mistake in the documentation, or do you have a small addition that would be so easy to add yourself rather than asking us to do it? You can update the documentation page directly.

Getting Permission to Update the Documentation

Please submit the **Atlassian Contributor License Agreement.**

Our Style Guide

Please read our short guidelines for authors.

How we Manage Community Updates

Here is a quick guide to how we manage community contributions to our documentation and the copyright that applies to the documentation:

- **Monitoring by technical writers.** The Atlassian technical writers monitor the updates to the documentation spaces, using RSS feeds and watching the spaces. If someone makes an update that needs some attention from us, we will make the necessary changes.
- **Wiki permissions.** We use wiki permissions to determine who can edit the documentation spaces. We ask people to sign the Atlassian Contributor License Agreement (ACLA) and submit it to us. That allows us to verify that the applicant is a real person. Then we give them permission to update the documentation.
- **Copyright.** The Atlassian documentation is published under a Creative Commons CC BY license. Specifically, we use a Creative Commons Attribution 2.5 Australia License. This means that anyone can copy, distribute and adapt our documentation provided they

acknowledge the source of the documentation. The CC BY license is shown in the footer of every page, so that anyone who contributes to our documentation knows that their contribution falls under the same copyright.

RELATED TOPICS

Tips of the Trade
Author Guidelines
Atlassian Contributor License Agreement

Tips of the Trade

Below are some links to external blog posts and articles containing technical tips and instructions on setting up and using JIRA. This page presents an opportunity for customers and community authors to share information and experiences.

The references here are links to technical 'how to' guides written by bloggers who use JIRA. For feature tours, solution tours and other information about bug and issue trackers, please refer to the Atlassian website and to our evaluator resources.

Please be aware that these are external blogs and articles.

Most of the links point to external sites, and some of the information is relevant to a specific release of JIRA. Atlassian provides these links because the information is useful and relevant at the time it was written. Please check carefully whether the information is still relevant when you read it, and whether it is relevant to your version of JIRA. Unless explicitly stated, Atlassian does not offer support for third-party extensions or plugins. The information in the linked blog posts has not been tested or reviewed by Atlassian. We recommend that you test all solutions on a **test** server before trying them on your production site.

On this page:

- JIRA 4 Linux Administrator's Guide and Workflow Management
- Install JIRA on RHEL5, Single Tomcat with AJP and MySQL
- JIRA Groups and JIRA Project Roles
- 'Send reminder on' custom field for Jira
- User Activity Statistics
- Showing custom fields in 'sub-task' columns
- Using User Properties
- Making it easier to maintain JIRA workflows
- Jira Workflow Report Update
- NetBeans 6.7 RC1 - and JIRA support (beta)
- JIRA To Omnifocus Script
- Update JIRA-issues with OmniFocus
- Git branches to handle contributor patches
- Setting issue security level by issue type
- Unique Issue ID Across Projects

Installation

JIRA 4 Linux Administrator's Guide and Workflow Management

- By: James Intriglia, on 'Getting Things Done in a Virtual World'
- About — This article covers the following topics:
  - Installing JIRA on different flavours of Linux with additional sections on how to configure and administer JIRA (This is a PDF document, whose link can be found at the end of this page.)
  - JIRA workflow diagrams speeding up JIRA application development
- Date: May 2010
- Related documentation: JIRA Installation and Upgrade Guide

Install JIRA on RHEL5, Single Tomcat with AJP and MySQL

- By: Brett Ryan, on the 'JIRA Community Space'
- About: How to install JIRA on Red Hat Enterprise Linux 5, using Apache Tomcat behind an Apache HTTP Server with AJP and MySQL
- Date: 9 April 2010
- Related documentation: JIRA Installation and Upgrade Guide

  Please be aware that this guide contains advanced procedures that should only be attempted by individuals who are familiar with configuring Apache Tomcat, Apache HTTP Server and AJP on Linux.

Administration

JIRA Groups and JIRA Project Roles

- By: Matt Doar, on blog 'jiradev.blogspot.com'
- About: How to set up JIRA permissions for specific projects
- Date: 27 September 2010
- Related documentation: Managing Groups, Managing Project Roles

'Send reminder on' custom field for Jira

- By: Sam Haldane, on blog 'blog.samhaldane.com'
- About: How to set up JIRA to send issue reminders to users on specified dates
- Date: 17 August 2009
- Related documentation: Adding a Custom Field, Saving Searches ('Issue Filters')

User Activity Statistics

- By: Zaccary Craven, on blog 'Tips and tricks for JIRA administrators'
- About: How to show a list of all usernames along with the number of times each user has created a comment
- Date: 19 January 2009
- Related documentation: Adding the Issue Statistics Gadget

Showing custom fields in 'sub-task' columns

- By: Zaccary Craven, on blog 'Tips and tricks for JIRA administrators'
- About: How to show the values of subtask custom fields on your issue screens
- Date: 8 December 2008
- Related documentation: Custom fields

Using User Properties

- By: Matt Doar, on blog 'Consulting Toolsmiths'
- About: How to add, display and filter the user properties with the JIRA Toolkit plugin
- Date: 20 February 2008
- Related documentation: Managing Users

Workflow

Making it easier to maintain JIRA workflows

- By: Matt Doar, on blog 'Consulting Toolsmiths'
- About: How to display the name of the screen used by each transition in a workflow in one place
- Date: 1 July 2009
- Related documentation: Configuring Workflow

Jira Workflow Report Update

- By: Jamie Echlin, on the 'onresolve team blog'
- About: Visualising JIRA workflows, with hints about a common problem when defining resolutions in JIRA workflows
- Date: 19 December 2008
- Related documentation: Configuring Workflow

Integration with Other Tools

NetBeans 6.7 RC1 - and JIRA support (beta)

- By: Fabrizio Giudici, on 'Fabrizio Giudici's Blog'
- About: How to get JIRA integration in NetBeans 6.7 RC1
- Date: 1 June 2009

JIRA To Omnifocus Script

- By: David Martinez, on blog 'Hackerdude'
- About: A script that logs into JIRA and creates OmniFocus tasks for each of the JIRA items that are assigned to you, so they sync to your Omnifocus for iPhone, you only have to keep track of one inbox, etc.
- Date: 4 March 2009

Update JIRA-issues with OmniFocus

- By: Alain Petignat, on blog 'sequenz'
- About: Updating JIRA fields (time estimate, due date and assignee) directly from OmniFocus
- Date: 12 April 2009

Development

Git branches to handle contributor patches

- By: Julien Ponge, on "JPz'log"
- About: Managing patches in pending state via Git, SVN and JIRA
- Date: 4 December 2008

Setting issue security level by issue type

- By: Jamie Echlin, on the 'onresolve team blog'
- About: Two ways to set JIRA issue security levels by issue type
- Date: 5 August 2008
- Related documentation: Configuring Issue Level Security

Unique Issue ID Across Projects

- By: Surya Suravarapu, on 'Surya Suravarapu's Blog'
- About: A plugin that allows you to have unique issue IDs across all your JIRA projects
- Date: 14 July 2009
- Related documentation:
  - Changing the Project Key Format
  - Change History

Have you written a technical tip for JIRA?

Add a comment to this page, linking to your blog post or article. We will include it if the content fits the requirements of this page.

Feedback?

Your first port of call should be the author of the linked blog post. If you want to let us know how useful (or otherwise) a linked post is, please add a comment to this page.

Other Sources of Information

JIRA documentation
Evaluator resources
Atlassian website
Atlassian forums
Atlassian blog
JIRA plugins

Tips via Twitter

This page displays a continuously-updated list of tweets from Atlassians and others, giving hints and tips about JIRA issue tracker. Anyone can write a tip and have it show on this page. The live Twitter stream shows recent tweets containing the word 'JIRATips'or the tag '#JIRATips'.

Want to join in? Just tweet with the tag '`#JIRATips`' somewhere in the text. Then grab a badge for your blog.

- Viewing the Tweets in Twitter
- Adding a JIRA Tweets Badge to your Blog
- Adding your own Tip

Please be aware that anyone can tweet anything.

Atlassian does not monitor the tips in this Twitter stream. Anyone can tweet anything they like. We display these tips because we believe most people will do the right thing and tweet good tweets. Please check that a tweet is relevant to you before following its advice.

Viewing the Tweets in Twitter

If you prefer, you can view the search in Twitter itself.

Adding a JIRA Tweets Badge to your Blog

Would you like to let other people know that you tweet your JIRA tips? Use the code samples below to add a badge to your blog or another social site.

Choose one of these options to add the badge:

- **Badge only**
  Copy the code below and paste it into your blog to include just the badge with a link to this documentation page:

```
<a
href="http://atlss.in/JIRATips?utm_source=jira-tips-badge&utm_medium=badge&utm_campaign=techw
title="Tips via Twitter for JIRA issue tracker"><img
src="http://confluence.atlassian.com/download/attachments/222200745/Twitter-Tips-JIRA.png"
alt="Tips via Twitter for JIRA issue tracker" border="0" ></a>
```

  This is what you will get:

- **Badge and words**
  Copy the code below and paste it into your blog to include the badge and some words encouraging other people to tweet too:

```
<a
href="http://atlss.in/JIRATips?utm_source=jira-tips-badge&utm_medium=badge&utm_campaign=techw
title="Tips via Twitter for JIRA issue tracker"><img
src="http://confluence.atlassian.com/download/attachments/222200745/Twitter-Tips-JIRA.png"
alt="Tips via Twitter for JIRA issue tracker" border="0" ></a>
<p>Got a JIRA tip? <a href="http://twitter.com/?status=My tip %23JIRATips" title="Tweet
at Twitter">Tweet it now</a> then <a
href="http://atlss.in/JIRATips?utm_source=jira-tips-badge&utm_medium=badge&utm_campaign=techw
title="Tips via Twitter for JIRA issue tracker">see it in the JIRA docs</a>.</p>
```

  This is what you will get:

Got a JIRA tip? Tweet it now then see it in the JIRA docs.

Adding your own Tip

Quick guide to tweeting a tip

Just tweet with the word '**`#JIRATips`**' somewhere in the text. Your tweet will appear in the Twitter stream on this page.

Would you like to share your information and experiences via Twitter and have your tweet appear on this page? **Awesome!** Here are the full instructions.

To tweet a JIRA tip,

1. Go to Twitter.com in your browser.
2. If you already have a Twitter username, sign in to Twitter now. If you do not have a Twitter username, sign up for one and follow the Twitter instructions to confirm your account details.
3. Enter your tip into the Twitter text box labelled '**What's happening**'. Note that your tip can contain a maximum of 140 characters:
    - Type the words for your tip.
    - If you want people to click through to a web page to see more details about your tip, enter a web address. If the web address is long, you can convert it to a shortened address at bit.ly or one of the other web services that offer URL shortening.
    - Include the key word `#JIRATips` somewhere in your tweet. This will ensure that your tip appears in the Twitter stream on this documentation page.
4. Click '**Tweet**' to send your tweet.
5. Refresh this documentation page to see your tweet appear. It may take a few minutes, depending on the volume of tweets that Twitter is handling.

Other Sources of Information

Tips of the Trade
JIRA documentation
Evaluator resources
Atlassian website
Atlassian forums
Atlassian blog
JIRA plugins

JIRA Documentation in Other Languages

Below are some links to JIRA documentation written in other languages. In some cases, the documentation may be a translation of the English documentation. In other cases, the documentation is an alternative guide written from scratch in another language. This page presents an opportunity for customers and community authors to share documentation that they have written in other languages.

Please be aware that these are external guides.

Most of the links point to external sites, and some of the information is relevant to a specific release of JIRA. Atlassian provides these links because the information is useful and relevant at the time it was written. Please check carefully whether the information is still relevant when you read it, and whether it is relevant to your version of JIRA. The information in the linked guides has not been tested or reviewed by Atlassian.

On this page:

- Manual de administración JIRA 3.13
- Handleiding JIRA
- JIRA

Spanish

Manual de administración JIRA 3.13

- By: Rocío Navarro Lacoba, published in the JIRA Community space on this wiki
- Topic: JIRA Administration Guide, a translation of the English JIRA 3.13 Administration Guide
- JIRA version: 3.13
- Date added: 25 November 2010
- Latest related English documentation: JIRA Administrator's Guide

Dutch

Handleiding JIRA

- By: Bart Wesdorp of theFactor.e, published in the JIRA Community space on this wiki
- Topic: JIRA User Guide, a quick-start guide to JIRA.
- JIRA version: Based on JIRA 4.1.
- Date added: 8 December 2010
- Latest related English documentation: JIRA User's Guide

Japanese

JIRA

- **By:** Go2Group and Atlassian, on the Atlassian Japan wiki
- **Topic:** JIRA user's guide, administrator's guide and developer's guide
- **JIRA version:** 4.0
- **Date added:** 14 December 2010
- **Latest related English documentation:** JIRA 5.0

Adding Your Own Guide to this Page

Have you written a guide for JIRA in another language? Add a comment to this page, linking to your guide. We will include it if the content fits the requirements of this page.

Giving Feedback about One of the Guides

If you have feedback on one of the guides listed above, please give the feedback to the author of the linked guide.

If you want to let us know how useful (or otherwise) one of these guides is, please add a comment to this page.

Other Sources of Information

JIRA documentation
Atlassian website
Atlassian blog
JIRA plugins

GreenHopper for JIRA Guide

GreenHopper 5.9 has been released! Read the release notes for more information on the features that have been added.

GreenHopper is a JIRA plug-in that adds a broad collection of agile project management capabilities to JIRA, and extends JIRA as a powerful platform for agile development teams. GreenHopper simplifies the planning and organisation of tasks, workflows and reporting for agile teams.

Please refer to the GreenHopper documentation for more information.