Documentation for JIRA 6.1
Contents

JIRA Documentation .................................................. 12
  JIRA 101 ........................................................................ 13
    Set up a JIRA site ....................................................... 14
    Add users ..................................................................... 16
    Create a project .......................................................... 18
    Create an issue ............................................................ 21
    Search for issues and create reports ............................... 23
    Configure permissions .................................................. 27
    Configure workflows and screens ................................... 30
  Extend JIRA ................................................................. 34
JIRA User's Guide .......................................................... 38
  JIRA Concepts ............................................................. 41
    What is a Project .......................................................... 41
    What is Workflow ......................................................... 42
    What is an Issue ........................................................... 43
  Getting Started ............................................................ 46
    Logging in to JIRA ......................................................... 46
    Exploring the JIRA Workspace ...................................... 47
    Using Keyboard Shortcuts ............................................. 47
    Using JIRA on a Mobile Device ....................................... 59
    Getting started with Bamboo ......................................... 60
  Working with an Issue ................................................... 60
    Adding a field to an issue ............................................... 60
    Attaching a File ........................................................... 64
    Attaching a Screenshot .................................................. 65
    Cloning an Issue ........................................................... 66
    Commenting on an Issue ................................................. 68
    Creating an Issue .......................................................... 70
    Creating a Sub-Task ....................................................... 71
    Editing an Issue ........................................................... 74
    Editing Rich-Text Fields ............................................... 74
    Emailing an Issue ........................................................ 78
    Labeling an Issue ........................................................ 80
    Linking Issues ............................................................. 81
    Logging Work on an Issue .............................................. 85
    Modifying Multiple ('Bulk') Issues ................................. 91
    Moving an Issue .......................................................... 98
    Scheduling an Issue ...................................................... 99
    Setting Security on an Issue .......................................... 100
    Viewing an Issue's Change History ................................. 100
    Viewing an Issue's Crucible Reviews .............................. 101
    Viewing an Issue's FishEye Changesets ........................... 101
    Viewing the Bamboo Builds related to an Issue ................. 102
    Watching and Voting on an Issue .................................... 102
    Integrating with code development tools .......................... 104
  Searching for Issues .................................................... 105
    Basic Searching ........................................................ 106
    Using Quick Search ..................................................... 109
    Advanced Searching ................................................... 112
    Advanced Searching Functions ...................................... 160
    Performing Text Searches ............................................. 187
    Using the Issue Navigator ............................................. 193
      Customizing your Issue Navigator ................................. 197
    Using Filters ............................................................ 199
    Working with Search Result Data .................................... 207
      Displaying Search Results as a Chart ......................... 207

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Displaying Search Results in XML</td>
<td>210</td>
</tr>
<tr>
<td>Exporting Search Results to Microsoft Excel</td>
<td>218</td>
</tr>
<tr>
<td>Exporting Search Results to Microsoft Word</td>
<td>219</td>
</tr>
<tr>
<td>Receiving Search Results as an RSS Feed</td>
<td>220</td>
</tr>
<tr>
<td>Receiving Search Results via Email</td>
<td>222</td>
</tr>
<tr>
<td>Sharing a Search Result</td>
<td>226</td>
</tr>
<tr>
<td>Generating Reports</td>
<td>227</td>
</tr>
<tr>
<td>Workload Pie Chart Report</td>
<td>228</td>
</tr>
<tr>
<td>User Workload Report</td>
<td>231</td>
</tr>
<tr>
<td>Version Workload Report</td>
<td>231</td>
</tr>
<tr>
<td>Time Tracking Report</td>
<td>233</td>
</tr>
<tr>
<td>Single Level Group By Report</td>
<td>235</td>
</tr>
<tr>
<td>Created vs Resolved Issues Report</td>
<td>236</td>
</tr>
<tr>
<td>Resolution Time Report</td>
<td>239</td>
</tr>
<tr>
<td>Pie Chart Report</td>
<td>242</td>
</tr>
<tr>
<td>Average Age Report</td>
<td>244</td>
</tr>
<tr>
<td>Recently Created Issues Report</td>
<td>247</td>
</tr>
<tr>
<td>Time Since Issues Report</td>
<td>250</td>
</tr>
<tr>
<td>Browsing a Project</td>
<td>253</td>
</tr>
<tr>
<td>Browsing a Project's Summary</td>
<td>254</td>
</tr>
<tr>
<td>Browsing a Project's Issues</td>
<td>256</td>
</tr>
<tr>
<td>Browsing a Project's Road Map</td>
<td>257</td>
</tr>
<tr>
<td>Browsing a Project's Change Log</td>
<td>258</td>
</tr>
<tr>
<td>Browsing a Project's Popular Issues</td>
<td>259</td>
</tr>
<tr>
<td>Browsing a Project's Versions</td>
<td>259</td>
</tr>
<tr>
<td>Browsing a Version's Summary</td>
<td>260</td>
</tr>
<tr>
<td>Browsing a Version's Issues</td>
<td>261</td>
</tr>
<tr>
<td>Browsing a Version's Popular Issues</td>
<td>262</td>
</tr>
<tr>
<td>Browsing a Version's Bamboo Builds</td>
<td>262</td>
</tr>
<tr>
<td>Browsing a Project's Components</td>
<td>263</td>
</tr>
<tr>
<td>Browsing a Component's Summary</td>
<td>264</td>
</tr>
<tr>
<td>Browsing a Component's Issues</td>
<td>265</td>
</tr>
<tr>
<td>Browsing a Component's Road Map</td>
<td>266</td>
</tr>
<tr>
<td>Browsing a Component's Change Log</td>
<td>267</td>
</tr>
<tr>
<td>Browsing a Component's Popular Issues</td>
<td>268</td>
</tr>
<tr>
<td>Browsing a Project's Labels</td>
<td>269</td>
</tr>
<tr>
<td>Browsing a Project's Bamboo Builds</td>
<td>270</td>
</tr>
<tr>
<td>Browsing a Project's FishEye Changsets</td>
<td>270</td>
</tr>
<tr>
<td>Browsing a Project's Crucible Reviews</td>
<td>272</td>
</tr>
<tr>
<td>Viewing a Project's Burndown Chart</td>
<td>272</td>
</tr>
<tr>
<td>Customizing the Dashboard</td>
<td>272</td>
</tr>
<tr>
<td>Managing Multiple Dashboard Pages</td>
<td>277</td>
</tr>
<tr>
<td>Changing the Look and Behavior of a Gadget</td>
<td>282</td>
</tr>
<tr>
<td>Adding the Activity Stream Gadget</td>
<td>284</td>
</tr>
<tr>
<td>Adding the Administration Gadget</td>
<td>286</td>
</tr>
<tr>
<td>Adding the Assigned To Me Gadget</td>
<td>287</td>
</tr>
<tr>
<td>Adding the Average Age Gadget</td>
<td>288</td>
</tr>
<tr>
<td>Adding the Bamboo Charts Gadget</td>
<td>291</td>
</tr>
<tr>
<td>Adding the Bamboo Plan Summary Chart Gadget</td>
<td>293</td>
</tr>
<tr>
<td>Adding the Bamboo Plans Gadget</td>
<td>295</td>
</tr>
<tr>
<td>Adding the Bugzilla ID Search Gadget</td>
<td>296</td>
</tr>
<tr>
<td>Adding the Calendar Gadget</td>
<td>297</td>
</tr>
<tr>
<td>Adding the Clover Coverage Gadget</td>
<td>299</td>
</tr>
<tr>
<td>Adding the Created vs Resolved Gadget</td>
<td>300</td>
</tr>
<tr>
<td>Adding the Crucible Charts Gadget</td>
<td>304</td>
</tr>
<tr>
<td>Adding the Favorite Filters Gadget</td>
<td>305</td>
</tr>
<tr>
<td>Adding the Filter Results Gadget</td>
<td>306</td>
</tr>
<tr>
<td>Adding the FishEye Charts Gadget</td>
<td>308</td>
</tr>
<tr>
<td>Adding the FishEye Recent Changsets Gadget</td>
<td>310</td>
</tr>
<tr>
<td>Adding the In Progress Gadget</td>
<td>311</td>
</tr>
<tr>
<td>Adding the Introduction Gadget</td>
<td>312</td>
</tr>
<tr>
<td>Adding the Issue Statistics Gadget</td>
<td>313</td>
</tr>
</tbody>
</table>

Creative Commons Attribution 2.5 Australia License.
Defining a Project .................................................. 453
Editing a Project Key ............................................. 458
Simple Issue Tracking project .................................. 462
Software Development project .................................. 463
Managing Project Role Membership .......................... 465
Defining a Component ............................................. 466
Managing Versions .................................................. 468
Running a Bamboo Build when Releasing a Version .... 470
Creating Release Notes ............................................ 471
Configuring Security ............................................... 472
Configuring Issue-level Security ............................... 475
Managing Project Permissions ................................... 479
Managing Global Permissions ................................... 486
Configuring Secure Administrator Sessions ............... 490
Preventing Security Attacks ..................................... 491
JIRA Cookies .......................................................... 493
JIRA Admin Helper ................................................... 495
Configuring Fields and Screens ................................. 497
Configuring Built-in Fields ....................................... 499
Defining 'Issue Type' Field Values ............................. 499
Defining 'Priority' Field Values ................................. 507
Defining 'Resolution' Field Values ............................. 508
Defining 'Status' Field Values .................................. 509
Translating Resolutions, Priorities, Statuses and Issue Types ........................................... 510
Adding a Custom Field ............................................. 511
Configuring a Custom Field ....................................... 513
Creating Help for a Custom Field ............................... 517
Specifying Field Behavior ......................................... 517
Associating Field Behavior with Issue Types .............. 522
Configuring Renderers ............................................. 525
Defining a Screen ................................................... 532
Associating a Screen with an Issue Operation ............. 536
Associating Screen and Issue Operation Mappings with an Issue Type .................................... 540
Configuring Workflow ............................................... 544
Activating workflow ............................................... 553
Classic mode workflow designer ................................. 555
Configuring workflow schemes ................................ 557
Working in text mode .............................................. 560
Sharing your workflow ............................................. 564
Advanced workflow configuration .............................. 569
Adding a custom event ............................................. 585
Configuring the initial status .................................... 587
Configuring JIRA with HipChat .................................. 589
Using validators with custom fields ......................... 590
Using XML to create a workflow ............................... 590
Workflow properties ............................................... 591
Configuring Email ................................................... 592
Configuring Email Notifications ............................... 592
Configuring JIRA's SMTP Mail Server to Send Notifications ........................................... 595
Creating a Notification Scheme ................................. 600
Customizing Email Content ..................................... 605
Creating Issues and Comments from Email ............... 606
Configuring JIRA to Receive Email from a POP or IMAP Mail Server ........................................ 622
Using Gmail as a JIRA Mail Server ......................... 624
Migrating from Other Issue Trackers .......................... 626
Importing Data from Bugzilla ..................................... 628
Importing Data from FogBugz for Your Server .......... 637
Importing Data from FogBugz On Demand .................. 644
Importing Data from Mantis ..................................... 650
Importing Data from Pivotal Tracker ......................... 658
Importing Data from Trac ......................................... 661
Importing Data from CSV ......................................... 664
Commonly Asked CSV Questions and Known Issues ........................................ 672
How to Import CSV Data with Pvcs Command ............................................. 674
Importing Data from JSON (beta release) .................................................. 674
Importing Data from Redmine ................................................................. 678
Moving or Archiving Individual Projects .................................................... 681
Archiving a Project .................................................................................. 681
Splitting a Jira instance .......................................................................... 682
Integrating with a Source Control System ................................................ 683
Integrating Jira with FishEye ...................................................................... 683
Integrating Jira with Subversion ................................................................. 686
Integrating Jira with Perforce ................................................................. 686
Integrating Jira with ClearCase ................................................................. 687
Integrating with a Build Management System ......................................... 687
Integrating Jira with Bamboo ................................................................. 687
Configuring Global Settings .................................................................... 687
Configuring Time Tracking ........................................................................ 687
Configuring Jira Options .......................................................................... 690
Configuring Advanced Settings ............................................................... 696
Setting Properties and Options on Startup ............................................... 698
Recognized System Properties for Jira ..................................................... 703
Advanced Jira Configuration ..................................................................... 705
Changing the constraints on historical time parameters in gadgets .......... 707
Changing the Default Order for Comments from Ascending to Descending 707
Limiting the number of issues returned from a search view such as an RSS feed 707
Configuring File Attachments ................................................................... 708
Configuring Issue Cloning ........................................................................ 713
Configuring Issue Linking ........................................................................ 713
Configuring the Whitelist ......................................................................... 715
Configuring Sub-tasks ............................................................................. 717
Managing Shared Filters .......................................................................... 719
Managing Shared Dashboards .................................................................... 720
Linking to Another Application ................................................................. 722
Server Administration ................................................................................ 723
Increasing Jira Memory ............................................................................ 723
Using the Database Integrity Checker ......................................................... 728
Precompiling Jsp Pages ............................................................................ 729
Logging and Profiling ............................................................................. 730
Logging email protocol details .................................................................. 733
Restoring Data .......................................................................................... 735
Restoring a Project from Backup .............................................................. 737
Optimizing Performance ........................................................................... 745
Backing Up Data ...................................................................................... 745
Automating Jira Backups ......................................................................... 747
Preventing users from accessing Jira during backups ............................. 749
Search Indexing ......................................................................................... 750
Re-Indexing after Major Configuration Changes .................................... 751
Using robots.txt to hide from Search Engines ........................................ 752
Updating your Jira License Details .......................................................... 752
Viewing your System Information ............................................................ 753
Monitoring Database Connection Usage .................................................. 759
Viewing Jira's Instrumentation Statistics ................................................ 761
Generating a thread dump ...................................................................... 764
Finding the Jira Support Entitlement Number (Sen) ............................... 768
Performance Testing Scripts ..................................................................... 769
Appendix A - Extending Jira ..................................................................... 778
Managing Add-ons .................................................................................. 780
Using the Issue Collector .......................................................................... 781
Advanced use of the Jira Issue Collector ................................................ 785
Listeners .................................................................................................. 796
Services .................................................................................................... 798
Jelly Tags .................................................................................................. 800
Jira Toolkit (Customer Support Extensions) ............................................. 835

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer Guides</td>
<td>835</td>
</tr>
<tr>
<td>Building JIRA from Source</td>
<td>835</td>
</tr>
<tr>
<td>How to Make a JIRA Patch</td>
<td>840</td>
</tr>
<tr>
<td>API Documentation</td>
<td>841</td>
</tr>
<tr>
<td>Managing Webhooks</td>
<td>842</td>
</tr>
<tr>
<td>JIRA Installation and Upgrade Guide</td>
<td>844</td>
</tr>
<tr>
<td>JIRA Requirements</td>
<td>844</td>
</tr>
<tr>
<td>Installing Java</td>
<td>847</td>
</tr>
<tr>
<td>Supported Platforms</td>
<td>849</td>
</tr>
<tr>
<td>End of Support Announcements for JIRA</td>
<td>853</td>
</tr>
<tr>
<td>Caveats in using Firefox 3.6.0 with JIRA</td>
<td>860</td>
</tr>
<tr>
<td>Installing JIRA</td>
<td>861</td>
</tr>
<tr>
<td>Installing JIRA on Windows</td>
<td>861</td>
</tr>
<tr>
<td>Uninstalling JIRA from Windows</td>
<td>864</td>
</tr>
<tr>
<td>Installing JIRA on Linux</td>
<td>864</td>
</tr>
<tr>
<td>Uninstalling JIRA from Linux</td>
<td>868</td>
</tr>
<tr>
<td>Installing JIRA from an Archive File on Windows, Linux or Solaris</td>
<td>868</td>
</tr>
<tr>
<td>Installing JIRA WAR</td>
<td>870</td>
</tr>
<tr>
<td>JIRA WAR Configuration Overview</td>
<td>871</td>
</tr>
<tr>
<td>Installing JIRA on Tomcat 6.0 or 7.0</td>
<td>874</td>
</tr>
<tr>
<td>Switching Application Servers to Apache Tomcat</td>
<td>880</td>
</tr>
<tr>
<td>Deploying Multiple Atlassian Applications in a Single Tomcat Container</td>
<td>881</td>
</tr>
<tr>
<td>Configuring Your JIRA Installation</td>
<td>881</td>
</tr>
<tr>
<td>Using the JIRA Configuration Tool</td>
<td>881</td>
</tr>
<tr>
<td>Running JIRA as a Service</td>
<td>886</td>
</tr>
<tr>
<td>Starting JIRA Automatically on Linux</td>
<td>892</td>
</tr>
<tr>
<td>Starting JIRA automatically on FreeBSD</td>
<td>895</td>
</tr>
<tr>
<td>Running the Setup Wizard</td>
<td>896</td>
</tr>
<tr>
<td>Next Steps - Creating a Project</td>
<td>903</td>
</tr>
<tr>
<td>Next Steps - Adding Users</td>
<td>908</td>
</tr>
<tr>
<td>Next Steps - Creating an Issue</td>
<td>913</td>
</tr>
<tr>
<td>Connecting JIRA to a Database</td>
<td>914</td>
</tr>
<tr>
<td>Connecting JIRA to PostgreSQL</td>
<td>915</td>
</tr>
<tr>
<td>Connecting JIRA to MySQL</td>
<td>919</td>
</tr>
<tr>
<td>Connecting JIRA to Oracle</td>
<td>925</td>
</tr>
<tr>
<td>Connecting JIRA to SQL Server 2005</td>
<td>929</td>
</tr>
<tr>
<td>Connecting JIRA to SQL Server 2008</td>
<td>933</td>
</tr>
<tr>
<td>Connecting JIRA to HSQLDB</td>
<td>939</td>
</tr>
<tr>
<td>Tuning Database Connections</td>
<td>943</td>
</tr>
<tr>
<td>Surviving Connection Closures</td>
<td>951</td>
</tr>
<tr>
<td>Switching Databases</td>
<td>952</td>
</tr>
<tr>
<td>Upgrading JIRA</td>
<td>954</td>
</tr>
<tr>
<td>Upgrading JIRA Manually</td>
<td>955</td>
</tr>
<tr>
<td>Upgrading JIRA with a Fallback Method</td>
<td>961</td>
</tr>
<tr>
<td>Upgrading JIRA Using a Rapid Upgrade Method</td>
<td>965</td>
</tr>
<tr>
<td>Skipping Major Versions When Upgrading JIRA</td>
<td>968</td>
</tr>
<tr>
<td>Disabling Auto-Export</td>
<td>968</td>
</tr>
<tr>
<td>Rolling Back a JIRA Upgrade</td>
<td>969</td>
</tr>
<tr>
<td>Migrating JIRA to Another Server</td>
<td>970</td>
</tr>
<tr>
<td>Migrating from JIRA OnDemand to a JIRA Installed Site</td>
<td>976</td>
</tr>
<tr>
<td>Establishing Staging Server Environments for JIRA</td>
<td>979</td>
</tr>
<tr>
<td>Important Directories and Files</td>
<td>981</td>
</tr>
<tr>
<td>JIRA Installation Directory</td>
<td>985</td>
</tr>
<tr>
<td>JIRA Home Directory</td>
<td>986</td>
</tr>
<tr>
<td>Setting your JIRA Home Directory</td>
<td>987</td>
</tr>
<tr>
<td>Tomcat security best practices</td>
<td>989</td>
</tr>
<tr>
<td>Customizing Your JIRA Installation</td>
<td>993</td>
</tr>
<tr>
<td>Changing JIRA's TCP Ports</td>
<td>993</td>
</tr>
<tr>
<td>Running JIRA over SSL or HTTPS</td>
<td>995</td>
</tr>
<tr>
<td>Installing Confluence and JIRA Together</td>
<td>100</td>
</tr>
<tr>
<td>Integrating JIRA with a Web Server</td>
<td>100</td>
</tr>
<tr>
<td>Integrating JIRA with IIS</td>
<td>100</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Integrating JIRA with Apache</td>
<td>101</td>
</tr>
<tr>
<td>Securing JIRA with Apache HTTP Server</td>
<td>102</td>
</tr>
<tr>
<td>Using Apache to Limit Access to the JIRA Administration Interface</td>
<td>102</td>
</tr>
<tr>
<td>Using Fail2Ban to limit login attempts</td>
<td>104</td>
</tr>
<tr>
<td>Deployment Planning Activity</td>
<td>104</td>
</tr>
<tr>
<td>JIRA Resources</td>
<td>104</td>
</tr>
<tr>
<td>Support Policies</td>
<td>104</td>
</tr>
<tr>
<td>Bug Fixing Policy</td>
<td>104</td>
</tr>
<tr>
<td>How to Report a Security Issue</td>
<td>104</td>
</tr>
<tr>
<td>New Features Policy</td>
<td>104</td>
</tr>
<tr>
<td>Patch Policy</td>
<td>104</td>
</tr>
<tr>
<td>Security Advisory Publishing Policy</td>
<td>104</td>
</tr>
<tr>
<td>Security Patch Policy</td>
<td>105</td>
</tr>
<tr>
<td>Severity Levels for Security Issues</td>
<td>105</td>
</tr>
<tr>
<td>Local JIRA documentation</td>
<td>105</td>
</tr>
<tr>
<td>JIRA FAQ</td>
<td>105</td>
</tr>
<tr>
<td>How JIRA Documentation Updates are Published</td>
<td>105</td>
</tr>
<tr>
<td>JIRA Administrators FAQ</td>
<td>105</td>
</tr>
<tr>
<td>Performance FAQ</td>
<td>106</td>
</tr>
<tr>
<td>Is your JIRA Running Slowly</td>
<td>106</td>
</tr>
<tr>
<td>Usage FAQ</td>
<td>106</td>
</tr>
<tr>
<td>Modifying the JIRA Footer</td>
<td>106</td>
</tr>
<tr>
<td>‘Road Map’, ‘Change Log’, and ‘Versions’ Project Tabs Are Not Visible</td>
<td>106</td>
</tr>
<tr>
<td>Why Do Linked Issues in JIRA Appear with a Strike-Through</td>
<td>106</td>
</tr>
<tr>
<td>How to Enable the FishEye Plugin from the Plugin Administration Screen</td>
<td>106</td>
</tr>
<tr>
<td>How to Add the Priority Field into the Email Subject</td>
<td>106</td>
</tr>
<tr>
<td>How to Change the Number of Rows Allowed in the Text Type Custom Field Renderer</td>
<td>106</td>
</tr>
<tr>
<td>How to Make a Federated JIRA Instance</td>
<td>106</td>
</tr>
<tr>
<td>How to Remove ‘Unknown’ Option from ‘Component’ and ‘Fix Versions’</td>
<td>106</td>
</tr>
<tr>
<td>Automatic Escalation of issues</td>
<td>106</td>
</tr>
<tr>
<td>How to Activate Header Row for Subtask List in Issue Detail View</td>
<td>106</td>
</tr>
<tr>
<td>How to Limit the Number of Characters Entered in a Summary Field</td>
<td>106</td>
</tr>
<tr>
<td>JIRA's Timestamp Doesn't Match the System Time</td>
<td>106</td>
</tr>
<tr>
<td>How to Attach a File During Issue Creation</td>
<td>106</td>
</tr>
<tr>
<td>How to Remove Duplicate Entries for Names and Groups</td>
<td>106</td>
</tr>
<tr>
<td>How to Export Users to CSV from JIRA</td>
<td>106</td>
</tr>
<tr>
<td>How Come JIRA does Not Show Direct SQL Data Modifications</td>
<td>107</td>
</tr>
<tr>
<td>How to get JIRA Pages to Render when URL Contains an Underscore</td>
<td>107</td>
</tr>
<tr>
<td>How to disable or enable the GOT FEEDBACK button</td>
<td>107</td>
</tr>
<tr>
<td>Reduce the number of users synchronised from LDAP to JIRA</td>
<td>107</td>
</tr>
<tr>
<td>How do I reduce my user count in JIRA</td>
<td>107</td>
</tr>
<tr>
<td>How to Bulk Edit Groups in JIRA</td>
<td>107</td>
</tr>
<tr>
<td>How to enable &quot;Attach Screenshot&quot; on Linux Machine</td>
<td>107</td>
</tr>
<tr>
<td>How to Configure Workflow to allow Certain Group to Perform Certain Workflow Transition</td>
<td>107</td>
</tr>
<tr>
<td>How to Create a FishEye's Changeset or Crucible's Review Link on JIRA's Comment or Description</td>
<td>107</td>
</tr>
<tr>
<td>How to make the 'Assignee' field required when 'Allow unassigned issues' is turn ON in General Configuration?</td>
<td>107</td>
</tr>
<tr>
<td>How to allow users to view sub-task only but not its parent issue</td>
<td>107</td>
</tr>
<tr>
<td>Why user does not get notification emails when he/she included in Notification Scheme?</td>
<td>107</td>
</tr>
<tr>
<td>How to change the Date Time Range Picker format to 24h format</td>
<td>107</td>
</tr>
<tr>
<td>What if you wish your Custom Field Appearing in Issue Navigator but not willing to set the context to All issue types</td>
<td>107</td>
</tr>
<tr>
<td>Why Fix/Affected Version sort differently in issue screen after picking more than one version?</td>
<td>107</td>
</tr>
<tr>
<td>Modify the gadget limit on a dashboard</td>
<td>107</td>
</tr>
<tr>
<td>How to import attachment using CSV</td>
<td>107</td>
</tr>
<tr>
<td>Log work optional view</td>
<td>108</td>
</tr>
<tr>
<td>Receiving Daily Due Date Subscription</td>
<td>108</td>
</tr>
<tr>
<td>How to add comments to Excel export</td>
<td>108</td>
</tr>
<tr>
<td>How to change the Original Step in Workflow Transition</td>
<td>108</td>
</tr>
<tr>
<td>How to Hide the &quot;Configure Fields&quot; Button on the Create Issue Screen</td>
<td>108</td>
</tr>
</tbody>
</table>
How to revert from New Issue Navigator layout to Old layout in JIRA 5.2.x .................................. 108
How to disable the Invite button in JIRA ................................................................. 108
How to display the issue type name in issue navigator column beside the image icon .......... 108
How to Edit Bundled or System Plugins ................................................................. 108
How to auto disable compatibility mode in Internet Explorer ................................. 108
How to attach an attachment in a JIRA issue using REST API ................................. 109
How to restore a single attachment to an issue ...................................................... 109
Why not able to edit assignee in issue screen? ......................................................... 109
How do I turn off notification when new user is created using SOAP API ............... 109
JIRA does not pick up the data after running an UPDATE or DELETE SQL statement ...... 109
Adding custom content to the front page ............................................................... 109
Allow editing of Closed Issues .............................................................................. 109
Allowing users to create issues anonymously ......................................................... 109
Anonymising JIRA Data ......................................................................................... 109
Appending Email Addresses to Comments Made by Anonymous Users when Using a Mail Handler ................................................................. 109
Asking for an attachment on the Create Issue page .............................................. 109
Automatically Populate Timezone from a Created Issue ........................................ 109
Can I store customer details, like company, address and contact information, in JIRA? .... 109
Change JIRA Browser Icon ................................................................................. 109
Changing Custom Field Types ............................................................................ 109
Changing Templates Used by Export to Excel from the Issue Navigator .................... 109
Changing the default attachment size limit ............................................................. 109
Changing the default session timeout .................................................................... 109
Changing the Default Tab Panel from Comments to All ......................................... 109
Changing the Due Date Input Format ................................................................... 109
Changing the Project Key ..................................................................................... 110
Changing the Size and Content of the Components Select List ............................... 110
Changing the Size of the Fix Versions and Affects Versions Select List ..................... 110
Changing the Size of the Text Area Custom Field ................................................. 110
Changing the Temporary Directory ..................................................................... 110
Configuring project specific security ..................................................................... 110
Controlling project visibility .................................................................................. 110
Using Project Level Security with Project Roles .................................................... 110
Using Project Level Security with User Groups ..................................................... 110
Connecting to SSL services ................................................................................... 111
Creating an Unassigned Issue .............................................................................. 111
Creating Issues via direct HTML links .................................................................. 111
Current Reporter Browse Project Permission ....................................................... 112
CVS ssh Jira Integration ........................................................................................ 112
Disabling Form Token Checking .......................................................................... 112
Displaying a Field Based on Another Field Selection ............................................. 112
Editing a custom field option ................................................................................ 112
Escalating issues (or sending email notifications) when the set turnaround time is exceeded ................................................................................................................. 112
Field Layout Schemes in JIRA 3.x ......................................................................... 112
Fields Allowing Custom HTML or JavaScript ........................................................ 112
Finding the Id for Issue Types .................................................................................... 112
Group Name Guidelines for JIRA ......................................................................... 112
How can I control the editing of issue fields via workflow? .................................. 112
Using a Workflow to control edit of an issue by changing Workflow XML ............. 112
Using Templates to control edit of an issue ............................................................ 112
How do I assign issues to multiple users ............................................................... 113
How do I delete a user account? ............................................................................ 113
How do I disable Firebug for JIRA? ....................................................................... 113
How Do I Use an SSL Certificate Generated Using openssl? .............................. 113
How the CreateOrCommentHandler works? ......................................................... 113
How to change Multi Select Custom field size using script .................................... 113
How to change the location of stdout and stderr logs .......................................... 113
How to clear the resolution field when the issue is reopened ............................... 113
How to configure comment field to become mandatory in workflow transition ....... 113
How to configure sub-task to have a specific screen? ............................................ 113
How to convert types using Jelly ........................................................................... 113
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to create a download link to a file</td>
<td>114</td>
</tr>
<tr>
<td>How to deactivate comments for closed issues</td>
<td>114</td>
</tr>
<tr>
<td>How to disable the Resolve issue screen while resolving issues</td>
<td>114</td>
</tr>
<tr>
<td>How to display a different format for the Number customfield</td>
<td>114</td>
</tr>
<tr>
<td>How to ensure the Road Map tab is visible</td>
<td>114</td>
</tr>
<tr>
<td>How to Get Unicode 'non-ASCII' Characters in HTTPS URL to Appear Correctly</td>
<td>114</td>
</tr>
<tr>
<td>How to have long component names display properly in the Issue Navigator</td>
<td>114</td>
</tr>
<tr>
<td>How to Rename the 'Priority' Field in the Issue Navigator</td>
<td>114</td>
</tr>
<tr>
<td>How to re-order statuses</td>
<td>114</td>
</tr>
<tr>
<td>How to re-order the list of issue operation in an issue</td>
<td>114</td>
</tr>
<tr>
<td>How to resize Free Text Field customfield</td>
<td>114</td>
</tr>
<tr>
<td>How to resize the 'Components' and 'Affects Versions' fields in the Issue Navigator</td>
<td>114</td>
</tr>
<tr>
<td>How to Restrict the Subversion Commits Tab to Selected Projects or Users</td>
<td>114</td>
</tr>
<tr>
<td>How to search by number range in the Issue Navigator</td>
<td>114</td>
</tr>
<tr>
<td>How to show a transition only when the Assignee is different from the Current User</td>
<td>114</td>
</tr>
<tr>
<td>How to update custom field values during workflow transition</td>
<td>115</td>
</tr>
<tr>
<td>Importing data</td>
<td>115</td>
</tr>
<tr>
<td>Importing user from LDAP</td>
<td>115</td>
</tr>
<tr>
<td>Neat JIRA LDAP tricks</td>
<td>115</td>
</tr>
<tr>
<td>JIRA as a Support System</td>
<td>115</td>
</tr>
<tr>
<td>Jelly Escalation</td>
<td>115</td>
</tr>
<tr>
<td>Simple Escalation</td>
<td>116</td>
</tr>
<tr>
<td>Letting customers only create issues</td>
<td>116</td>
</tr>
<tr>
<td>Linking to local file under Firefox</td>
<td>116</td>
</tr>
<tr>
<td>Login problems</td>
<td>116</td>
</tr>
<tr>
<td>Mail error - Unable to relay</td>
<td>116</td>
</tr>
<tr>
<td>Making JIRA login case insensitive for JIRA 3.13.x</td>
<td>116</td>
</tr>
<tr>
<td>Outward Link Description and Inward Link Description</td>
<td>116</td>
</tr>
<tr>
<td>Parsing utf-7 emails</td>
<td>116</td>
</tr>
<tr>
<td>Project-specific email templates</td>
<td>116</td>
</tr>
<tr>
<td>QuickSearch guesses the issue key prefix (sometimes)</td>
<td>116</td>
</tr>
<tr>
<td>Receiving a Daily Summary of Updated Issues</td>
<td>116</td>
</tr>
<tr>
<td>Receiving Notification for Select Issues or Updates</td>
<td>116</td>
</tr>
<tr>
<td>Removing Commas for Values Held in Number Field Custom Field Type</td>
<td>116</td>
</tr>
<tr>
<td>Removing invalid characters from XML backups</td>
<td>116</td>
</tr>
<tr>
<td>Removing NONE from the Issue Security Drop-Down List</td>
<td>117</td>
</tr>
<tr>
<td>Re-order workflow transactions</td>
<td>117</td>
</tr>
<tr>
<td>Resolved issues appearing in Open issues filters</td>
<td>117</td>
</tr>
<tr>
<td>Restricting the Visibility of Worklog on an Issue</td>
<td>117</td>
</tr>
<tr>
<td>Retrieving the JIRA Administrator</td>
<td>117</td>
</tr>
<tr>
<td>Scheme Entity Relations Map</td>
<td>117</td>
</tr>
<tr>
<td>Sending JIRA Data to Support</td>
<td>117</td>
</tr>
<tr>
<td>Setting Additional Fields for Issues Created from Email</td>
<td>117</td>
</tr>
<tr>
<td>Setting a Default Value in the Description Field</td>
<td>118</td>
</tr>
<tr>
<td>Setting Priority field value based on customfield value</td>
<td>118</td>
</tr>
<tr>
<td>Showing Extended Timestamp in the Created Column of the Issue Navigator</td>
<td>118</td>
</tr>
<tr>
<td>Single Sign-on</td>
<td>118</td>
</tr>
<tr>
<td>Tracking the Time Taken for Each Workflow Transition</td>
<td>118</td>
</tr>
<tr>
<td>Troubleshooting Issue Creation Via Email</td>
<td>118</td>
</tr>
<tr>
<td>Using JIRA to Manage reusable modules</td>
<td>118</td>
</tr>
<tr>
<td>We already have users &amp; groups defined elsewhere - can JIRA make use of these?</td>
<td>118</td>
</tr>
<tr>
<td>Where are the application server logs?</td>
<td>118</td>
</tr>
<tr>
<td>Why doesn't JIRA have a Severity field like Bugzilla?</td>
<td>118</td>
</tr>
<tr>
<td>Workflows Guidebook</td>
<td>118</td>
</tr>
<tr>
<td>XML format for import &amp; export files</td>
<td>118</td>
</tr>
<tr>
<td>Installation Notes</td>
<td>118</td>
</tr>
<tr>
<td>Configuring IIS with Tomcat</td>
<td>118</td>
</tr>
<tr>
<td>Database Notes</td>
<td>118</td>
</tr>
<tr>
<td>Incorrect database type specified</td>
<td>118</td>
</tr>
<tr>
<td>Restarting JIRA from the Setup Wizard</td>
<td>119</td>
</tr>
<tr>
<td>Database limitations on number of projects</td>
<td>119</td>
</tr>
<tr>
<td>JIRA and HSQL</td>
<td>119</td>
</tr>
</tbody>
</table>
JIRA Documentation

Get started

JIRA 101 guide
The complete JIRA user's guide
Install JIRA
Create a project
Create an issue
Find issues
Gather information on a project

Are you evaluating JIRA for your team? Check out the JIRA 101 guide.

Administer JIRA

The complete JIRA administrator's guide
Migrate from another issue tracker
Add more users
Customize screens
Customize fields
Customize workflows

Need more resources? Check out the JIRA Resources page.
Do more with JIRA

- Use JIRA in an enterprise environment
- Install add-ons for JIRA
- Make JIRA agile with JIRA Agile
- Test web applications with JIRA Capture
- Contribute to the JIRA Documentation

Do you want to customize JIRA for your team? Check out the documentation for developers.

What’s new in JIRA? »

JIRA 101

Welcome to the JIRA 101 guide! This is a hands-on tutorial. You'll be setting up your own JIRA site and trying it out, as we take you on a tour of the basic concepts in JIRA. Along the way, we will highlight some of the key JIRA features and point you to a number of helpful resources.

Roll up your sleeves. We’re about to get started!

- Wait! I don’t know what JIRA is...
  - Watch this video for an overview of JIRA in 4 minutes:

Using this tutorial

This tutorial is designed for someone who is evaluating JIRA for purchase. However, you can still use this tutorial if you want to learn the basic concepts in JIRA. Throughout this tutorial we will highlight whether you need to be an administrator to complete a stage, or whether it is available to all users.

- You must be an administrator to complete this stage.

- Anyone can complete this stage.

We recommend that you do each of the stages in order. It will take about an hour to complete this entire tutorial. However, you can stop after any stage and resume it later, if you don’t have time to complete the tutorial in one
Other resources

If you are already comfortable with the basic JIRA concepts, try these other resources:

- **Atlassian Quickstart**: Want to track bugs? Manage an Agile development project? Gather customer feedback? These short guides show you how to use JIRA for **specific use cases**.
- **Atlassian University**: Atlassian University is a paid learning resource for you and your organisation. Each course uses hands-on interactive tutorials to walk you through **key functionality** in JIRA, like inline editing or issue filters.
- **Atlassian Answers**: Got a question about JIRA? Atlassian Answers has a large community of users that can help you find an answer. Atlassian staff are also active on Atlassian Answers.

So what is a 101 anyway?

*In US universities a "101" course is a general course that is often used as an introduction to a particular subject area - it covers the basics, and gives you a taste of what you're in for... just like this tutorial!*

Set up a JIRA site

You will need your own JIRA instance for this tutorial. Let's get you set up with a **JIRA OnDemand** site. JIRA OnDemand is our hosted JIRA offering. You will be up and running with your own JIRA instance in a few minutes without installing a thing! If you have a JIRA site already, you can skip this stage.

**STEP 1 OF 8**

**Step 1. Sign up for a JIRA OnDemand site**

Signing up for **JIRA OnDemand** will provide you with a fully-functional JIRA OnDemand site for one month. The JIRA OnDemand signup process requires a credit card. Don't be alarmed though, during the one month trial you will not be charged anything and can cancel at any time.

2. Read the 'JIRA Trial FAQ'. Don't skip this step, the FAQ has important information about what you are signing up for.
   You can find the link under the 'OnDemand' and 'Download' panels, if the FAQ is not showing.
3. Click **Start My Free Trial** for OnDemand.
4. If JIRA Agile is selected, click the x (cross symbol) to deselect it. Click Next.
5. Fill in the form and submit it. Note down the site address and site administrator details.

Can't use JIRA OnDemand?
If you cannot use JIRA OnDemand, instructions for installing JIRA on your own server are available below.

- Installing JIRA on Windows
- Installing JIRA on Linux
- Installing JIRA on Mac OS X
  (note, Mac OS X recommended for evaluations only)

Tips for filling in your JIRA OnDemand form:

- Don’t add any other applications. You won’t need them for this tutorial and you can always add them later.
- Stick with 10 users for now. You won’t need more than 10 users for this tutorial and you can always add more later.
- You can choose to keep your site after your trial. Keep this in mind when picking a site address.
- If you don’t have a my.atlassian.com account, you will set one up during the signup process. This account is used to manage your licenses for Atlassian products. It is different to the administrator account that you also create during signup: the administrator account is used to access your new JIRA OnDemand site.

That's it! Grab a quick coffee — it will take about 10 minutes for your JIRA OnDemand instance to be provisioned. You will receive an email when your JIRA instance is ready.

Step 2. Create the demonstration project

Access your new site via your browser, using the site address that you entered when signing up (e.g. https://jira101.atlassian.net). Log in to your new site using the site administrator credentials that you entered when signing up (e.g. jsmith).

This is the first screen that you will see when accessing your new JIRA OnDemand site:

You will see this dialog whenever you create a new project. This allows you to set up different types of projects with a single click, including Scrum and Kanban.

Choose Demo Project and click Next. This will create a new project that has sample data. We won’t be using the Demo Project for this tutorial, however the issues in the project contain handy information about JIRA. Take a few minutes to browse through them.

Note, if you have already created projects in your JIRA site, the Demo Project won’t be available. Don’t worry, you won’t need it for this tutorial.
Congratulations! You now have a new JIRA OnDemand site set up with the sample 'Demonstration' project.

Next steps

Next you will be adding users to your new JIRA site and learning about how JIRA’s licensing works.

Click here? Please leave us your feedback before you go!

Click to give feedback...

Click to access the survey (you will be redirected to the Wufoo site): http://atlassian.wufoo.com/forms/jira-101-feedback/

Add users

It will get pretty lonely if you are the only JIRA user. You can add users, invite users, or allow new users to sign themselves up. Get your team involved!

In this stage, you will add new users to JIRA. We will also explain JIRA user-based licensing and show you how it works.

STEP 2 OF 8

Step 1. Add a few users

You will be adding three users to JIRA in this step: alana, emma and william. You add more or choose your own usernames if you like, but please note that we will be referring to these usernames later in the tutorial.

You should be logged in to JIRA as an administrator after setting up your site. If not, log in under your administrator account.

1. Navigate to the User Management screen using the quick operation: . + users.
2. Choose **Create User** to add a new user. Specify the username as **alana** and tick the **Add to developers group** checkbox. Set the rest of the fields to whatever you want.

3. Create two more users with the usernames **emma** and **william**, following the same process outlined in previous step. Don't forget to tick the **Add to developers group** checkbox.

---

**How do quick operations work?**

"Use the quick operation: . + abc"  
This means press the . key on your keyboard, type abc in the dialog that appears, then press the Enter key on your keyboard.

---

**Step 2. Remove one user from your license**

JIRA has user-based licensing. This allows you to manage your costs as your team grows or shrinks, by adding or removing users from your license.

Every user that can access JIRA is counted towards your license. New users are granted access by default. If you exceed the number of users allowable by your license, no one will be allowed to create issues in JIRA.

The user count is managed in JIRA OnDemand by configuring the **application access** for each user. In this step, we will show you how to remove one user's access to JIRA.

1. Navigate to the 'Application Access' screen using the quick operation: . + application access. The screen will show that you have six remaining users available under your license.

   Your trial OnDemand license permits 10 users for JIRA. You have created four users so far: your administrator account and accounts for Alana, Emma and William.

2. Uncheck the checkbox in the JIRA column for **emma** and choose the **Update** button. The screen will show that you have seven remaining users available under your license.

Emma will not be able to log in to JIRA anymore. Feel free to try, if you want to check.
Congratulations! You have added three new users to JIRA and removed JIRA access from one!

Next steps

Next you will be creating your own project in JIRA and customizing the look and feel.

Stopping here? Please leave us your feedback before you go!

Click to give feedback...

Click to access the survey (you will be redirected to the Wufoo site): http://atlassian.wufoo.com/forms/jira-101-feedback/

Create a project

In this stage, you will create a project. A JIRA project is a collection of issues. Your team could use a JIRA project to coordinate the development of a product, track a project, manage a help desk, and more, depending on your requirements.

You will also be customizing the JIRA look and feel in this stage. We have provided sample data below for you to use. You can use your own data and pictures, but the screenshots in this tutorial will show the sample data.

What is JIRA Junior?

You will notice that the sample data in this tutorial refers to the "JIRA Junior" product:

We invented JIRA Junior as an April's Fools joke. Sadly, it is not a real product.
Step 1. Create a project

In this step, you will be creating a project for your team to use. You should still be logged in to JIRA as an administrator, from the previous stage. If not, log in under that administrator account.

1. Choose Projects > Create Project and choose Blank Project.
2. Enter JIRA Junior as the project name. Leave the project key set to its default value of JJ.
3. Choose Submit to create your new project.

The project for "JIRA Junior" will be created and you will be shown the project summary page.

Looking good! Let's make a few tweaks to really make your new project stand out.

About project keys

Each project has a unique name (e.g. JIRA Junior) and a unique key (e.g. JJ). The project key becomes the first part of that project's issue keys, e.g. JJ-1, JJ-2, etc.

Step 2. customize your project

In this step, you will be customizing the project details and logo for your new project. Customizing your project helps your users to identify it more easily, particularly if you have many projects in your JIRA site.

You should be on the project summary page for the JIRA Junior project from the previous step, if not go to Projects > JIRA Junior (JJ).

1. Choose the Administration tab > Edit Project button.
2. Click the Project Avatar image and choose the Choose file button on the dialog that appears.
3. Upload an image for the project avatar. Here's a sample image that you can use:

   ![JIRA Junior](https://example.com/jira-avatar.png)

4. Add a few more details, like a URL and a Description. These fields are only for display, so you can enter anything you want.
5. Choose Update to save your changes.

We are not done yet. Let's polish up your site and user profile, so you can see how easy it is to customize the look and feel of JIRA.

Step 2. customize your site

JIRA gives you control over the look and feel of your entire site. You can change the color scheme, logo,
favicon, starting page (default dashboard) and more, to enhance the branding of your organisation or make it consistent with other systems.

In this step, you will be changing the site logo and the color scheme for your JIRA site.

1. Navigate to the Look and Feel configuration for your JIRA site using the quick operation: . + look and feel .
2. In the Logo section > Upload from File field, choose the Choose File button.
3. Choose an image for your JIRA logo in the dialog that appears. Here’s a sample image that you can use:

![Attassiah Jr.](image)

4. Choose the Upload Logo button on the Look and Feel screen. Your logo will be uploaded. Note that the color scheme for your site will also be automatically changed to match the colors of your logo.

About the color scheme
You can also manually change the colors of different elements of JIRA in the Colors section of the Look and Feel screen, if you don’t like the automatic color scheme.

Step 3. Update your user avatar

Now that your site is branded, let’s get you a better picture for your user avatar. JIRA has a set of default avatar images to choose from, but in this step, you will be adding your own avatar picture.

Adding an avatar picture helps other users identify you at a glance, when you interact with JIRA. For example, when you comment on an issue, your avatar picture will display next to your comment.

1. Choose the user dropdown in the JIRA header. It will look like this: , Choose Profile from the menu.
2. Click the Avatar image and choose the Choose file button on the dialog that appears.
3. Upload an image for your avatar. You will be able to crop it before it is resized for you. Here’s a sample image that you can use:

![Avatar](image)

4. Navigate back to the JIRA Junior project summary to see all of your new changes: Projects > JIRA Junior (JJ).

Gravatars for user avatars
JIRA also supports Gravatars for user avatars. If you enable Gravatar support for your site, JIRA will automatically use the Gravatar associated with each user’s email address in JIRA.

✔ Congratulations! You have created a new project and customized the look and feel of JIRA.
Next steps

Next you will learn how to create an issue in JIRA.

Stopping here? Please leave us your feedback before you go!

Click to give feedback...

Click to access the survey (you will be redirected to the Wufoo site): http://atlassian.wufoo.com/forms/jira-101-feedback/

Create an issue

An issue is the most basic entity in JIRA. Issues can represent different things depending on your team and project. For example, your development team could use issues to represent feature stories, development tasks and bugs for a product.

In this stage, you will create an issue. We will show you how to manage this issue through its lifecycle and how you can involve your team members along the way.

STEP 4 OF 8

Step 1. Create an issue

Let's start by creating your first issue.
1. Choose Create issue in the JIRA header to open the 'Create Issue' dialog.
2. Fill out the fields using the sample data is shown below. Only the fields with * are mandatory.
   - Project: JIRA Junior
   - Issue Type: Story
   - Summary: As a kid, I want to have fun creating issues
   - Description: Make everything fun!
   - Leave all other fields blank or at their default values.
3. Choose Create to create your new issue. A confirmation message will display for a few seconds. Note, the issue key of your new issue: JJ-1.

Create another issue with an Issue Type of Bug and Summary of JIRA Junior logo is not pink enough. Set the rest of the fields to anything you like.
Step 2. Edit the issue

In this step, we will show you how to find an existing issue using the quick search, and how to edit it using inline editing. For example, you may want to add more information to an issue, attach new files or screenshots, and more.

Inline editing is the quickest way to edit an issue. However, please note that blank fields are hidden on the View Issue screen. You can only edit these via the Edit Issue dialog (Edit button on View Issue screen), which provides you access to all issue fields.

1. In the Quick Search box in the JIRA header, type JJ-1 and press Enter on your keyboard. The first issue that you created will display. The quick search is only one of the search methods in JIRA. We will cover the others in the next stage.

2. Hover over the Priority field. A pencil icon will appear with a Click to edit label.
3. Click the Priority field and change it to Critical. A tick and cross icon will appear.
4. Click on the tick icon or anywhere outside of the field to save your change.

Step 3. Resolve the issue

Every issue has a lifecycle. In JIRA, the lifecycle of an issue is managed by a workflow. A workflow consists of the issue statuses (e.g. Open) and the transitions between each status (e.g. Start progress). Here is the default workflow used in JIRA:

In this step, you will be resolving your issue. To do this, you will use the Start Progress transition to take your issue from Open to In Progress, then the Resolve transition to take your issue from In Progress to Resolved.

You should be viewing the issue JJ-1. If not, open it using the Quick Search. You will notice that all of the workflow-related controls are grouped together:

1. Choose the Start Progress. The status of your issue will be changed to In Progress.
3. Leave the fields at their default values and choose the Resolve. The status of your issue will be changed to Resolved.
Notice that resolving an issue prompted you to enter more information, whereas starting progress on the issue did not. Some issue transitions have screens associated with them in the default JIRA workflow. JIRA allows you to prompt the user with screens for any transition, if you configure your own workflow.

Step 4. Assign the issue to another user

You won’t be managing every issue in your project, if you are using JIRA for your team. Let’s get another user involved by assigning them an issue.

1. Find and open the issue JJ-2 by entering JJ-2 in the Quick Search box and pressing Enter on your keyboard.
2. Choose Assign on the issue. You can also use the quick operation: + assign
3. Type alana in the Assignee field and select her as the assignee from the dropdown list that appears.
4. Type (don’t copy and paste) the following text in the Comment field, then choose Assign.

```
Hi @william JIRA Junior is *super fun* now (see JJ-1). Alana is going to fix the logo.
```

You will notice a few things when you enter the comment:
• When you start typing after the @ symbol, you will be prompted to choose a user: william, in this example. William will be sent an email notification that links to the issue, when you save. This feature is called mentioning a user.
• On choosing Assign:
  • The issue is assigned to Alana with a comment added to it.
  • A link is automatically created to the issue JJ-1 in the issue comment.
  • super fun is shown as bold text in the issue comment. If you edit an issue description or comment and click the 💡 icon, you will see other text formatting options.

Congratulations! You have learned the basics of working with an issue.

Next steps

Next you will learn how to search for issues and create reports.

Stopping here? Please leave us your feedback before you go!

Click to give feedback...

Click to access the survey (you will be redirected to the Wufoo site): http://atlassian.wufoo.com/forms/jira-101-feedback/
Search for issues and create reports

Knowing how to create an issue is important, but your team is going to be working with more than one issue! You will need to know how your team is tracking, as well as help them stay on top of their backlog.

In this stage, we will show you how to work with multiple issues. You will learn how to use different search techniques to find issues. We will also show you how to share search results with your team and report on issues.

STEP 5 OF 8

Before you start, you are going to need a few more issues. Create a few more in your JIRA Junior project, using the sample data below.

Tip: Tick the Create another checkbox before you click Create, when you need to create multiple issues.

- **Issue Type** = Story and **Summary** = As a kid, I want to hide things from dad and mum
- **Issue Type** = Bug and **Summary** = Can't set priority to Super Duper
- **Issue Type** = Story, **Summary** = As a kid, I want to share issues with my friends and **Assignee** = alana
- **Issue Type** = Story, **Summary** = As a kid, I want to groom my backlog and **Assignee** = william

Step 1. Search for issues

In this example, we are going tackle a common scenario: searching for all unresolved issues assigned to you. You might regularly run a search like this to check your backlog of work.

1. Choose **Issues > Search for Issues**. You should see all 13 issues (including issues from the Demo project) in your JIRA site.
2. Set **Assignee = Current User** in the search criteria.

   ![Search for Issues](image)

   Notice that the search results refresh when you select new criteria.

3. Choose **More > type Resolution** then select it.

   ![Resolution](image)

4. Set **Resolution = Unresolved**. The search results will show the nine issues that are unresolved and assigned to you.
About the advanced search (JQL)

Power users may like to try the advanced search (similar to SQL), by clicking the Advanced link next to the search criteria. For example, enter `assignee = currentUser()` to find all issues assigned to you.

Advanced Searching

Learn more...

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. For example, the following simple query will find all issues in the "JIRA Junior" project:

```
project = "JIRA Junior"
```

(This example uses the Project field, the EQUALS operator, and the value "JIRA Junior").

If you are thinking that it would be handy to be able to rerun this search, we have got you covered! Hover over the icon in the top right and choose My Open Issues. Keep this screen open for the next step.

Saved searches, like this default one, are referred to as filters. You can also save your own searches instead of choosing from the default ones. We are just about to tackle this: see the next step below.

Step 2. Save your search

If you run a search with the same criteria frequently, you may want to save it as a filter. This lets you run the search again with a single click, rather than selecting the same criteria every time. For example, you may use a filter to review your open tasks for the day.

In this step, you will find all stories in the JIRA Junior project, then save it as a filter.

2. Set Project = JIRA Junior and Type = Story as the criteria. You should have four issues in your results: JJ-1, JJ-3, JJ-5 and JJ-6.
3. Choose Save As (above the search criteria), enter JIRA Junior stories as the Filter Name and save it.
That's it! Hover over the icon in the top right. You can see your new filter under the **Favorite Filters** section. Just click it to run it.

Let's look at some of the ways that you can use your new issue filter.

**Step 3. Share your search results**

Getting your team on the same page is easy with shared filters. You could share a filter with your team that shows the unresolved stories for a development iteration, or the critical issues in a support backlog.

Here are two ways that you can share search results:

1. **Email the search results**
   - Run the desired filter, then choose **Share**. Enter the users that you want to share the filter with and they will be emailed a link to your filter (if you have email notifications set up).

   **What does it look like?**

   ![Share Link to Filter]

   Share the search results via a dashboard

   The dashboard is the screen that all JIRA users see when they first log in. You can show a filter's results on a dashboard and share it with other users.

   **Show me how...**

   1. Choose **Dashboards > Manage Dashboards**, then choose **Create new dashboard**.
   2. Name your dashboard **JIRA Junior** and choose the **+Add** button next to **Add Shares** to share it with everyone.
   3. Leave the other fields and choose **Add**.
   4. Choose **JIRA Junior** in the **Favorite Dashboards** section to configure it.
   5. Choose **add a new gadget** to open the 'Gadget Directory'.
   6. Enter filter results in the search box and choose **Add It Now**.
   7. Enter **JIRA Junior stories** in the **Saved filter** field and choose **Save**.

   Other users can now add this dashboard by choosing it as a favourite.

**Step 4. Create a report**
JIRA provides you with a number of different reports to help you track your team's progress. You can view the time tracking data, time taken to resolve issues, workload by user, and more.

In this example, you will create a simple report based on the ‘JIRA Junior stories’ filter that you created previously. The report will show all JIRA Junior stories by assignee.

1. Choose **Projects > JIRA Junior**.
2. On the project summary page, choose **Single Level Group By Report** in the **Reports** section (you may need to scroll down).
3. Set **Filter** to **JIRA Junior stories**. Leave **Statistic Type** set to **Assignee**.
4. Choose **Next**. The 'Single Level Group By Report' for the 'JIRA Junior stories' filter will be displayed.

Try creating a few other reports in the **Reports** section of the JIRA Junior project summary.

![Single Level Group By Report](image)

**Congratulations! You have created an issue filter, shared it and used it in a report.**

**Next steps**

Next you will learn how to configure a project.  

**Stopping here? Please leave us your feedback before you go!**

- **Click to give feedback...**

  Click to access the survey (you will be redirected to the Wufoo site):  

**Configure permissions**

You won't want every user in your team to have the same level of access to JIRA. For example, you may want to restrict who can administer JIRA, or prevent users from viewing a project. In this stage, you will learn about the different permissions in JIRA and try setting permissions for a project.

![Configure permissions](image)

**STEP 6 OF 8**

Before we start, it is important to understand how JIRA permissions work. There are two levels of permissions in JIRA: global permissions and project permissions.

**Global permissions** cover a small set of functions that affect all projects in JIRA. For example, permission to administer JIRA. They can only be assigned to groups.

- **Show me...**
Project permissions cover a set of more granular functions that affect a single project in JIRA. For example, permission to create issues in a project. They can be assigned to groups, users and roles.

Show me...
1. **About Roles, Groups and Users**

A role is a project-specific set of groups and/or individual users. Roles are a way of abstracting users/groups from project permissions.

*Learn more...*

For example, in your team, you may have 'scrum masters' who are responsible for assigning issues in projects. Your product managers (PM) are 'scrum masters' on every project, but individual developers may also be 'scrum masters' on specific projects.

In JIRA, you can define a 'scrum master' role that includes the PM team on all new projects. You can then define a set of permissions that grant the 'assign issue' permission to this role, and apply this set of permissions to all projects. Individual developers can be added to the 'scrum master' role on each project, as needed.

This is much easier to manage than defining different sets of permissions for each project.

Let's try it out! In the following steps, you will use project permissions to hide a new project from some of your users.

**Step 1. Create a new project role**

This project role will only contain users that you want to view a particular project. We will assign permissions to this role in the next step.

1. Navigate to the 'Project Role Browser' screen using the quick operation: . + roles.
2. Add a new project role: **Big Kids**.
3. Add yourself and Alana to the Managers project role via Manage Default Members > Edit (under Default Users). Do not add William.

**Step 2. Configure a new permission scheme**

The 'Browse Projects' permission controls whether a user can browse a project, i.e. whether they can view the project. You will assign this permission to your new role.

1. Navigate to the 'Permission Schemes' screen using the quick operation: . + permissions schemes.
2. Copy the Default Permission Scheme, then edit the copied scheme and change the name to Confidential Permission Scheme.

3. Click Permissions for Confidential Permission Scheme. For the Browse Projects permission:
   - Choose Delete for 'Project Role (Users)'.
   - Choose Add, select Big Kids in the Project Role field and save.

<table>
<thead>
<tr>
<th>Project Permissions</th>
<th>Users / Groups / Project Roles</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator Projects</td>
<td>Project Role (Administrators)</td>
<td>Add</td>
</tr>
<tr>
<td>Browse Projects</td>
<td>Project Role (Big Kids)</td>
<td>Add</td>
</tr>
<tr>
<td>View Issue Source Tab</td>
<td>Project Role</td>
<td>Add</td>
</tr>
</tbody>
</table>

Step 3. Associate the scheme with a project

Finally, let's associate the permission scheme with a new project.

1. Choose Project > Create Project and choose the Blank Project.
2. Name the project Top Secret Project and Submit.
3. Choose Administration (tab) > Permissions (left menu) > Actions > Use a different scheme.
4. Set the Scheme to Confidential Permission Scheme and Associate.

The only users that can browse your new project are Alana and yourself (note, default members are only added to a role for new projects). Try browsing the project with William. You could also use this approach to restrict users from creating issues, adding comments, closing issues, etc, in a project.

✔ Congratulations! You configured project permissions to hide a project from users.

Next steps

Next you will learn how to edit workflows and screens in JIRA.

Stopping here? Please leave us your feedback before you go!

* Click to give feedback...

Click to access the survey (you will be redirected to the Wufoo site): http://atlassian.wufoo.com/forms/jira-101-feedback/

Configure workflows and screens
Every team works differently. The process for one project may differ from other projects. If the default JIRA workflow, screens and fields don’t suit, you can customize them.

In this stage, you will import a new workflow and configure it with a new screen that has a custom field. We will also explain how workflows, screens and fields relate in JIRA.

**STEP 7 OF 8**

Step 1. Import a workflow

If you don’t want to use the default workflow (described in a previous step) for your project, you can use a custom workflow instead. In this step, we will show you how to import a custom workflow from the Atlassian Marketplace. You could also configure a workflow manually, but we won’t tackle that in this tutorial.

1. In the ‘Top Secret Project’, choose **Administration** (tab) > **Workflows**.
2. Choose **Add Workflow** > **Choose from Marketplace**.
3. Choose the **Import** button for **IT Service Desk Workflow (ITIL)** and complete the wizard (leave all values at defaults).
4. On the ‘Assign Issue Types to “It Service Desk Workflow (itil)”’ dialog, tick the checkbox next to **Issue Type**. All issue types will be selected, just ignore the warning symbols. Choose the **Finish** button.

**What is the Atlassian Marketplace?**

The Atlassian Marketplace hosts over 1,000 add-ons (free and commercial) that can enhance your Atlassian applications. These range from add-ons that provide a suite of new functionality to ones that tweak individual features in your product. For example, the JIRA Agile add-on turns JIRA into an Agile planning tool.

You will see the new workflow on the ‘Workflows’ page in ‘DRAFT’ status. Any time you change a workflow for a project, it will create a draft copy of the workflow. Choose the **Publish** button.
That's it! If you had issues in your project, JIRA would have helped you migrate them to the new workflow. Try creating a new issue and transitioning it through the new workflow to see it in action.

Step 2. Configure a custom field and a screen

Configuring the workflow steps still may not be enough to fit your processes. You may want to prompt users to enter certain information during the lifecycle of an issue. JIRA lets you configure the fields and screens that display during a workflow to do this.

In the previous step, you would have noticed a screen appeared during the 'Triage' transition of your new workflow:

![Screen example](image)

In this step, you are going to replace this screen with your own screen that has a new custom field. Before you try this, it is helpful to understand the relationships between workflow, screens and fields:

Show me...

It looks complicated, but don't be alarmed. You will see how it fits together as you try it out.

1. Navigate to the 'Screens' screen using the quick operation: . + screens.
2. Choose Add Screen.
3. Enter the Name as **Assign Tester Screen**, and click **Add**. Don’t add any fields for now.
4. Navigate to the 'Custom Fields' screen using the quick operation: . + **custom fields**.
5. Choose **Add Custom Field**.
6. Choose the **User Picker** field type, then click **Next**.
7. Enter the **Field Name** as **Tester**. Leave the rest of the fields at their default values (note how you can change the context of the field, if you wanted), and click **Finish**.
8. Associate your new field to the **Assign Tester Screen**, and click **Update**.

You now have a new screen with a new custom field. Let's add it to your project's workflow.

1. Choose **Projects** > **Top Secret Project (TSP)** > **Administration** > **Workflows**
2. Choose the **Edit** button (hover over the workflow diagram to enable it).
3. Hover over the **Triage** box in the diagram, then choose the **cog icon** > **Edit transition**.
4. Set the **Transition View** to **Assign Tester Screen**, then click **OK**.
5. Choose **Publish Draft** to save your changes (no need to save a backup copy).

That’s it! Try creating a new issue and transitioning it through the new workflow. When you triage your issue, you should see this screen:

---

### Triage

**Tester:**

Start typing to get a list of possible matches.

**Comment:**

---

**Using the admin helper to troubleshoot problems**

If you can’t find your custom field on the View, Create or Edit Issue screens, choose **Where is my field?** under **More Actions** or **Configure Fields** and enter your field to find out why it is not displaying.

The admin helper can also help you troubleshoot permissions and notifications (quick operation: . + **permission helper** or . + **notification helper**).

**Congratulations! You have configured your project to use a new workflow and a new screen with a custom field.**

**Next steps**

Next you will learn how to extend JIRA.

**Stopping here? Please leave us your feedback before you go!**

[Click to give feedback...](http://atlassian.wufoo.com/forms/jira-101-feedback/)
Extend JIRA

We are almost there! You have covered all the core concepts of JIRA, however we have only looked at the basics of what you can do with JIRA. In this stage, we will show you how to get the most out of your new JIRA site. This includes using JIRA on your mobile device, installing add-ons and integrating JIRA with other applications.

Some of these features are powerful but complex, so we will only show you how to get started, then point you to detailed documentation elsewhere. By the end of this stage, you should have a better idea of how you can extend JIRA to suit your team's needs.

**STEP 8 OF 8**

Step 1. Use JIRA on the go

Simply browse to the URL for your JIRA OnDemand site, using the mobile browser on your iPhone or an Android phone. JIRA will display the mobile-optimised version of the page.

Step 2. Enable add-ons

Is your team using Scrum or Kanban? Do you manage timesheets and resources for the team? Does your team need better testing tools? Add-ons can help you extend JIRA to fit your needs.

An add-on (or plugin) is an installable component that supplements or enhances the functionality of JIRA in some way. In this step, we will show you how to enable add-ons. Let's start with the JIRA Suite Utilities add-on that adds a range of workflow-related functionality to JIRA.

1. Navigate to the 'Manage add-ons' screen using the quick operation: . + manage add-ons.
2. Choose JIRA Suite Utilities > Enable.

That's it! This is a free add-on, so you don't need a license to use it.

Next, let's add a commercial add-on: Atlassian's JIRA Agile add-on that turns JIRA into an Agile planning tool. Don't worry about the cost, you will get a one-month free trial and can cancel at any time.
Add-ons in JIRA OnDemand

You cannot install add-ons in JIRA OnDemand. However, a set of add-ons are bundled with JIRA OnDemand. You can enable any of these.

If you install your own JIRA instance, you can install any add-on from the Atlassian Marketplace or even build your own!

1. Navigate to the 'Manage add-ons' screen using the quick operation: . + manage add-ons.
2. Choose JIRA Agile > Free Trial.
3. You will be redirected to my.atlassian.com. The 'Configure Your OnDemand Instance' page will show a JIRA Agile license for 10 users. Choose Apply Changes at the bottom of the page to add JIRA Agile to your site.
4. You will be redirected back to your JIRA OnDemand site. JIRA Agile will be added (this may take a few minutes). Choose Get Started on the 'Licensed and Ready to Go' confirmation dialog to try it out when it is ready.

That's it! You may also want to enable other commercial plugins for JIRA, like JIRA Capture (bug reporting) or Tempo (timesheets).

Step 3. Add Confluence to your OnDemand site

JIRA helps your team manage a fundamental part of any project: tracking work. However, your team may also need help managing code, automating builds, organizing documentation, etc. JIRA seamlessly integrates with other Atlassian applications that help solve these problems.

In this step, we will show you how to add Confluence to your OnDemand site. Atlassian's Confluence is a wiki-based tool that helps your team create, share and collaborate on content (e.g. business requirements, software specifications, test plans, etc). Don't worry about the cost, you will get a one-month free trial and can cancel at any time.

1. Log in to your my.atlassian.com account. This is the account that you created in the first stage of this tutorial.
2. Find the Licenses section on the account page and choose the + (plus sign) of your Atlassian OnDemand subscription.
3. In the Actions section, choose Configure Products & Users.
4. Choose Start a Free Trial for Confluence to add a license for 10 users to your OnDemand site, and choose Apply Changes.

You have added Confluence to your OnDemand site (you may need to wait a few minutes). Try accessing it via https://<your_site_name>.atlassian.net/wiki/
Step 4. Integrate JIRA with a source control system

In this step, we will show you how to integrate JIRA OnDemand with Bitbucket: Atlassian's code hosting tool for Git and Mercurial. This will allow you to do things like see code commits in JIRA and move a JIRA issue through a workflow via commit messages. We are just going to show you how to connect the two services to get you started.

1. Log in to your my.atlassian.com account. Start a free trial for Bitbucket, similar to how you started a free trial for Confluence in the previous step.
2. You'll receive an email from Bitbucket inviting you to join the Bitbucket team linked to your JIRA OnDemand. Find it and click Join your new team.
3. On the invitation page, sign up for a new Bitbucket account to join the team (or log in, if you already have a Bitbucket account).
   Your team account URL will be https://bitbucket.org/<ondemand-account_name> or https://bitbucket.org/<ondemand-account_name>-ondemand (e.g. https://bitbucket.org/jirai01).
4. Check that JIRA OnDemand is connected to Bitbucket: In JIRA OnDemand, use the quick operation: dvcs accounts. You should see your Bitbucket account listed on the ‘DVCS Accounts’ screen.

   JIRA and code hosting

   If you have installed JIRA on your own servers, you may also want to consider integrating it with Stash: Atlassian's behind the firewall Git management for your source. Both downloadable JIRA and JIRA OnDemand also integrate with GitHub.

   JIRA OnDemand to Bitbucket are now connected. You will need to do further configuration in Bitbucket to use it...
with JIRA effectively, such as add your team members to your Bitbucket team and create repositories. The full procedure is here: Getting started with Bitbucket and OnDemand (start at step 3).

What next?

So that’s it — we hope this guide has helped you get a feel for JIRA.

Your OnDemand trial is free for one month and then you have a couple of options:

- **Keep this OnDemand instance**
  You don’t need to do anything if you want to keep your OnDemand instance. When the evaluation expires your credit card will be charged and you can continue using your OnDemand site. You can manage your account at my.atlassian.com.

- **Cancel your OnDemand trial**
  Decided not to continue? You can cancel at any time during the first month.
  To cancel your OnDemand trial, log in to your my.atlassian.com account and choose Cancel evaluation for your Atlassian OnDemand Evaluation. This will include the JIRA, JIRA Agile, Confluence and Bitbucket licenses that you set up during this tutorial.

- **Switch to installed JIRA**
  If you want the greater flexibility that comes with hosting your own JIRA instance, you can cancel your OnDemand trial and switch to installed JIRA. Refer to the JIRA Installation and Upgrade Guide for information about installing JIRA.
  If you want to keep the content you created in your OnDemand instance, see Migrating from JIRA OnDemand to a JIRA Installed Site.
  If you were evaluating JIRA using the installable trial, you might find the information on Switching Databases useful.

You can find more information about managing your OnDemand subscription at www.atlassian.com/licensing/ondemand

Feedback

Let us know what you thought of this tutorial via this quick five minute survey:

Click to access the survey (you will be redirected to the Wufoo site): http://atlassian.wufoo.com/forms/jira-101-feedback/

Resources for topics covered in this guide
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
- Using JIRA on a Mobile Device
- Getting started with Bamboo

Working with an Issue

- Adding a field to an issue
- 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
- Labeling 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
- Integrating with code development tools

Searching for Issues

- Basic Searching
- Using Quick Search
- Advanced Searching
- Advanced Searching Functions
- Performing Text Searches
- Using the Issue Navigator
  - Customizing your Issue Navigator
- Using Filters
- Working with Search Result Data
  - Displaying Search Results as a Chart
  - Displaying Search Results in XML
  - Exporting Search Results to Microsoft Excel
  - Exporting Search Results to Microsoft Word
  - Receiving Search Results as an RSS Feed
  - 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

**Customizing the Dashboard**

• Managing Multiple Dashboard Pages
• Changing the Look and Behavior 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 Favorite 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 Chart 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
- Changing your JIRA Home Page
- Requesting Add-ons

Data collection policy

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

JIRA Concepts

Read the following pages for information about 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:

![Project: Website Issues (Key: WEB)](image)

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.

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**

A JIRA workflow is the set of statuses and transitions that an issue goes through during its lifecycle. The following diagram shows JIRA's built-in workflow:
What is an Issue

Different organizations use JIRA to track different kinds of issues. Depending on how your organization 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 (click to enlarge image):

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>JIRA's system workflow can customized by your JIRA administrator.</td>
</tr>
<tr>
<td>Resolve Issue</td>
<td>You can access an issue in JIRA from a search result or from a dashboard gadget that provides access to issues.</td>
</tr>
<tr>
<td>Reopen Issue</td>
<td>A JIRA issue typically looks like this (click to enlarge image):</td>
</tr>
<tr>
<td>Resolve Issue</td>
<td>The fields shown in the above screenshot are:</td>
</tr>
<tr>
<td>Reopen Issue</td>
<td>Field</td>
</tr>
<tr>
<td>Reopen Issue</td>
<td>Log work, attach files &amp; screenshots, create sub-tasks, move, link, or clone the issue.</td>
</tr>
<tr>
<td>Add a field or access the Admin helper from this menu</td>
<td>Transition the issue in its workflow here.</td>
</tr>
<tr>
<td>Email this issue to others</td>
<td></td>
</tr>
<tr>
<td>Export this issue to other formats, such as MS Word</td>
<td></td>
</tr>
<tr>
<td>To create a code branch in Bitbucket or Stash, click the Create Branch link</td>
<td></td>
</tr>
</tbody>
</table>

Your JIRA issues may look different to the above screenshot if your administrator has customized JIRA for your organization.
| **Project** | The parent **project** to which the issue belongs. In this case, Angry Nerds. |
| **Key** | A unique identifier for this issue, in the example above: ANGRY-304. (The characters to the left of the hyphen represent the **project** to which this issue belongs.) |
| **Summary** | A brief one-line summary of the issue. For example, "Red Angry Nerd is scary." |
| **Type** | See below for a list of types. |
| **Status** | The stage the issue is currently at in its lifecycle (**workflow**). See below for a list of statuses. |
| **Priority** | The importance of the issue in relation to other issues. (See below for a list of priorities). |
| **Resolution** | A record of the issue's **resolution**, if the issue has been resolved or closed. (See below for a list of resolutions). |
| **Affects Version(s)** *(if applicable)* | Project **version(s)** for which the issue is (or was) manifesting. |
| **Fix Version(s)** *(if applicable)* | Project **version(s)** in which the issue was (or will be) fixed. |
| **Component(s)** *(if applicable)* | Project **component(s)** to which this issue relates. |
| **Labels(s)** *(if applicable)* | **Labels** to which this issue relates. |
| **Environment** *(if applicable)* | The hardware or software environment to which the issue relates. |
| **Description** | A detailed description of the issue. |
| **Links** | A list of links to related issues. (Strikethrough text, **link** indicates that an issue has been resolved.) |
| **Assignee** | The person to whom the issue is currently assigned. |
| **Reporter** | The person who entered the issue into the system. |
| **Votes** | The number shown indicates how many **votes** this issue has. |
| **Watchers** | Number shown indicates how many people are watching this issue. |
| **Due** *(if applicable)* | The date by which this issue is scheduled to be completed. |
| **Created** | The time and date on which this issue was entered into JIRA. |
| **Updated** | The time and date on which this issue was last edited. |
| **Resolved** | The time and date on which this issue was resolved. |
| **Estimate** | The **Original Estimate** of the total amount of time required to resolve the issue, as estimated when the issue was created. |
### Remaining

<table>
<thead>
<tr>
<th>Remaining</th>
<th>The Remaining Estimate, i.e. the current estimate of the remaining amount of time required to resolve the issue.</th>
</tr>
</thead>
</table>

### Logged

<table>
<thead>
<tr>
<th>Logged</th>
<th>The sum of the Time Spent from each of the individual work logs for this issue.</th>
</tr>
</thead>
</table>

### Development

<table>
<thead>
<tr>
<th>Development</th>
<th>If you use Bitbucket or Stash to manage your code repositories, you can create code branches in your code development tools directly from JIRA issues. See Integrating with code development tools for details.</th>
</tr>
</thead>
</table>

### Agile

<table>
<thead>
<tr>
<th>Agile</th>
<th>Let's you view your issue on your Scrum or Kanban board.</th>
</tr>
</thead>
</table>

Some of the most important fields are described as below.

#### 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 customized this list to suit your organization.

- **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 organization 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 customized by your JIRA administrator to suit your organization.

- **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 customized the available statuses to suit your organization.

- **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'. A resolution is usually set when the status is changed. The default resolutions are listed below; note that your JIRA administrator may have customized these fields.

- **Fixed** — The issue is fixed.
- **In Progress** — The issue is still current, but being actively worked on.
- **Resolved** — The issue has been resolved, but the fix has not been verified.
- **Reopened** — The issue was Reopened, and the fix is checked back in.
- **Accepted** — The fix has been accepted, and it is considered to be fixed.
- **Closed** — The issue is closed, and the fix has been checked in.
- **Invalid** — The issue is invalid, and it cannot be fixed.
- **Other** — The issue is closed for other reasons.
ed these to suit your organization.

**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
- Using JIRA on a Mobile Device
- Getting started with Bamboo

#### 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.

**On this page:**

- Log in to JIRA
- Cannot remember either your username or password
- Sign up for an account

**Log in to JIRA**

1. Enter your **Username** and **Password** and click the **Log In** button.
2. If you have not changed your JIRA home page, the **dashboard** will be displayed. Otherwise, your chosen JIRA home page will be displayed instead.

**Please Note:**

- Selecting the **Remember my login on this computer** check box will prevent you from being automatically logged out of JIRA on a given browser and computer. However, your session will not be preserved, e.g. last search, current project, etc.
Cannot remember either your username or password

1. Click Can't access your account?
2. Fill in the fields on the 'Can't access your account?' page, as follows:
   - If you cannot remember your password, select the Password option and Enter your username in the field provided.
   - If you cannot remember your username, select the Username option and Enter your email address specified in your JIRA user profile.
3. Click Send. A new password will emailed to the email address specified in your user profile.

If you have forgotten the email address specified in your user profile, you will need to contact your JIRA administrator for help.

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.

1. Click the Sign up link in 'Not a member? Sign Up for an account'.
2. Enter your details.
3. Click the Sign up button to create your account.

Exploring the JIRA Workspace

The Dashboard is the first page you see (by default) after logging in 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 customized to display 'gadgets' showing many different types of information, depending on your areas of interest.

Please Note:

- Your JIRA dashboard page may look different from the one in this screenshot, as the logo and colours may have been customized by your JIRA administrator. The links in the navigation bar, however, will be the same.
- If you have changed your JIRA home page from the default (dashboard), the page you see after logging in to JIRA will be different.

Screenshot: JIRA dashboard (click to enlarge)
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.

On this page:
- Keyboard Shortcuts
- Modifier Keys
- Accessing the Keyboard Shortcuts Dialog Box
- Disabling and Re-enabling Keyboard Shortcuts

Keyboard Shortcuts

Global Shortcuts

These shortcuts are available from any JIRA screen.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Keyboard Shortcut</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to Dashboard</td>
<td>g then d</td>
<td>Directs you to the Dashboard screen.</td>
</tr>
<tr>
<td>Browse to a Project</td>
<td>g then p</td>
<td>Directs you to your current project browser screen.</td>
</tr>
<tr>
<td>Find Issues</td>
<td>g then i</td>
<td>Opens the Issue Navigator, where you can search for issues using either the Simple Search or Advanced Search features.</td>
</tr>
<tr>
<td>Quick Search</td>
<td>/</td>
<td>Directs your cursor to the Quick Search text field in the top right-hand corner.</td>
</tr>
<tr>
<td>Create an Issue</td>
<td>c</td>
<td>Opens the Create Issue dialog box for creating an issue.</td>
</tr>
<tr>
<td>Open shortcut help</td>
<td>?</td>
<td>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.</td>
</tr>
<tr>
<td>Feature</td>
<td>Modifier Key(s)</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Dashboards drop-down menu</td>
<td>Modifier key(s) + d</td>
<td>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.</td>
</tr>
<tr>
<td>Projects drop-down menu</td>
<td>Modifier key(s) + p</td>
<td>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.</td>
</tr>
<tr>
<td>Issues drop-down menu</td>
<td>Modifier key(s) + i</td>
<td>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.</td>
</tr>
<tr>
<td>Administration Search dialog box</td>
<td>g then g</td>
<td>Opens the Administration Search dialog box.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
</tbody>
</table>

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

'View Issue' and 'Issue Navigator' Shortcuts

These shortcuts are available from JIRA's View Issue and Issue Navigator screens only.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Keyboard Shortcut</th>
<th>Description</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>View selected Issue</td>
<td>o or Enter</td>
<td>Opens the currently selected issue on the Issue Navigator.</td>
<td>Issue Navigator</td>
</tr>
<tr>
<td>Feature</td>
<td>Key</td>
<td>Description</td>
<td>Reference</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Next Issue</td>
<td>j</td>
<td>Navigates to the next issue. This keyboard shortcut only applies to a View Issue screen if you got there via the Issue Navigator.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Previous Issue</td>
<td>k</td>
<td>Navigates to the previous issue. This keyboard shortcut only applies to a View Issue screen if you got there via the Issue Navigator.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Back to the Navigator</td>
<td>u</td>
<td>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.</td>
<td>View Issue</td>
</tr>
<tr>
<td>Hide/Show Left Section</td>
<td>[</td>
<td>(left square bracket) Hides or shows the left section on the Issue Navigator.</td>
<td>Issue Navigator</td>
</tr>
<tr>
<td>Next Activity</td>
<td>n</td>
<td>Navigates to the next item in the activity section of the View Issue screen.</td>
<td>View Issue</td>
</tr>
<tr>
<td>Previous Activity</td>
<td>p</td>
<td>Navigates to the previous item in the activity section of the View Issue screen.</td>
<td>View Issue</td>
</tr>
<tr>
<td>Focus search field</td>
<td>f</td>
<td>Focuses the cursor in the Advanced Search text field or the Simple Search's Query text field.</td>
<td>Issue Navigator</td>
</tr>
<tr>
<td>Escape field</td>
<td>Esc</td>
<td>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.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Edit Issue</td>
<td>e</td>
<td>Opens the Edit Issue dialog box (if you have appropriate permission), where you can edit the issue.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Feature</td>
<td>Key</td>
<td>Description</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>Assign Issue</td>
<td>a</td>
<td>Opens the Assign dialog box (if you have appropriate permission), where you can assign the issue to another JIRA user.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Comment on Issue</td>
<td>m</td>
<td>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.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Edit Issue Labels</td>
<td>I (lower-case 'L')</td>
<td>Opens the Labels dialog box, where you can edit the labels associated with the issue.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Share Issue</td>
<td>s</td>
<td>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.</td>
<td>View Issue and Issue Navigator</td>
</tr>
<tr>
<td>Operations dialog box</td>
<td>. (full-stop/period)</td>
<td>Opens the Operations dialog box, from which you can perform any permitted issue operation on the current JIRA issue by doing either of the following:</td>
<td>View Issue and Issue Navigator</td>
</tr>
</tbody>
</table>
• 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.
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 customized 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 assignee 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.

---

**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.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Keyboard Shortcut</th>
<th>Description</th>
</tr>
</thead>
</table>

---

Note that some options in the Action menu will only be available if you have the necessary permissions, or if certain features have been enabled by your JIRA administrator.
<table>
<thead>
<tr>
<th>Action</th>
<th>Modifier key(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form Submit</td>
<td>Modifier key(s) + s (Alt + s only in Chrome on Windows or Linux/Solaris)</td>
<td>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.</td>
</tr>
<tr>
<td>Cancel Form</td>
<td>Modifier key(s) + ` (backquote)</td>
<td>Cancels any currently open form in JIRA.</td>
</tr>
<tr>
<td>Escape Field</td>
<td>Esc</td>
<td>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.</td>
</tr>
<tr>
<td>Back</td>
<td>u</td>
<td>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.</td>
</tr>
<tr>
<td>'Remember my login' check box</td>
<td>Modifier key(s) + r</td>
<td>Automatically selects the 'Remember my login on this computer' check box. This shortcut only applies to JIRA's login forms.</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>Web Browser</th>
<th>Mac OS X</th>
<th>Windows</th>
<th>Linux/Solaris</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefox</td>
<td>Ctrl</td>
<td>Alt + Shift</td>
<td>Alt + Shift</td>
</tr>
<tr>
<td>Internet Explorer</td>
<td></td>
<td>Alt</td>
<td></td>
</tr>
<tr>
<td>Safari</td>
<td>Ctrl + Alt/Option</td>
<td>Ctrl</td>
<td></td>
</tr>
<tr>
<td>Chrome</td>
<td>Ctrl + Alt/Option</td>
<td>Alt + Shift</td>
<td>Alt + Shift</td>
</tr>
</tbody>
</table>

**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 customize ‘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.

To open the keyboard shortcuts dialog box:

Choose > Keyboard Shortcuts.

You can also open this dialog by pressing ? (Shift + /) on your keyboard. Note, your cursor must not be focused in a JIRA text field for ? to work.

If you have JIRA plugins installed, you may have additional keyboard shortcuts available. For example, if you have JIRA Agile 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 JIRA Agile in the upper-left section). However, the keyboard shortcuts in the Agile Shortcuts section only function in JIRA Agile 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 section.
d 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.

Using JIRA on a Mobile Device

When you view a JIRA page on a mobile device, such as an iPhone or an Android phone, JIRA will display an optimised version of the page. JIRA chooses the mobile or desktop interface based on your device.

The JIRA mobile interface is designed for viewing and interacting with issues on the go. If you need full access to JIRA, you can always switch to the JIRA desktop interface via the mobile menu (shown in the screenshots below).

What does JIRA look like on a mobile device?

Viewing issues ("Assigned to Me" filter) on a mobile device

JIRA menu on a mobile device

Viewing an issue on a mobile device

What can you do in JIRA on a mobile device?

The JIRA mobile interface has been designed to give users quick access to their issues on the go. This includes;

- Viewing issues, comments, attachments, issue links and your favourite filters.
- Performing basic operations like adding comments, watching or voting on issues and assigning issues to users.

If you need to create or modify issues on the go, you can still do so by switching to the desktop interface via the mobile menu (shown in the screenshots above).

Frequently asked questions

- What mobile devices are supported?
- Do I need to install an app to view JIRA on a mobile device?
- Can I access my JIRA OnDemand site via a mobile device?
- Why can’t I view my custom field in JIRA on my mobile?

What mobile devices are supported?

See Supported Platforms for details of supported mobile devices.
Do I need to install an app to view JIRA on a mobile device?

No, JIRA is viewed on a mobile device via a web interface (optimised for mobile devices), not an app. Simply browse to your JIRA server's URL using your mobile browser to bring up the mobile interface for JIRA.

Can I access my JIRA OnDemand site via a mobile device?

Yes, just enter the URL of your JIRA OnDemand site in your mobile web browser.

Why can't I view my custom field in JIRA on my mobile?

The JIRA Mobile interface will show custom fields in the issue details screen. Custom fields that have their own custom field renderer will not display on the JIRA Mobile interface. You will need to switch to the desktop interface to view these fields.

Can I disable JIRA mobile for my site?

You can disable JIRA mobile for your site, so that users will only be able to access the desktop view of JIRA on their mobile device.

JIRA mobile is a implemented as a add-on in JIRA, so you can disable it by disabling the add-on. For instructions on disabling add-ons, see Managing Add-ons. Note, JIRA mobile is a System Plugin.

Getting started with Bamboo

Bamboo is a continuous integration (CI) server. Bamboo assists software development teams by providing:

- automated building and testing of software source-code status.
- updates on successful/failed builds.
- reporting tools for statistical analysis.

Please see the following page for information about getting started with Bamboo:

- Understanding the Bamboo CI Server - a conceptual overview of using Bamboo for continuous integration (CI).

Working with an Issue

The following pages contain information on working with an issue:

- Adding a field to an issue
- 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
- Labeling 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
- Integrating with code development tools

Adding a field to an issue

JIRA Admins can add an existing field or create a custom field while in View Issue with the Admin > Add field option.
Custom fields enable JIRA administrators to quickly customize the way issues look. In addition to adding this new field to the View issue screen, it is also added to the Create and Edit screens as well. This means you can edit the field you just created instantly, and input data into that field the next time you add an issue.

Finally, when creating a custom field, you can choose between Standard and Advanced types. For Standard types, a preview image is shown for each type, so you can see what you are creating in advance.

Adding a new field

Use this procedure to add a new field while in the View issue screen.

1. Select the Admin > Add field option. You are presented with this screen, where you can enter your field name (as shown here):

2. Once you have entered an acceptable field name (it cannot be the name of a field that already exists), click Create.
3. Select a field type from the displayed list and click Next.
4. Configure the selection criteria for your field, as shown in the example below:

Click **Create** when you are happy with your field options.

5. You are presented with your options. Make a selection and click **Submit**. This field now displays on your issue screen:
Adding a field that already exists

Use this procedure to add a field that already exists while in the View issue screen.

1. Select the **Admin > Add field** option. Click the down arrow on the Name selection box, as shown here:

   ![Add Field Dialogue](image)

   **Note:** Only fields that aren't already in use on the displayed issue are available.

2. Select a field from the list and click **Add**.
3. If necessary, select any display options and click **Submit**.
You will see a message display briefly on your View issue screen that tells you the issue was added.

## Attaching a File

JIRA allows you to attach files to an issue.

ℹ️ 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

ℹ️ 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. Select **More > Attach Files**.
3. The Attach Files dialog box is displayed.
4. Click **Browse** to search for your files.

ℹ️ 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.

- If you use Firefox 3.6 or later or Chrome browsers, you can select multiple files in the browser dialog box.
- If you select a file by mistake, clear its check box to prevent the file from being attached to the issue.
- By default, the maximum size of any one file is 10MB, although this limit can be customized by your JIRA administrator. See Configuring File Attachments for more information.
- File names cannot contain any of these characters: '\', '/', '\n', '%', ':', '$', '?', '*'.

5. Optional: Enter a comment about the files(s) you are attaching.

ℹ️ 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.

ℹ️ If your JIRA administrator has disabled Thumbnails in JIRA's Attachment Settings, then the image files will be part of the list of attached files instead.

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 to see previews of the adjacent attached images.

### 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 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:

   ![Manage Attachments](image)

   - Note: Only users with the appropriate Delete Issues project permissions can remove attachments.
3. Locate the file you wish to delete and click the Delete Attachment icon:

Attaching a Screenshot
JIRA allows you to attach screenshots to an issue.

**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. Select More > Attach Screenshot.
3. The Attach Screenshot window opens 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 Yes.
4. Ensure that you have captured an image to your operating system's clipboard and click the Paste button to paste the image.
5. Enter a file name for the screenshot you are attaching. (The file name defaults to `screenshot-1`.)
   - A valid file name cannot contain any of these characters: `'`, `/`, `\`, `'`, `¥`, `:`, `'`, `$`, `?`, `*`.
6. Optional: Enter a comment for the screenshot you are attaching.
   - 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 default security level for the comment is All Users.) The comment is added to the JIRA issue, with the selected security level, when the screenshot is attached.
7. Click Attach to add the captured image to your 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 favorite 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 favorite imaging application and select the copy option from the appropriate menu to capture the image into the system clipboard.

**Cloning an Issue**
Cloning, or copying, an issue lets you 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.

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.

A clone issue retains the following information:

- **Summary**
  (With optional prefix that can be customized by your JIRA system administrator; see Configuring Issue Cloning for details.)
- **Description**
- **Assignee**
- **Environment**
- **Priority**
- **Issue Type**
- **Security**
- **Reporter**
  (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**
  (Once the clone has been saved, you can move it to another project as described in Moving an Issue.)

The content of custom fields is also cloned.

**Things that aren't cloned:**

- Time tracking
- Comments
- Issue history

**Creating a Clone Issue**

**To clone an issue:**

1. Open the JIRA issue you wish to clone.
   - 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 Create.

**Cloned Issue Linking Behavior**

By default, when an issue is cloned, JIRA automatically creates a link between the original and cloned issue using the pre-existing link type name 'Cloners'.
Your JIRA system administrator can customize this default behavior 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 customize this default behavior 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).
- You automatically become a watcher of the issues that you comment on. You can disable this via the Preferences > Autowatch option in your profile.

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.
   - 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: %20Restricted 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.

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 or expand a comment, click the arrow 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.

**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.

1. Browse to the comment you wish to link to.
2. Click the Permalink (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 at the top of the screen to open the Create Issue dialog box.
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 that are not shown on this dialog box or you want to hide existing fields:
     a. Click the Configure Fields button at the top right of the screen.
     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: To create a series of similar issues — with the same Project and Issue Type — select the Create another check box at the bottom of the dialog.
5. When you are satisfied with the content of your issue, click the Create button.
   - If you selected the Create another check box (above), a new Create Issue dialog appears. This issue is 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.
- In certain text fields for an issue, you can link to other issues, insert macros, insert images and more. For more information, see Editing Rich-Text Fields.
- 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.
- You may automatically become a watcher of the issues that you create, depending on the Autowatch setting in your user profile. Note, if you have not changed this setting, you will inherit the global Autowatch settings set by your JIRA administrator (in Administration > User Preferences).
- With appropriate configuration by your JIRA administrator, it is also possible to create issues via email.

Screenshot: Example 'Create Issue' dialog box
Sub-task issues are useful for splitting up a parent issue into a number of smaller tasks that can be assigned and tracked separately. This can provide 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
- Working with sub-tasks
- Searching for sub-tasks
- 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 need to have the Create Issue permission in the parent issue's project. 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.

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 you would like to be the parent issue of the sub-task you are about to create.
2. Select More > Create Sub-Task. You will see the Create Subtask screen.
3. Fill in the details as needed, and then click Create at the bottom of the page.

Tip: You can customize the Create Subtask dialog to show fields you use most often. To do this, click Configure Fields at the top right corner of the dialog, and use the All and Custom links to switch between the default
Working with sub-tasks

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

- **Show open sub-tasks only** – The sub-task list has two views: *Show All* and *Show Open*. *Show All* lists all sub-tasks, regardless of status, while *Show Open* only shows sub-tasks that have not been resolved (i.e. do not have a Resolution).
- **Time Tracking** – The colored bars show the time-tracking data for the issue and its sub-tasks.

![Time Tracking](image)

- **Perform actions on sub-tasks** – Click on the right side of the sub-task in the list to display the Actions dropdown menu:

![Actions Dropdown](image)

**Tip:** You can also type a period `.' to access issue actions.

- **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 appear in the Issue Type dropdown list in the search form.
To search standard issues only, click **All Standard Issue Types.**
To search sub-task issues only, click **All Sub-Task Issue Types.**
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** list, the search returns all the standard and sub-task issues that meet the search criteria.

Converting a standard issue to a sub-task

1. Navigate to the issue you would like to convert.
2. Select **More > Convert to Sub-Task.**
3. In the **Step 1. Select Parent Issue and Sub-Task Type** screen, type or select the appropriate parent issue type and the new issue type (i.e. a sub-task issue type). Click **Next.**
4. If the issue's current status is not an allowed status for the new issue type, the **Step 2. Select New Status** screen is displayed. Select a new status and click **Next.**
5. In the **Step 3. Update Fields** screen you will be prompted to enter any additional fields if they are required. Otherwise, you will see the message 'All fields will be updated automatically'. Click **Next.**
6. The **Step 4. Confirmation** screen is displayed. If you are satisfied with the new details for the issue, click **Finish.**
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.

**Note:** You will not be able to convert an issue to a sub-task if the issue has sub-tasks of its own. You 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 you would like convert.
2. Select **More > Convert to Issue.**
3. In the **Step 1. Select Issue Type** screen, select a new issue type (i.e. a standard issue type) and click **Next.**
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 is displayed. Select a new status and click **Next.**
5. In the **Step 3. Update Fields** screen you will be prompted to enter any additional fields if they are required. Otherwise, you will see the message 'All fields will be updated automatically'. Click **Next.**
6. The **Step 4. Confirmation** screen is displayed. If you are satisfied with the new details for the issue, click **Finish.**
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
   
   Alternatively, hover your mouse over a field and click the pencil icon to edit it inline.
   
   ✔ **Keyboard shortcut:** , then enter the desired field name.
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.

**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.
- In certain text fields for an issue, you can link to other issues, insert macros, insert images and more. For more information, see [Editing Rich-Text Fields](#).

**Screenshot: Customizing the fields on the Edit Issue dialog**

![Edit Issue dialog](https://example.com/edit-issue-dialog.png)

**Related topics**

- [Linking Issues](#)
- [Emailing an Issue](#)
- [Editing Rich-Text Fields](#)

**Editing Rich-Text Fields**

When you create, edit or comment on a JIRA issue, some fields may display two small icons at the bottom of the text area: a blue screen – the **Preview** icon – and a question-mark – the **Help** icon. The presence of these icons indicates that this field supports JIRA’s **Text Formatting Notation**, which allows 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).

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:

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
<th>Enabled by default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Anchor Macro</strong></td>
<td>Create an anchor that allows people to link to a specific point in a page. Usage:</td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td><img src="example.png" alt="Anchor Macro Example" /></td>
<td></td>
</tr>
<tr>
<td><strong>Code Macro</strong></td>
<td>Format blocks of source-code or XML. The default language is Java but you can specify JavaScript, ActionScript, XML, HTML and SQL too. Usage:</td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td><img src="example2.png" alt="Code Macro Example" /></td>
<td></td>
</tr>
<tr>
<td><strong>Quote Macro</strong></td>
<td>Generate blockquotes that may contain multiple paragraphs or complex markup. Usage:</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>
|                | `{quote}
This is text
from another source
{quote}` |
| **No Format Macro** | Create blocks of text where other wiki formatting is not applied. Usage: |
|                  | `{noformat}
This is text
\#&@!@(*!!@()#@\"
\``
{macros}
_wont_work_here
{noformat}` |
<table>
<thead>
<tr>
<th>Panel Macro</th>
<th>Draw a panel with the following optional parameters:</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• title: Title of the panel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• borderStyle: The style of the border this panel uses (solid, dashed and other valid CSS border styles)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• borderColor: The color of the border this panel uses</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• borderWidth: The width of the border this panel uses</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• bgColor: The background color of this panel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• titleBGColor: The background color of the title section of this panel</td>
<td></td>
</tr>
</tbody>
</table>

```{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=#FFF FCE}

A block of text surrounded with a *panel* yet _another_ line

{panel}```
### Color Macro

Change the color of the contained text. Usage:

```
{color:red}
look ma, red text!
{color}
```

You can use any of the sixteen standard HTML colors ('aqua','black','blue','fuschia','green ','gray','lime', 'maroon','navy','olive','purple','red',' silver','teal','white','yellow'), or any hexadecimal color code (e.g. '#336699').

### Lorem Ipsum Macro

Insert paragraphs of "lorem ipsum" space-filler text. Usage:

```
{loremipsum}
```

### 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>
```

---

**Emailing an Issue**

You can 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.

You can also email an issue any email address — not only just JIRA users — using the Share option. This is useful on JIRA sites where projects can be accessed anonymously.

**Sharing an issue**

To email other JIRA users a link to an issue, use the Share option. This feature also allows you to 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...
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 individuals you wish 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 list of users appears.
4. Add an optional Note.
5. Click the Share button at the bottom of the displayed window.

**Note:**
- Any specified recipients will receive an email message whose body contains the content of the Note (if one was included) 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. For details, see Sharing a Search Result.

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.

**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
- Labeling an Issue

**Labeling an Issue**

**Labeling** allows you to categorize 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**

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 that have this label. You can also:

- browse for labeled 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**).

If your JIRA administrator has added any **custom fields** of type ‘Label’, they will be shown below the ‘Labels’ field.

**Related topics:**

- **Browsing a Project’s Labels**
- **Adding the Labels Gadget**

**Adding and removing labels for an issue**

1. View the issue you wish to label.
2. Select **More > Labels**. The Labels dialog will appear.
   
   🔄 Keyboard shortcut: I
3. Add or remove the desired label(s) and click **Update**.
   **Note:** You can add multiple labels by typing them separated by spaces.

### Linking Issues

**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 **customize** 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.

**On this page:**

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

**Issue links within an issue look like this:**

**Screenshot:** the *Issue Links* section within an issue

**Note:** **Resolved** issues are displayed in strike-through font, e.g. **DEMO-1**.

To create links on JIRA issues, you need to have the **Link Issues permission** in the project(s) to which the issues belong.
Creating a link to another issue on the same JIRA site

To create a link to another issue on the same JIRA site:

1. Open the issue you wish to link to another issue in the same JIRA site.
2. Select More > Link to display the Link dialog box.
   
   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 from the This issue dropdown list.
   
   If your JIRA system administrator has configured fully reciprocal application links between your JIRA site and another one, a Server dropdown list may appear above the This issue list. If this is the case, ensure your JIRA site appears or has been selected from the Server list.

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 search for an issue link to use the Find JIRA issues popup, which allows you to perform either a simple text search or an advanced search for issues.

5. Optional: Add a Comment to describe why you are linking these issues.
6. Click the Link button at the bottom of the dialog.

Creating a link to an issue on another JIRA site

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

This feature is not available in OnDemand.

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

1. Open the issue you wish to link to another issue.
2. Select More > Link to display the Link dialog box.
   Keyboard Shortcut: '.' + start typing link
3. Ensure that the JIRA issue item is selected at the left of the dialog box.
   **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 behavior means the application links configured between your JIRA site and the remote JIRA site use OAuth authentication.**
4. If your JIRA site is connected to multiple remote JIRA sites, choose the relevant JIRA site from the Server dropdown list.
5. Choose the type of link to be created from the This issue dropdown list.
6. Type the Issue key of the issue on the remote JIRA site that you want to link to. Alternatively, you can search for issues on the remote JIRA site by clicking the search for an issue link, which opens the Find JIRA issues popup.
   You can link to any issue on the remote JIRA site to which you have access on that site.
7. 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.
8. Optional: Add a Comment to describe why you are linking these issues.
9. Click the Link button at the bottom of the dialog.

**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'.'

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**

[This feature is only supported in Confluence versions 4.0 or later.]

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. Open the issue you wish to link to another issue.
2. Select More > Link to display the Link dialog box.
   ✔️ Keyboard Shortcut: '.' + start typing link
3. Click the Confluence Page option at the left of the dialog box.
   ✏️ This option is not available if your JIRA system administrator has not configured an application link (remote application) 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 behavior 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: Add a Comment to describe why you are linking these issues.
7. Click the Link button at the bottom of the dialog.

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:

'CContent 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 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. Open the issue you wish to link to another issue.
2. Select More > Link to display the Link dialog box.
   ✔️ 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: Add a Comment to describe why you are linking these issues.
7. Click the Link button at the bottom of the dialog.
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
- Customized 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 Log Work dialog, shown below:

![Log Work - JRADEV-19481](image)

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 of the screen and complete the required field details.

2. 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.*
3. 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’.

   ![Tip](image)
   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.

4. Click **Update** at the bottom of the screen.

   ![Tip](image)
   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 d 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 **More > Log Work**.
3. The **Log Work** dialog box is 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.
   - 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.
   - 'Use existing estimate of ...' — Select this if you do not want to change the issue's Remaining Estimate value.

   This option is only displayed if no time estimates have been specified.
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 **Log** at the bottom of the dialog to submit the form and close this window.
   - 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. Select either **Workflow > Resolve Issue** or **Workflow > Close Issue**.
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 **Resolve** (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 screens in JIRA’s default workflow (or a custom 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** (pencil) icon, located at the right of the work log entry.
5. The **Edit Work Log** screen is 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 Log 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.

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. The Delete Worklog screen is 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 ...
6. Click Delete 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).

Customized 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 customized 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 customized JIRA operation that allows you to log work and/or specify time estimates on the same JIRA screen. For example, assuming that your JIRA administrator has added the Time Tracking fie
3. 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 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.
   - 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.

   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.

See also

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

Modifying Multiple (‘Bulk’) Issues

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

- Workflow Transition
  - Lets you transition multiple issues through a workflow at once — e.g. resolve a collection of issues.
- Delete
  - Lets you delete multiple issues at once.
- Move
  - Lets you move multiple issues between projects and/or issue types. Please see the Bulk Move section for further details.
- Edit
  - Lets you edit multiple fields in multiple issues at once. Please see the Bulk Edit section for further details.
- Watch / Stop Watching
  - Lets you start or stop watching multiple issues.
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

To disable mail notification for a particular bulk operation, deselect 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. Read more about this here.

Performing a Bulk Operation

1. Perform a search with the required filters to produce an issue result set.
2. Select Tools > Bulk Change.
3. The next screen allows the selection of the issues on which to perform the bulk operation.
4. The next screen allows the bulk operation to be selected, see the table below for the operation you are performing.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete</td>
<td>The final step is confirmation of the delete operation on the selected issues.</td>
</tr>
<tr>
<td>Edit</td>
<td>The next screen provides a list of the available edit operations that can be performed on the issues selected. After selecting the required Edit operation(s), the final step is confirmation of the edit operation(s) on the selected issues.</td>
</tr>
<tr>
<td>Move</td>
<td>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.</td>
</tr>
<tr>
<td>Workflow Transition</td>
<td>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.</td>
</tr>
</tbody>
</table>
**Watch**

The final step is a confirmation of the watch operation on the selected issues.

**Stop Watching**

The final step is a confirmation of the unwatch operation on the selected issues.

**Bulk Operation Screens**

See the images below for examples of the Bulk Operation workflow.
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 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.

Selected issues are 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.

Note: This does 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. You can also 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.

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.

For the following fields, JIRA provides 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.

**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.

<table>
<thead>
<tr>
<th>Available Operations</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change Affects Version/s</td>
<td>• Selected issues belong to one project, and that project has version/s&lt;br&gt;• This field is not hidden in any field configurations the selected issues belong to&lt;br&gt;• Current user has ‘edit issue’ permission for all the selected issues</td>
</tr>
<tr>
<td>Change Assign To</td>
<td>• This field is not hidden in any field configurations the selected issues belong to&lt;br&gt;• Current user has ‘assign issue’ permission for all the selected issues</td>
</tr>
<tr>
<td>Change Comment</td>
<td>• This field is not hidden in any field configurations the selected issues belong to&lt;br&gt;• Current user has ‘comment issue’ permission for all the selected issues</td>
</tr>
<tr>
<td>Change Component/s</td>
<td>• Selected issues belong to one project, and that project has component/s&lt;br&gt;• This field is not hidden in any field configurations the selected issues belong to&lt;br&gt;• Current user has ‘edit issue’ permission for all the selected issues</td>
</tr>
</tbody>
</table>
### 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:
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 these 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.

To move an issue:

1. View the issue that you wish to move.
2. Select **More > Move**.
3. The first page of the **Move Issue** wizard is displayed. Select the project that you wish to move your issue to, and if required/desired, change the issue type. Click **Next** to continue.

![Move Issue Wizard](image)

4. If you are required to change the status of your issue (see explanation above), the **Select Status** page appears. 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 is displayed. 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.

Troubleshooting

- Restricted comments appear to be removed after moving the issue. See this article: Restricted comments disappear after moving an issue to a new project.

Scheduling an Issue

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. 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 field.
- To search for issues that are due before a certain date, enter the date into the Due Before text field. For example, to find all issues that are due before 1st July 2010, enter 1-7-2010 into the Due Before field.

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.

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

Use the Due Date popup to do the following:

- 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, 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 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.
box in the third row. Select the third radio button 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. 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 administrator 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 dropdown field, select the desired security level for the issue.
   - You can only see the Security Levels you belong to.
3. Save the issue. It is now only accessible to members of the specified 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 is 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. Open the History tab in the Activity section.
3. The list of changes to the issue will display, similar to the screen shot below.
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.

Your JIRA administrator must have configured the FishEye plugin on your JIRA server, if you want to view the reviews for an issue. You will also need the 'View Development Tools' 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.

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.

Your JIRA administrator must have configured the FishEye plugin on your JIRA server, if you want to view the changesets for an issue. You will also need the 'View Development Tools' 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.

See also

- Browsing a Project's FishEye Changesets

**Viewing the Bamboo Builds related to an Issue**

If your organisation uses [Atlassian's Bamboo](https://confluence.atlassian.com/display/ATTLIB/Atlassian+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.

You will need the 'View Development Tools' permission in the appropriate projects, if you want to view the 'Builds' tab for issues.

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.

---

**On this page:**

- Issue voting
- Issue watching
- Permissions
  - Mandatory project permissions
  - Optional global permissions

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

To vote for an issue:

1. Search for or view the issue you want to vote on and access its 'view issue' page.
2. Click Vote for this issue to instantly vote for the issue.
   • At any subsequent time when logged in, click this again to remove your vote.
   
   ![Votes and Watchers options]

To view a list of people who have already voted on an issue:

1. Search for or view the issue you want to vote on and access its 'view issue' page.
2. Click the number of votes to view the list of people who have voted for the issue.

Issue watching

To watch an issue:

1. Search for or view the issue you want to watch and access its 'view issue' page.
2. Click Start watching this issue to instantly become a watcher of the issue.
   • At any subsequent time when logged in, click this again to stop watching the issue.

To view existing users of, add new users to or remove existing users from an issue:

1. Search for or view the issue you want to watch and access its 'view issue' page.
2. Do either of the following:
   • Click the number of watchers to open the Add Watchers popup.

   ![Add Watchers popup]

   • Select More Actions > Watchers to open the Watchers page.

   ![Keyboard shortcut: ' ' + start typing watchers]

3. Use the Add Watchers popup or Watchers page to:
• View the list of existing users who are watching the issue,
• Add users as new watchers of the issue or
• Remove existing watchers.

**Tips:**

• If you have the optional global permissions (below), an autocompleted list of users appears as you begin typing a username or full name in either the Add Watchers popup (or Add Watchers field on the Watchers page), from which you can select a user via your keyboard’s cursor keys.

• When using the Add Watchers popup, you can remove an existing watcher from an issue by moving your mouse pointer over their name and clicking the ‘bin’ icon that appears to the right of their name.

• When using the Watchers page, if you have the optional global permissions, click the ‘user-picker’ icon to open the User Picker popup, from which you can select users.

**Permissions**

**Mandatory project permissions**

JIRA incorporates the following project 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 project permissions are granted by your JIRA administrator, through a Permission Scheme.

**Optional global permissions**

When adding watchers to an issue, to access:

• An autocompleted list of users in the Add Watchers popup, or

• The User Picker popup on the Watchers page,

your JIRA administrator must have granted you the Browse Users Global Permission. If you do not have this permission, however, you can still add users by specifying their usernames in either:

• The Add Watchers popup or

• The Add Watchers field on the Watchers page.

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.

**Integrating with code development tools**

If you use Bitbucket or Stash to manage your code repositories, you can create code branches in your code development tools directly from JIRA issues. This makes it easy to start working on a new branch for a particular bug or feature.

To create a code branch in Bitbucket or Stash, click the Create Branch link in the “Development” section on an issue. Only users with the “View Development Tools” project permission will have access to the “Development” section on issues.

Create branch link
Your code development tools must be connected to JIRA via application links in order for you to use this feature.

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. JIRA also makes custom fields available as search criteria, allowing you to refine your searches even further.

The search can be saved as a filter in JIRA, allowing you to recall the same search and run it again or even share it with other users.

With JIRA searches, you can:

- Modify and reuse search results
- Sort and action individual issues
- Share and export issues
- Display issues on your dashboard

On this page:

- Step 1. Define your search criteria
- Step 2. Modify the search results
- Step 3. Save your search
- Step 4. Working with search result data

Related Topics:

- Basic Searching
- Advanced Searching
- Using the Issue Navigator
- Using Filters

Step 1. Define your search criteria

The first step in searching for issues is to define the criteria for your new search. You can define your search criteria in three different ways: using the quick search, using the basic search or using the advanced search.

If you have saved a search previously (i.e. as a filter) and want to run it again, you can run the filter to load the saved search criteria.

<table>
<thead>
<tr>
<th>Search method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quick search</td>
<td>The quick search is the fastest way to define search criteria. However, it is less precise than other search methods for complex queries (e.g. project = JIRA AND status = Open AND priority = High). The quick search is in the navigation bar. For instructions, see Using Quick Search.</td>
</tr>
<tr>
<td><strong>Basic search</strong></td>
<td>The basic search is more precise than the quick search, but easier to use than the advanced search. It provides a user-friendly interface that lets you define complex queries, without needing to know how to use JQL. For instructions, see Basic Searching.</td>
</tr>
<tr>
<td><strong>Advanced search</strong></td>
<td>The advanced search is the most powerful of the three search methods. You can specify criteria that cannot be defined in the other searches (e.g. <code>ORDER BY</code> clause). However, you need to know how to construct structured queries using the JIRA Query Language (JQL) to use this feature. For instructions, see Advanced Searching.</td>
</tr>
</tbody>
</table>

### About filters
A JIRA filter is a saved set of search criteria, similar to a bookmark. Some filters are predefined in JIRA – such as My Open Issues, Reported by Me, Recently Viewed, All Issues – these are known as *system filters*. A filter may also be a set of search criteria defined and saved by a user.

To use a filter, simply click it. Your favorite filters are shown in the panel on the left. You can view all of your filters as well as find filters shared with you here.

When you run a filter, the saved search criteria is loaded for you in the displayed issue view. The criteria is displayed in either the basic search or advanced search, depending the search you are using at the time.

For more information about filters, see Using Filters.

#### Step 2. Modify the search results
Once you have defined and run your search, you can optionally modify the search results. JIRA lets you change the sort order of the search results, as well as action each of the individual issues.

For instructions, see Using the Issue Navigator.

#### Step 3. Save your search
To run your search again in future, save your search as a filter. Simply click **Save as** and enter a name for the filter. The filter is created and added to your favorite filters.

For more information, see Using Filters.

#### Step 4. Working with search result data
You can export or share the search results in a number of different formats. Search results can be shared via email or RSS. You can export the search results to a number of formats (e.g. Microsoft Excel). You can also display the search results on your dashboard as a list or a chart.

For more information, see Working with Search Result Data.
Basic Searching
This page describes how to define and execute a basic search. You can also define and execute a search using the quick search or using JQL (advanced searching).

Screenshot: Basic search (click to view full size image)

Performing a basic search

To perform a new basic search in JIRA:

   - If there are existing search criteria, click the New filter button to reset the search criteria.
   - If the Basic link is showing (in other words, you are in the Advanced searching mode), click Basic to switch search modes. This is what you should see:

2. Enter the criteria for the search. You can search against specific fields and/or search for specific text, as described below:
   - **Searching against specified fields** — To search against specified fields, you need to select the fields then specify the fields values that you want to find.
     a. Select fields as criteria — The Project, Issue Type, Status and Assignee fields are always available as criteria. If you want to search using additional fields as criteria, click More and tick the desired fields.
     i. Can't find the field you want? Field is displaying greyed out text? See the Troubleshooting section below.
     b. Specify the value for each field — Click the field criterion and tick/enter the desired value in the dropdown. If the desired value is not shown in the list, start typing the name of value the to filter the list, as shown here:
About the suggested users/groups for the Assignee and Reporter fields — A user/group will be suggested if you have recently selected them in an Assignee/Reporter field for an issue, or viewed the user's profile page.

- **Searching against specified text** — Enter the desired text in the *Contains text* text box and press Enter. The Summary, Description, Comments, Environment fields and all text-based custom fields will be searched. You can use modifiers in your search text, such as wildcards and logical operators, see Performing Text Searches.

3. The search results will automatically update, unless your administrator has disabled automatic updates of search results. If so, you need to click the **Update** button on the field dropdown after every change.

**Saved search**

Search results are saved as filters. For details, see Using Filters.

To perform a saved search:

1. Choose **Issues > Search for Issues**.
2. Choose any filter from the list on the left:
   - System filter — My Open Issues, Reported by Me, Recently Viewed, All Issues
   - Favorite filters (listed alphabetically)
   - Find filters lets you search for any filter that's been shared, which you can then subscribe to (adding it to your Favorite Filters).
3. After selecting a filter, the search results are displayed. The search criteria for the filter is also displayed. If you choose the Recently Viewed system filter, this switches you to advanced search due to the fact that basic search cannot represent the `ORDER BY` clause in this filter.
4. You can add, remove or modify the search criteria to refine the search results. You can also save the modified search criteria (if you are updating your own filter), or save a copy of the search criteria as a new filter.

**Screenshot: List of filters**
Troubleshooting

- **Why can't I find the field I want to choose?** Some fields are only valid for a particular *project/issue type context*, see Configuring Fields and Screens for details. For these fields, you must select the applicable project/issue type. Otherwise, the field is not available for selection.

- **Why are the field criteria displaying in grey text?** Some fields are only valid for a particular *project/issue type context*, see Configuring Fields and Screens for details. If you choose a field in your search, then remove all projects/issue types that reference the field, then the field is invalid. The invalid field does not apply to your search and displays in grey text.

- **Why is there a red exclamation mark 😱 in my field?** Some field values are only valid for a particular *project/issue type context*. For example, you may have configured a project to use a status *In QA Review* in its workflow. If you select this project and status in your search, then change the search to filter for a project that doesn't use *In QA Review*, the status will be invalid and ignored in the search.

- **Why don't my search results automatically update?** Your search results will always update automatically whenever any fields are changed, provided that your administrator has not disabled automatic updates of search results. Ask your administrator whether they have disabled automatic updates of search results.

**Next steps**

- Read Using the Issue Navigator to find out how to sort with the issues displayed in the search results, or view individual issues.
- Skip to Working with Search Result Data to find out how to save your search results as an issue filter, export your search results and share your search results.

**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'.

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:

<table>
<thead>
<tr>
<th>Search Term</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>my</td>
<td>Find issues assigned to me.</td>
<td>my open bugs</td>
</tr>
<tr>
<td>r:</td>
<td>Find issues reported by you, another user or with no reporter, using the prefix r: followed by a specific reporter term such as <code>me</code>, a username or <code>none</code>.</td>
<td>r:me — finds issues reported by you. r:samuel — finds issues reported by the user whose username is &quot;samuel&quot;. r:none — finds issues with no reporter.</td>
</tr>
<tr>
<td>&lt;project name&gt; or &lt;project key&gt;</td>
<td>Find issues in a particular project,</td>
<td>test project TST tst</td>
</tr>
<tr>
<td>overdue</td>
<td>Find issues that were due before today.</td>
<td>overdue</td>
</tr>
<tr>
<td>created:</td>
<td>Find issues with a particular Created, Updated, or Due Date using the prefixes <code>created:</code>, <code>update d:</code>, or <code>due:</code>, respectively. For the date range, you can use <code>today</code>, <code>to morrow</code>, <code>yesterday</code>, a single date range (e.g. <code>-1w</code>), or two date ranges (e.g. <code>-1w,1w</code>). Note that date ranges cannot have spaces in them. Valid date/time abbreviations are: <code>w</code> (week), <code>d</code> (day), <code>h</code> (hour), <code>m</code> (minute).</td>
<td>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.</td>
</tr>
<tr>
<td>Tag</td>
<td>Description</td>
<td>Examples</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;priority&gt;</code></td>
<td>Find issues with a particular Priority.</td>
<td>blocker, major, trivial</td>
</tr>
<tr>
<td><code>&lt;issue type&gt;</code></td>
<td>Find issues with a particular Issue Type. Note that you can also use plurals.</td>
<td>bug, task, bugs, tasks</td>
</tr>
<tr>
<td><code>&lt;resolution&gt;</code></td>
<td>Find issues with a particular Resolution.</td>
<td>fixed, duplicate, cannot reproduce</td>
</tr>
<tr>
<td><code>&lt;c&gt;</code></td>
<td>Find issues with a particular Component(s). You can search across multiple components.</td>
<td>c:security — finds issues with a component whose name contains the word &quot;security&quot;.</td>
</tr>
<tr>
<td><code>&lt;v&gt;</code></td>
<td>Find issues with a particular Affects Version(s). To find all issues belonging to a 'major' version, use the wildcard symbol '.*'.</td>
<td>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.</td>
</tr>
<tr>
<td><code>&lt;ff&gt;</code></td>
<td>Find issues with a particular Fix For Version(s). Same usage as <code>&lt;v&gt;</code> (above).</td>
<td>v:3.0* — finds any issue whose version number is (for example): 3.2, 3.2-beta, 3.2.1, 3.2.x</td>
</tr>
<tr>
<td><code>*</code></td>
<td>Wildcard symbol ‘*’. Can be used with <code>&lt;v&gt;</code> and <code>&lt;ff&gt;</code>.</td>
<td>v:3.2* — finds any issue whose version number is (for example): 3.2, 3.2-beta, 3.2.1, 3.2.x</td>
</tr>
</tbody>
</table>

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 **simple search Query** 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 8, you can add your JIRA site as a search engine/provider via the dropdown menu next to the browser's search box. 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.

Advanced Searching

The instructions on this page describe how to define and execute a search using the advanced search. You can also define and execute a search using the quick search or using basic 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**. For 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**.
On this page:
- What is an Advanced Search?
- How to Perform an Advanced Search
  - Performing Text Searches
  - Using Auto-complete
  - Switching between 'Advanced' and 'Simple' Search
  - Setting Precedence of Operators
- Keywords Reference
- List of Keywords
- Operators Reference
- Fields Reference

Related topics:
- Basic Searching
- Using Quick Search
- Performing Text Searches

How to Perform an Advanced Search

1. Choose Issues > Search for Issues. The issue navigator will be displayed.
   - If there are existing search criteria, click the New filter button to reset the search criteria.
   - If the Advanced link is showing, click it to switch to advanced searching.
2. Type your query using the fields, operators and field values or functions.
3. Click the Search button to run your query.

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:

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.

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', particularly 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'.

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.

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.

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

- Find all open, urgent issues that are assigned to jsmith:
  
  ```javascript
  status = open and priority = urgent and assignee = jsmith
  ```

- Find all issues in a particular project that are not assigned to jsmith:
  
  ```javascript
  project = JRA and assignee != jsmith
  ```

- Find all issues for a specific release which consists of different version numbers across several projects:
  
  ```javascript
  project in (JRA,CONF) and fixVersion = "3.14"
  ```

- Find all issues where neither the Reporter nor the Assignee is Jack, Jill or John:
  
  ```javascript
  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:
  
  ```javascript
  reporter = jsmith or reporter = jbrown
  ```

- Find all issues that are overdue or where no due date is set:
  
  ```javascript
  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:
  
  ```javascript
  assignee not in (jsmith)
  ```
**Find all issues that were not created by either jsmith or jbrown:**

\[
\text{not (reporter = jsmith or reporter = jbrown)}
\]

### 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:

  \[
  \text{duedate = empty}
  \]
  
  or
  
  \[
  \text{duedate is empty}
  \]

### 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:

  \[
  \text{duedate = null}
  \]
  
  or
  
  \[
  \text{duedate is null}
  \]

### 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

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.

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:
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
```
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
  ```

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"
  ```

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:

\[\text{votes} < 4\]

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:

  \[\text{votes} \leq 4\]

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

  \[\text{updated} \leq "-4w 2d"\]

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 \textsc{equals} \textsc{( = )} statements, but is shorter and more convenient. That is, typing \texttt{reporter IN (tom, jane, harry)} is the same as typing \texttt{reporter = "tom" OR reporter = "jane" OR reporter = "harry".}

Examples

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

  \[\text{reporter in (jsmith, jbrown, jones)}\]

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

  \[\text{reporter in (Jack, Jill)} \text{ or assignee in (Jack, Jill)}\]

- Find all issues in version 3.14 or version 4.2:

  \[\text{affectedVersion in ("3.14", "4.2")}\]

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
  ```

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 word "issue" and the word "collector":

```
summary ~ "issue collector"
```

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
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
  ```

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 "2011/02/02"
  ```

- Find issues that were resolved by Joe Smith during 2010:
  
  ```
  status WAS "Resolved" BY jsmith DURING ("2010/01/01","2011/01/01")
  ```

```
**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")
  ```

**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:
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 "2011/02/02"`

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()

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.

List of Fields:

- Affected Version
- Assignee
- Category
- Comment
- Component
- Created
- Custom Field
- Description
- Due
- Environment
- Epic Link
- Filter
- Fix Version
- Issue Key
- LastViewed
- Level
- Original Estimate
- Parent
- Priority
- Project
- Remaining Estimate
- Reporter
- Resolution
- Resolved
- Sprint
- 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).

\[
\text{It is safer to search by version ID than by version name}
\]

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

\[
\text{affectedVersion}
\]

Field Type

VERSION

Supported Operators

<table>
<thead>
<tr>
<th></th>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS IN</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>IS</td>
<td>NOT</td>
<td>IN</td>
<td>NOT IN</td>
<td>WAS</td>
<td>WAS IN</td>
<td>WAS NOT</td>
<td>WAS NOT IN</td>
<td>CHANGED</td>
</tr>
</tbody>
</table>

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 Functions

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

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

Examples

- Find issues with an AffectedVersion of 3.14:

\[
\text{affectedVersion = "3.14"}
\]

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

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

\[
\text{affectedVersion = "Big Ted"}
\]

- Find issues with an AffectedVersion ID of 10350:

\[
\text{affectedVersion = 10350}
\]
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 | NOT | IN | NOT | IN | WAS | WAS | NOT | NOT | IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | × | × | × | × | × | × | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ▼|

Supported Functions

When 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.)
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 | WA | WAS | WA | S NOT | WA | S NOT | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | ✓ | x | x | x | x | x | ✓ | ✓ | ✓ | ✓ | x | x | x | x | x | x |

Supported Functions

n/a

Examples

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

  category = "Alphabet Projects"

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 | WA | WAS | WA | S NOT | WA | S NOT | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| x | x | ✓ | ✓ | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

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

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WA S</th>
<th>WA S NOT</th>
<th>WA S NOT IN</th>
<th>CH A N GE D</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

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)`
• 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.
Syntx

```
created
```

Alias:

```
createdDate
```

Field Type

DATE

Supported Operators

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WA</th>
<th>WAS</th>
<th>WA S</th>
<th>WA S NOT</th>
<th>WA S NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
</tr>
</tbody>
</table>

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()
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 | NOT | IN | NOT | W | AS | W | AS | W | AS | NOT | IN | CHANGED |
  |---|----|---|-----|---|-----|---|-----|---|-----|---|-----|---|---|---|---|---|-----|---|
  | ✔ | ✔ | ❌ | ❌ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ❌ | ❌ |

- Picker, select, check-box and radio button fields:

  | = | != | ~ | !~ | > | >= | < | <= | IS | NOT | IN | NOT | W | AS | W | AS | W | AS | NOT | IN | CHANGED |
  |---|----|---|-----|---|-----|---|-----|---|-----|---|-----|---|---|---|---|---|-----|---|
  | ✔ | ✔ | ❌ | ❌ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ❌ | ❌ |

- Text fields:

  | = | != | ~ | !~ | > | >= | < | <= | IS | NOT | IN | NOT | W | AS | W | AS | W | AS | 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()
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":

  ```
  ```

- Find issues where the "Location" Custom Field has no value:

  ```
  location != empty
  ```

Search for issues where the Description contains particular text.

**Description**

JIRA text-search syntax can be used.

Note: this field does not support auto-complete.

**Syntax**

```text
description
```
• Find issues where the Description contains the exact phrase "Please see screenshot":

\[ \text{description} \sim \text{"Please see screenshot"} \]

**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

\[ \text{due} \]

Alias:

\[ \text{dueDate} \]

**Field Type**

DATE

**Supported Operators**

| = | != | ~ | !~ | > | >= | < | <= | IS | IS NOT | IN | NOT IN | WA | WAS | WA S NOT | WA S 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 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:

\[
due >= "2011/01/01" \text{ and } due <= "2011/01/31"
\]

• 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

<table>
<thead>
<tr>
<th>=</th>
<th>!</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT</th>
<th>IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Supported Functions

n/a

Examples

• Find issues where the Environment contains text that matches “Third floor” (i.e. a “fuzzy” match):
Find issues where the Environment contains the exact phrase "Third floor":

```
environment ~ "Third floor"
```

Epic Link

ℹ️ Only available if you are using JIRA Agile 6.1.2 or later.

Search for issues that belong to a particular Epic in JIRA Agile. The search is based on either the epic's Name, 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

```
"epic link"
```

Field Type

Epic Link Relationship (this is a custom type created by JIRA Agile).

Supported Operators

```
<table>
<thead>
<tr>
<th>Operator</th>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>S NOT</th>
<th>WAS</th>
<th>NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td></td>
</tr>
</tbody>
</table>
```

Supported Functions

When used with the IN or NOT IN operators, **epic link** supports:

- `issueHistory()`
- `linkedIssues()`
- `votedIssues()`
- `watchedIssues()`

Examples

- Find issues that belong to epic "Jupiter", which has issue key ANERDS-317:
  
  ```
  "epic link" = ANERDS-317
  ```

  or

  ```
  "epic link" = Jupiter
  ```
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**

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WA</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

**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
```
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 | 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 Functions

When 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
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

```plaintext
issueKey
```

Aliases:

```plaintext
id
issue
key
```

Field Type

ISSUE

Supported Operators

| = | != | ~ | !~ | > | >= | < | <= | IS | NOT | IN | NOT | IN | WAS | SIN | WAS | NOT | 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":

  ```plaintext
  issueKey = ABC-123
  ```

LastViewed

Search for issues that were last viewed 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

```
lastViewed
```

Field Type
DATE

Supported Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>✓</td>
</tr>
<tr>
<td>!=</td>
<td>✓</td>
</tr>
<tr>
<td>~</td>
<td>x</td>
</tr>
<tr>
<td>!~</td>
<td>x</td>
</tr>
<tr>
<td>&gt;</td>
<td>✓</td>
</tr>
<tr>
<td>&gt;=</td>
<td>✓</td>
</tr>
<tr>
<td>&lt;</td>
<td>✓</td>
</tr>
<tr>
<td>&lt;=</td>
<td>✓</td>
</tr>
<tr>
<td>IS</td>
<td>✓</td>
</tr>
<tr>
<td>IS NOT</td>
<td>✓</td>
</tr>
<tr>
<td>IN</td>
<td>✓</td>
</tr>
<tr>
<td>NOT IN</td>
<td>✓</td>
</tr>
<tr>
<td>WAS</td>
<td>✓</td>
</tr>
<tr>
<td>WAS NOT</td>
<td>✓</td>
</tr>
<tr>
<td>WAS IN</td>
<td>✓</td>
</tr>
<tr>
<td>WAS NOT IN</td>
<td>✓</td>
</tr>
</tbody>
</table>

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 last viewed before 12th December 2010:

  ```
  lastViewed < "2010/12/12"
  ```

- Find all issues last viewed on or before 12th December 2010:

  ```
  lastViewed <= "2010/12/13"
  ```

- Find all issues last viewed on 12th December 2010 before 2:00pm:

  ```
  lastViewed > "2010/12/12" and created < "2010/12/12 14:00"
  ```

- Find issues last viewed less than one day ago:
Find issues last viewed in January 2011:

```
lastViewed > "2011/01/01" and created < "2011/02/01"
```

Find issues last viewed on 15 January 2011:

```
lastViewed > "2011/01/15" and created < "2011/01/16"
```

### 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 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", lev1)
  ```

- Search for issues with a Security Level ID of 123:
level = 123

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

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!=</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

Supported Functions

n/a

Examples

- Find issues with an Original Estimate of 1 hour:

  \[ \text{originalEstimate} = 1h \]

- Find issues with an Original Estimate of more than 2 days:

  \[ \text{originalEstimate} > 2d \]

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 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 Priority 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 NOT | WAS NOT IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Supported Functions**

It is safer to search by Priority 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.
Examples

- Find issues with a Priority of "High":

\[
\text{priority} = \text{High}
\]

- Find issues with a Priority ID of 10000:

\[
\text{priority} = 10000
\]

\[^{top \ of \ fields} \ | \ ^{top \ of \ topic}\]

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). In the rare case where there is a project whose project key is the same as another project's name, then the project key takes preference and hides results from the second project.

Note: this field supports auto-complete.

Syntax

\[
\text{project}
\]

Field Type

PROJECT

Supported Operators

| = | != | ~ | != | > | >= | < | <= | IS | NOT | IN | NOT | IN | WAS | NOT | WAS | NOT | 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":

\[
\text{project} = \text{"ABC Project"}
\]

- Find issues that belong to the Project that has the key "ABC":

\[
\text{project} = \text{"ABC"}
\]

- Find issues that belong to the Project that has the ID "1234":

\[
\text{project} = \text{"1234"}
\]
**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**

```plaintext
remainingEstimate
```

**Alias:**

```plaintext
timeEstimate
```

**Field Type**

**DURATION**

**Supported Operators**

| = | != | ~ | != | > | >= | < | <= | IS | NOT | IN | NOT | IN | WAS | NOT | WAS | NOT | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ☑ | ☑ | ☑ | ☑ | ☑ | ☑ | ☑ | ☑ |

**Supported Functions**

n/a

**Examples**

- Find issues with a Remaining Estimate of more than 4 hours:

  ```plaintext
  remainingEstimate > 4h
  ```

**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**

```plaintext
reporter
```
Field Type

**USER**

Supported Operators

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!=</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WA S</th>
<th>WA S NOT</th>
<th>WA S NOT IN</th>
<th>CH AN GE D</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

Supported Functions

When 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":

  ```
  reporter = "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**
Field Type

RESOLUTION

Supported Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Supported Functions</th>
</tr>
</thead>
</table>

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

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

<table>
<thead>
<tr>
<th></th>
<th>!=</th>
<th>~</th>
<th>!=</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>❌</td>
</tr>
</tbody>
</table>

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:
  
  ```text
  resolved <= "2010/12/31"
  ```

- Find all issues that were resolved before 2.00pm on 31st December 2010:
  
  ```text
  resolved < "2010/12/31 14:00"
  ```

- Find all issues that were resolved on or before 31st December 2010:
  
  ```text
  resolved <= "2011/01/01"
  ```

- Find issues that were resolved in January 2011:
  
  ```text
  resolved > "2011/01/01" and resolved <= "2011/02/01"
  ```

- Find issues that were resolved on 15 January 2011:
  
  ```text
  resolved > "2011/01/15" and resolved < "2011/01/16"
  ```
• Find issues that were resolved in the last hour:

```
resolved > -1h
```

---

**Sprint**

ℹ️ Only available if you are using **JIRA Agile**.

Search for issues that are assigned to a particular **sprint** in JIRA Agile. This works for active sprints and future sprints. The search is based on either the sprint name or the sprint ID (i.e. the number that JIRA automatically allocates to a sprint).

**Syntax**

```
sprint
```

ℹ️ If you have multiple sprints with similar (or identical) names, you can simply search by using the sprint name — or even just part of it. The possible matches will be shown in the autocomplete drop-down, with the sprint dates shown to help you distinguish between them. (The sprint ID will also be shown, in brackets).

**Field Type**

- **Number**

**Supported Operators**

<table>
<thead>
<tr>
<th>Operator</th>
<th>Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>✔</td>
</tr>
<tr>
<td>!=</td>
<td>✔</td>
</tr>
<tr>
<td>~</td>
<td>✗</td>
</tr>
<tr>
<td>!~</td>
<td>✗</td>
</tr>
<tr>
<td>&gt;</td>
<td>✗</td>
</tr>
<tr>
<td>&gt;=</td>
<td>✗</td>
</tr>
<tr>
<td>&lt;</td>
<td>✗</td>
</tr>
<tr>
<td>&lt;=</td>
<td>✗</td>
</tr>
<tr>
<td>IS</td>
<td>✔</td>
</tr>
<tr>
<td>NOT</td>
<td>✗</td>
</tr>
<tr>
<td>IN</td>
<td>✔</td>
</tr>
<tr>
<td>NOT IN</td>
<td>✗</td>
</tr>
<tr>
<td>WAS</td>
<td>✔</td>
</tr>
<tr>
<td>NOT WAS</td>
<td>✗</td>
</tr>
<tr>
<td>WA</td>
<td>✔</td>
</tr>
<tr>
<td>NOT WA</td>
<td>✗</td>
</tr>
<tr>
<td>IN WA</td>
<td>✔</td>
</tr>
<tr>
<td>NOT IN WA</td>
<td>✗</td>
</tr>
<tr>
<td>CHANGED</td>
<td>✗</td>
</tr>
</tbody>
</table>

**Supported Functions**

- `openSprints()`
- `closedSprints()`

**Examples**

• Find issues that belong to sprint 999:

```
sprint = 999
```

• Find issues that belong to sprint "February 1":

```
sprint = "February 1"
```

• Find issues that belong to either "February 1", "February 2" or "February 3":

```
sprint in ("February 1","February 2","February 3")
```

• Find issues that are assigned to a sprint:
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 NOT | WAS IN | 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 = Open
  ```
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**

```markdown
summary
```

**Field Type**

**TEXT**

**Supported Operators**

<table>
<thead>
<tr>
<th></th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

**Supported Functions**

n/a

**Examples**

- Find issues where the Summary contains text that matches "Error saving file" (i.e. a "fuzzy" match):

  ```markdown
  summary ~ "Error saving file"
  ```

- Find issues where the Summary contains the exact phrase "Error saving file":

  ```markdown
  summary ~ '""Error saving file""
  ```

**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
Notes:

- The **text** master-field can only be used with the **CONTAINS** operator ("~" and "!~").
- JIRA text-search syntax can be used with these fields.

**Syntax**

```text

```

**Field Type**

**TEXT**

**Supported Operators**

<table>
<thead>
<tr>
<th>Supported Operators</th>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
</table>

**Supported Functions**

n/a

**Examples**

- Find issues where a text field matches the word "Fred":

  ```text
text ~ "Fred"
  ```

  or

  ```text
text ~ Fred
  ```

- Find all issues where a text field contains the exact phrase "full screen":

  ```text
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**

```text

```
Alias:

issueType

Field Type

ISSUE_TYPE

Supported Operators

| = | != | ~ | !~ | > | >= | < | <= | IS | NOT | IN | NOT IN | WAS | 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
Supported Functions

n/a

Examples

- Find issues where the Time Spent is more than 5 days:

  ```
  timeSpent > 5d
  ```

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

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 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 > "2011/01/01" 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
```
USER

Supported Operators

<p>| | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Supported Functions

When 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:
  
  \[
  \text{voter} = \text{currentUser}()
  \]

- Search for issues for which the user "jsmith" has voted:
  
  \[
  \text{voter} = \text{"jsmith"}
  \]

- Search for issues for which a member of the group "jira-developers" has voted:
  
  \[
  \text{voter in membersOf("jira-developers")}
  \]

^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

\[
\text{votes}
\]

Field Type

NUMBER

Supported Operators

<p>| | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Supported Functions

n/a
Examples

- Find all issues that have 12 or more votes:
  
  \[\text{votes} \geq 12\]

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

\[\text{watcher}\]

Field Type

USER

Supported Operators

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| = | != | ~ | ^= | > | >= | < | <= | IS | IS NOT | IN | NOT IN | WAS | WAS NOT | WAS NOT IN | CHANGED |
| ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |

Supported Functions

When 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:
  
  \[\text{watcher} = \text{currentUser()}\]

- Search for issues that the user "jsmith" is watching:
  
  \[\text{watcher} = \text{"jsmith"}\]

- Search for issues that are being watched by a member of the group "jira-developers":
  
  \[\text{watcher in membersOf("jira-developers")}\]
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 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: \( \text{workRatio} = \frac{\text{timeSpent}}{\text{originalEstimate}} \times 100 \)

Note: this field does not support auto-complete.
Syntax

```
workRatio
```

Field Type

NUMBER

Supported Operators

| = | != | ~ | !~ | > | >= | < | <= | IS | IS NOT | IN | NOT IN | WAS | 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:

\[
\text{workRatio} > 75
\]

Advanced Searching Functions

The instructions on this page describe how to use functions in JQL to define structured search 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).

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.

This document also covers the reserved characters and words that JIRA uses.

List of Functions

- cascadeOption()
- closedSprints()
- componentsLeadByUser()
- currentLogin()
- currentUser()
- earliestUnreleasedVersion()
- endOfDay()
- endOfMonth()
- endOfWeek()
- endOfYear()
- issueHistory()
- lastLogin()
- latestReleasedVersion()
- linkedIssues()
- membersOf()
- now()
- openSprints()
- projectsLeadByUser()
- projectsWhereUserHasPermission()
- projectsWhereUserHasRole()
- releasedVersions()
- standardIssueTypes()
- startOfDay()
- startOfMonth()
- startOfWeek()
- startOfYear()
- subtaskIssueTypes()
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 NOT | IN  | NOT IN | WA S | WA S NOT | WA S NOT IN | CH AN GE D |
|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |  |  |  |
| x | x | x | x | x | x | x | x | x | x | x | x |
```

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:

```java
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:

```java
referrer in cascadeOption("\"none\",none)
```

^top of functions | ^^top of topic

### closedSprints()

**i** Only available if you are using JIRA Agile.

Search for issues that are assigned to a completed Sprint. (Note that it is possible for an issue to belong to both a completed Sprint(s) and an incomplete Sprint(s).)

See also `openSprints()`.

Syntax

```java
closedSprints()
```

**Supported Fields**

- **Sprint**

**Supported Operators**

<table>
<thead>
<tr>
<th>Equal</th>
<th>Not Equal</th>
<th>Greater than</th>
<th>Greater than or equal to</th>
<th>Less than</th>
<th>Less than or equal to</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>☑</td>
</tr>
</tbody>
</table>

**Examples**

- Find all issues that are assigned to a completed Sprint.

```java
sprint in closedSprints()
```

^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 | NOT | IN | NOT IN | WA | S | NOT | WA | NOT | 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()
Examples

- Find issues that have been created during my current session:
  
  ```
  created > currentLogin()
  ```

**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 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()
  ```

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

```plaintext
earliestUnreleasedVersion(project)
```

Supported Fields

- **AffectedVersion**
- **FixVersion**
- custom fields of type Version

Supported Operators

| = | != | ~ | !~ | > | >= | < | <= | IS | NOT | IN | NOT | WA | S | WA | NOT | WA | NOT | CHANGED |
| ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |

Examples

- Find issues whose **FixVersion** is the earliest unreleased version of the ABC project:

  ```plaintext
  fixVersion = earliestUnreleasedVersion(ABC)
  ```

- Find issues that relate to the earliest unreleased version of the ABC project:

  ```plaintext
  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

```plaintext
endOfDay()
```

or

```plaintext
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 NOT | IN | NOT IN | WA S | WA S NOT | WA S | NO T | CH AN GE D |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✔ | ✔ | ✗ | ✗ | ✔ | ✔ | ✔ | ✔ | ✗ | ✗ | ✗ | (onl y in pre dic ate) | (onl y in pre dic ate) | (onl y in pre dic ate) | (onl y in pre dic ate) |

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

```java
endOfMonth()
```

or

```java
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
Examples

- Find issues due by the end of this month:

  ```java
due < endOfMonth()
  ```

- Find issues due by the end of next month:

  ```java
due endOfMonth("+1")
  ```

- Find issues due by the 15th of next month:

  ```java
due endOfMonth("+15d")
  ```

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

```java
endOfWeek()
```
or

```java
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
### Examples

- Find issues due by the end of this week:

```java
due < endOfWeek()
```

- Find issues due by the end of next week:

```java
due < endOfWeek("+1")
```

---

### endOfYear()

Perform searches based on the end of the current year. See also startOfDay, startOfWeek and startOfMonth; and endOfDay, endOfWeek, endOfMonth and endOfYear.

```java
startOfYear()
```

or

```java
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("+l")` is the same as `endofYear("+ly")`.
- If the plus/minus `(+/-)` sign is omitted, plus is assumed.

### Supported Fields

- `Created`
- `Due`
- `Resolved`
- `Updated`
- `custom` fields of type Date/Time

### Supported Operators

```
<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

(only in predicate) (only in predictate) (only in predictate) (only in predictate)
```

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>!=</td>
<td>~</td>
<td>!~</td>
<td>&gt;</td>
<td>&gt;=</td>
<td>&lt;</td>
<td>&lt;=</td>
<td>IS</td>
<td>NOT</td>
</tr>
<tr>
<td>IN</td>
<td>NOT</td>
<td>IN</td>
<td>WAS</td>
<td>WA</td>
<td>NOT</td>
<td>WAS</td>
<td>NOT</td>
<td>IN</td>
<td>CHANGED</td>
</tr>
</tbody>
</table>

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
lastLogin()

Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT</th>
<th>WA S</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>^</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(only in predicate)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Examples

- Find issues that have been created during my last session:

  created > lastLogin()

\(^\text{top of functions} \, ^\text{top of topic}\)

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

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT</th>
<th>WA S</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(only in predicate)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Examples

- Find issues whose FixVersion is 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)

### 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**

```python
linkedIssues(issueKey)
```

or

```python
linkedIssues(issueKey,linkType)
```

### Supported Fields

- **Issue**

### Supported Operators

<table>
<thead>
<tr>
<th></th>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

### Examples

- Find issues that are linked to a particular issue:

  ```python
  issue in linkedIssues(ABC-123)
  ```

- Find issues that are linked to a particular issue via a particular type of link:

  ```python
  issue in linkedIssues(ABC-123,"is duplicated by")
  ```

### membersOf()

Perform searches based on the members of a particular group.

**Syntax**

```python
membersOf()
```
Supported Fields

- Assignee
- Reporter
- Voter
- Watcher
- custom fields of type User

Supported Operators

| = | != | ~ | !~ | > | >= | < | <= | IS | NOT | IN | NOT IN | WAS | NOT | WAS | NOT | IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

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)
  ```

now()

Perform searches based on the current time.

Syntax

```
now()
```

Supported Fields

- Created
- Due
- Resolved
- Updated
**custom fields of type Date/Time**

<table>
<thead>
<tr>
<th>Supported Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
</tr>
<tr>
<td>✓</td>
</tr>
</tbody>
</table>

**Examples**

- Find issues that are overdue:

  
  
  duedate < now() and status not in {closed, resolved}

**openSprints()**

- Only available if you are using JIRA Agile.

Search for issues that are assigned to a Sprint which has not yet been completed. (Note that it is possible for an issue to belong to both a completed Sprint(s) and an incomplete Sprint(s).)

See also `closedSprints()`.

**Syntax**

```
openSprints()
```

**Supported Fields**

- `Sprint`

**Supported Operators**

<table>
<thead>
<tr>
<th>Supported Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
</tr>
<tr>
<td>✗</td>
</tr>
</tbody>
</table>

**Examples**

- Find all issues that are assigned to a Sprint which has not yet been completed.

  
  
  sprint in openSprints()
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

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!=~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>NOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Project Permissions</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Permission</td>
<td>Explanation</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Administer Projects</td>
<td>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').</td>
</tr>
<tr>
<td>Browse Projects</td>
<td>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.</td>
</tr>
<tr>
<td>View Development Tools</td>
<td>Permission to view the related source code commits (e.g. CVS, Subversion, FishEye, etc) for an issue, in a 'Source' tab. Note that for CVS, to view the related source code commits, the project needs to be associated with at least one Repository.</td>
</tr>
<tr>
<td>View (Read-Only) Workflow</td>
<td>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.</td>
</tr>
</tbody>
</table>

### Issue Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign Issues</td>
<td>Permission to assign issues to users. Also allows autocompletion of users in the Assign Issue dropdown. (See also Assignable User permission below)</td>
</tr>
<tr>
<td>Assignable User</td>
<td>Permission to be assigned issues. (Note that this does not include the ability to assign issues; see Assign Issue permission above).</td>
</tr>
<tr>
<td>Close Issues</td>
<td>Permission to close issues. (This permission is useful where, for example, developers resolve issues and testers close them). Also see the Resolve Issues permission.</td>
</tr>
<tr>
<td>Create Issues</td>
<td>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).</td>
</tr>
<tr>
<td>Delete Issues</td>
<td>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.</td>
</tr>
<tr>
<td>Role</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Edit Issues</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Link Issues</strong></td>
<td>Permission to link issues together. (Only relevant if Issue Linking is enabled).</td>
</tr>
<tr>
<td><strong>Modify Reporter</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Move Issues</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Resolve Issues</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Schedule Issues</strong></td>
<td>Permission to schedule an issue — that is, to edit the 'Due Date' of an issue. In older versions of JIRA this also controlled the permission to view the 'Due Date' of an issue.</td>
</tr>
<tr>
<td><strong>Set Issue Security</strong></td>
<td>Permission to set the security level on an issue to control who can access the issue. Only relevant if issue security has been enabled.</td>
</tr>
<tr>
<td><strong>Voters &amp; Watchers Permissions</strong></td>
<td><strong>Explanation</strong></td>
</tr>
<tr>
<td><strong>Manage Watcher List</strong></td>
<td>Permission to manage (i.e. view/add/remove users to/from) the watcher list of an issue.</td>
</tr>
<tr>
<td><strong>View Voters and Watchers</strong></td>
<td>Permission to view the voter list and watcher list of an issue. Also see the Manage Watcher List permission.</td>
</tr>
<tr>
<td><strong>Comments Permissions</strong></td>
<td><strong>Explanation</strong></td>
</tr>
<tr>
<td><strong>Add Comments</strong></td>
<td>Permission to add comments to issues. Note that this does not include the ability to edit or delete comments.</td>
</tr>
<tr>
<td><strong>Delete All Comments</strong></td>
<td>Permission to delete any comments, regardless of who added them.</td>
</tr>
<tr>
<td><strong>Delete Own Comments</strong></td>
<td>Permission to delete comments that were added by the user.</td>
</tr>
<tr>
<td><strong>Edit All Comments</strong></td>
<td>Permission to edit any comments, regardless of who added them.</td>
</tr>
<tr>
<td><strong>Edit Own Comments</strong></td>
<td>Permission to edit comments that were added by the user.</td>
</tr>
</tbody>
</table>
### Attachments Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Attachments</td>
<td>Permission to attach files to an issue. (Only relevant if attachments are enabled). Note that this does not include the ability to delete attachments.</td>
</tr>
<tr>
<td>Delete All Attachments</td>
<td>Permission to delete any attachments, regardless of who added them.</td>
</tr>
<tr>
<td>Delete Own Attachments</td>
<td>Permission to delete attachments that were added by the user.</td>
</tr>
</tbody>
</table>

### Time Tracking Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work On Issues</td>
<td>Permission to log work against an issue, i.e. create a worklog entry. (Only relevant if Time Tracking is enabled).</td>
</tr>
<tr>
<td>Delete All Worklogs</td>
<td>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.</td>
</tr>
<tr>
<td>Delete Own Worklogs</td>
<td>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.</td>
</tr>
<tr>
<td>Edit All Worklogs</td>
<td>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.</td>
</tr>
<tr>
<td>Edit Own Worklogs</td>
<td>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.</td>
</tr>
</tbody>
</table>

### Supported Fields

- **Project**

### Supported Operators

- `=`
- `!=` (not equal)
- `~` (approximately equals)
- `!~` (not approximately equals)
- `>`
- `>=`
- `<`
- `<=`
- `IS`
- `IS NOT`
- `IN`
- `NOT IN`
- `WA`
- `WA NOT`
- `WA IN`
- `WA NOT IN`
- `CHANGED`

### Examples

- Find open issues in projects where you have the "Resolve Issues" permission:

  ```plaintext
  project in projectsWhereUserHasPermission("Resolve Issues") AND status = Open
  ```

- Find issues in projects where you have a specific role.

  ```plaintext
  projectsWhereUserHasRole()
  ```

  Note that this function is only available if you are logged in to JIRA.

Syntax
projectsWhereUserHasRole(rolename)

Supported Fields
- Project

Supported Operators

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Examples
- Find open issues in projects where you have the "Developers" role:

  project in projectsWhereUserHasRole("Developers") AND status = Open

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

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Examples
- Find issues whose FixVersion is a released version of the ABC project:
Find issues that relate to released versions of the ABC project:

```java
(affectedVersion in releasedVersions(ABC)) or (fixVersion in releasedVersions(ABC))
```

**standardIssueTypes()**

Perform searches based on "standard" Issue Types, that is, search for issues which are not sub-tasks.

See also `subtaskIssueTypes()`.

**Syntax**

```java
standardIssueTypes()
```

**Supported Fields**

- **Type**

**Supported Operators**

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td></td>
<td>✗</td>
<td></td>
<td>✗</td>
<td></td>
<td>✗</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Examples**

- Find issues that are not subtasks (i.e. issues whose Issue Type is a standard issue type, not a subtask issue type):

  ```java
  issuetype in standardIssueTypes()
  ```

**startOfDay()**

Perform searches based on the start of the current day. See also `startOfWeek`, `startOfMonth` and `startOfYear`; and `endOfDay`, `endOfWeek`, `endOfMonth` and `endOfYear`.

**Syntax**

```java
startOfDay()
```

or

```java
startOfDay("inc")
```
where \( \text{inc} \) 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. \(\text{startOfDay}('+1')\) is the same as \(\text{startOfDay}('+1d')\).
- 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 NOT | WAS NOT IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | (only in predicate) |

**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")
  ```

**startOfMonth()**

Perform searches based on the start of the current month. See also \(\text{startOfDay}, \text{startOfWeek}\) and \(\text{startOfYear}\); and \(\text{endOfDay}, \text{endOfWeek}, \text{endOfMonth}\) and \(\text{endOfYear}\).

**Syntax**

```
startOfMonth()
```

or

```
startOfMonth("inc")
```

where \( \text{inc} \) 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 | WA S | WAS | WA S NOT | WAS NOT | CH AN GE D |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | x | x | ✓ | ✓ | ✓ | ✓ | x | x | x | x | (onl y in pre dic ate) | (onl y in pre dic ate) | (onl y in pre dic ate) | (onl y in pre dic ate) |

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.

See Setting Locale in JIRA for more information.

Syntax

```
startOfWeek()
```

or
startOfWeek("inc")

where \textit{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

<table>
<thead>
<tr>
<th></th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT IN</th>
<th>WAS</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Examples

- Find new issues since the start of this week:
  
  \[
  \text{created} > \text{startOfWeek()}
  \]

- Find new issues since the start of last week:

  \[
  \text{created} > \text{startOfWeek("-1")}
  \]

\textit{top of functions} | \textit{top of topic}

\textbf{startOfYear()}

Perform searches based on the start of the current year. See also \texttt{startOfDay}, \texttt{startOfWeek} and \texttt{startOfMonth}; and \texttt{endOfDay}, \texttt{endOfWeek}, \texttt{endOfMonth} and \texttt{endOfYear}.

\[
\text{startOfYear()}
\]

or

\[
\text{startOfYear("inc")}
\]

where \textit{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").
Supported Fields

- Created
- Due
- Resolved
- Updated
- custom fields of type Date/Time

Supported Operators

| = | != | ~ | !~ | > | >= | < | <= | IS | IS NOT | IN | NOT IN | WAS | WAS NOT | WAS NOT IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✔️ | ✔️ | ❌ | ❌ | ✔️ | ✔️ | ✔️ | ✔️ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ |

Examples

- Find new issues since the start of this year:

  created > startOfYear()

- Find new issues since the start of last year:

  created > startOfYear("-1")

subtaskIssueTypes()

Perform searches based on issues which are sub-tasks.

See also standardIssueTypes().

Syntax

subtaskIssueTypes()
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 | NOT | IN | NOT IN | WA | S | WA | S NOT | WA | S NOT IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |

Examples

- Find issues whose FixVersion is 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)
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 | WA | S | WAS | NOT | 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 | WA | S | WAS | NOT | WAS NOT | WAS NOT IN | CHANGED |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Examples**

- Find issues that you are watching:

```
issue in watchedIssues()
```

**Supported Fields**

- Issue
Supported Operators

<table>
<thead>
<tr>
<th>=</th>
<th>!=</th>
<th>~</th>
<th>!~</th>
<th>&gt;</th>
<th>&gt;=</th>
<th>&lt;</th>
<th>&lt;=</th>
<th>IS</th>
<th>IS NOT</th>
<th>IN</th>
<th>NOT</th>
<th>WAS</th>
<th>WAS IN</th>
<th>WAS NOT</th>
<th>WAS NOT IN</th>
<th>CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Examples

- Find issues that you have recently viewed:

```java
issue in issueHistory()
```

![top of functions](#) | ![top of topic](#)

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:
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", "check", "checkpoint", "collate", "collation", "column", "commit", "connect", "continue",
"count", "create", "current", "date", "decimal", "declare", "decrement", "default", "defaults", "define", "delete",
"delimiter", "desc", "difference", "distinct", "divide", "do", "double", "drop", "else", "empty", "encoding",
"first", "float", "for", "from", "function", "go", "goto", "grant", "greater", "group", "having",
"identified", "if", "immediate", "in", "increment", "index", "initial", "inner", "inout", "input", "insert",
"int", "integer", "intersect", "intersection", "into", "is", "isempty", "isnull", "join", "last", "left",
"less", "like", "limit", "lock", "long", "max", "min", "minus", "mode", "modify",
"modulo", "more", "multiply", "next", "noaudit", "not", "notin", "nowait", "null", "number", "object",
"of", "on", "option", "or", "order", "outer", "output", "power", "previous", "prior", "privileges",
"public", "raise", "raw", "remainder", "rename", "resource", "return", "returns", "revoke", "right", "row",
"rowid", "rownum", "rows", "select", "session", "set", "share", "size", "sqrt", "start", "strict",
"string", "subtract", "sum", "synonym", "table", "then", "to", "trans", "transaction", "trigger", "true",
"uid", "union", "unique", "update", "user", "validate", "values", "view", "when", "whenever", "where",
"while", "with"

You can use either single quote-marks (') or double quote-marks (").

(Note for JIRA administrators: this list is hard coded in the JqlStringSupportImpl.java file.)

Performing Text Searches

This page provides information on how to perform text searches. It applies to both basic searches and advanced searches (when used with the CONTAINS operator). This page also applies to quick search when performing a text search on the fields that this feature supports.

Acknowledgements:
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). If you combine multiple terms without specifying any Boolean operators, they will be joined using AND operators.

**Note:** All query terms in JIRA are case insensitive.

### 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). If you combine multiple terms without specifying any Boolean operators, they will be joined using AND operators.

**Note:** All query terms in JIRA are case insensitive.

### On this page:

- Query terms
- Term modifiers
- Boosting a term: ^
- Boolean operators
- Grouping
- Escaping special characters: \ or \\
- Reserved words
- Word stemming
- Limitations

### Related topics:

- Searching for Issues
- Using Quick Search
- Basic Searching
- Advanced Searching

### Term modifiers

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

**Wildcard searches:** ? and * | **Fuzzy searches:** ~ | **Proximity searches**

**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.

**Wildcards 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:

```
tet?
```

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. The feature request for this is JIRA-6218.

**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 that are 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.

**OR | AND | Required term: + | NOT | Excluded term: -**

**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 "confluence" use the query:

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

or

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

**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"
```
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 bugs and either atlassian or jira, 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: \ or \\

⚠ Please be aware that due to a bug in JIRA (JIRA-25092), it is currently not possible to search issues for most of the special characters mentioned below, even if they have been properly escaped in your query. The only characters which currently work with search when properly escaped are:

+ - & | ~ * : 

JIRA supports the ability to search issues for special characters by escaping them in your query syntax. The current list of such characters is:

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

To escape these characters, type a backslash character '\' before the special character (or if using Advanced Searching, type two backslashes '\\' before the special character).

For example, to search for (1+1) in either a simple or quick search, use the query:

\(1\+1\)

and to search for [example] in the summary of an advanced search (in JIRA Query Language or JQL), use the query:
Please note: If you are using Advanced Searching — please see Reserved Characters for more information about how these characters and others are escaped in JIRA Query Language.

Reserved words

To keep the search index size and search performance optimal in JIRA, the following English reserved words (also known as 'stop words') are ignored from the search index and hence, JIRA’s text search features:

"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"

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

Your JIRA administrator can make JIRA index these reserved words (so that JIRA will find issues based on the presence of these words) by changing the Indexing Language to Other (under Administration > System > General Configuration).

Word stemming

Since JIRA cannot search for issues containing parts of words (see below), word 'stemming' allows you to retrieve issues from a search based on the 'root' (or 'stem') forms of words instead of requiring an exact match with specific forms of these words. The number of issues retrieved from a search based on a stemmed word is typically larger, since any other issues containing words that are stemmed back to the same root, will also be retrieved in the search results.

For example, if you search for issues using the query term 'customize' on the Summary field, JIRA stems this word to its root form 'custom' and will retrieve all issues whose Summary field also contains any word that can be stemmed back to 'custom'. Hence, the following query:

summary ~ "customize"

will retrieve issues whose Summary field contains the following words:

- customized
- customizing
- customs
- customer
- etc.

Please Note:

- Your JIRA administrator can disable word stemming (so that JIRA will find issues based on exact matches with words) by changing the Indexing Language to Other (under Administration > System > General Configuration).
- Word stemming applies to all JIRA fields (as well as text fields).
- When JIRA indexes its fields, any words that are 'stemmed' are stored in JIRA's search index in root form only.

Limitations

Please note that the following limitations apply to JIRA's search:

Whole words only
JIRA cannot search for issues containing parts of words but on whole words only. The exception to this are words which are stemmed.

This limitation can also be overcome using fuzzy searches.

Using the Issue Navigator

JIRA is optimised to let you quickly navigate lists of issues, switch to a detail view from the basic list view and back again, and take action on those items that need your immediate attention.

List View

If you’ve used JIRA for some time, the list view is the view you are used to seeing; namely, the search results from a filter, a basic search, a quick search, or an advanced search. You only see one set of search results at any given time, even if you have multiple browser windows open. And with stable search, your search results remain constant until you choose to refresh them. This provides you with a constant set of search results that you can work from when triaging issues. Since your search results don’t change as you are making changes to the list, you maintain the original list you started with.

Detail View

JIRA is optimized to let you quickly navigate lists of issues, switch to a detail view from the basic list view and back again, and take action on those items that need your immediate attention. The detail view lets you view an issue within the context of the filter it’s a part of. This is great when you are triaging issues and you need a bit more context.

Filters

You can easily save your searches, then display them as needed. In JIRA, we call these filters. Filters are basically bookmarked collections of issues.

Example basic search results in the issue navigator (click to view full size image)

You can also hide the filter panel to gain more screen real estate. Simply click Undock at the top right of the panel. The filter panel collapses as show below:
Docking and undocking the filter panel (click to view full size image)

To redisplay the panel, click the arrow and then select the word Dock that appears. This puts the filter panel back in its place.

On this page:
- List View
- Detail View
- Filters
- Working with issues
- Viewing individual issues

Related topics:
- Advanced Searching
- Basic Searching
- Customizing your Issue Navigator
- Using Filters
- Sharing a Search Result
- Working with Search Result Data

Working with issues

To get started, go to Issues > Search for Issues. From this point, there are many different ways to work with issues and filters, described below.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced</td>
<td>Switches from basic to advanced search, or JIRA Query Language (JQL) search. For details, see Advanced Searching.</td>
</tr>
<tr>
<td>Basic</td>
<td>Switches from advanced to basic search, or User Interface search. For details, see Basic Searching.</td>
</tr>
<tr>
<td>Keyboard shortcuts</td>
<td>Use the 'j' and 'k' keys on your keyboard to move between issues. The row for the currently selected issue is highlighted in blue and any keyboard shortcuts are performed on this issue.</td>
</tr>
</tbody>
</table>
### Sort
Click any field header in the search results to sort by that field in ascending order. Click again to sort in descending order. **Note:**
- You cannot sort by the 'Images' column nor the sub-task aggregate columns (i.e. all columns beginning with "").
- If you sort the search results from an advanced search, an 'ORDER BY' clause will be added to your JQL query to reflect the order of issues in your search results. If the JQL query already has an 'ORDER BY' clause, it will be updated.
- You can choose different fields to display in your Issue Navigator. See Customizing your Issue Navigator.

### Filters
Click a filter name to run the filter. The JIRA system filters – My Open Issues, Reported by Me, Recently Viewed, All Issues – and your favorite filters are displayed in the list.

For details, see Using Filters.

### Share
Click to email the search results to other users or email addresses. From the Share dialog, you can also find a permanent link to the search results.

For details, see Sharing a Search Result.

### Export
Click and select from the dropdown menu to view/export your search results in various formats, e.g. XML, MS Excel, etc. For details, see Working with Search Result Data.

### Tools
This is the actions menu for the search results, not for individual issues.
<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulk Change</td>
<td>Enables operations to be performed on multiple issues at once. These operations are performed on the result set of a search. The following list details the available bulk operations:</td>
</tr>
</tbody>
</table>
|               | • **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.                                             |
|               | • **Edit**   
|               | This operation allows multiple fields in multiple issues to be edited at once.                                                               |
|               | For details, see **Modifyi ng Multiple ('Bulk') Issues**.                                                                                     |
| Configure Columns | You can customize your Issue Navigator by choosing:                                                                                      |
| Set filters column order | Lets you set an associated Column Order with a saved filter. Displays the same screen discussed in **Customizing your Issue Navigator**. |
### Issue actions

This is the actions menu for an issue — located next to an issue in the search results — not for all of the search results.

Click and select from the dropdown menu to action the issue. If the icon is not showing, hover over the issue. For details, see Working with an Issue.

ℹ️ Some options in this menu will only be available if you have the necessary permissions, or if certain features have been enabled by your JIRA administrator.

---

**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).

**Customizing your Issue Navigator**

JIRA lets you change the columns of the table of search results for any search results displayed using the List view (as opposed to the Detail View). Click Columns at top right of the issue table to open the column configuration dialog, shown below.
Column Configuration Dialog

This displays the list of the columns used in the current table of results. Choose the columns you want with checkboxes and click **Done** to finish. Notice that the Filter option is greyed out, this is because the the issue table results are not coming from a filter. See Changing the column configuration for your own filters for an example of using this dialog to set the displayed columns for your own filters.

### Sorting and rearranging columns

- To sort issues, just click on a column header.
- To rearrange the column layout, press and hold the mouse button to enter "column drag mode."

### My Defaults, Filter, and System

If the currently selected button is **My Defaults**, this indicates that the columns you are seeing are from your user account preferences. **Filter** is an available option whenever the issue search results come from a saved filter. If you are a JIRA Admin, you will also see the **System** tab, where you can change the columns for all users who have not set their own defaults.

JIRA administrators can configure the columns that appear in the Issue Navigator for all users that do not have personal column filters defined. When administrators are configuring default 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.

JIRA administrators can also select which views are available in the JIRA system, as views are configurable via plugins.
Changing the column configuration for your own filters

If you are searching using a saved filter and if the filter is owned by you, use the Filter button to customize the columns displayed when users see results from that filter. When sharing a filter with other users, it's sometimes helpful to choose the relevant columns for those results. For example, if your filter searches for issues that are open bugs, you may decide to remove the columns for status and issue type for that filter since they will all be the same. Filters don't always have columns configured, but when they do, those columns will be shown unless the user chooses to use their defaults using the My Defaults button.

For any JIRA filters that you own, you can change the displayed columns as follows.

1. Click on the name of a JIRA filter you own.
2. Click the Columns button at top right of the currently displayed columns. This opens the column configuration dialog.
3. Select or deselect checked items in the list.
4. Click Done when you are finished.

Troubleshooting

If you cannot find a column, please make sure that you haven't run in to any of the following restrictions:

- You can only see columns for issue fields that have not been hidden and that you have permissions to see.
- It is possible to add any of the existing custom fields to the column list, as long as the fields are visible and you have the right permissions.
- Some custom fields, even if selected, do 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.

Using Filters

JIRA's powerful issue search functionality is enhanced by the ability to save searches, called filters in JIRA, for later use. You can do the following with JIRA filters:

- Share and email search results with your colleagues as well as people outside of your organization
- Create lists of favorite filters
- Have search results emailed to you according to your preferred schedule
- View and export the search results in various formats (RSS, Excel, etc)
- Display the search results in a report format
- Display the search results in a dashboard Gadget

Screenshot: Issue filter results in list view (click to view full size image)
To redisplay the panel, click the arrow and then select the word **Dock** that appears. This puts the filter panel back in its place.

**On this page:**
- Running a filter
- Saving a search as a filter
- Managing your existing filters
- Managing other user’s shared filters

**Related topics:**
- Searching for Issues
- Using the Issue Navigator
Running a filter

Run a filter by clicking it. System filters – *My Open Issues, Reported by Me, Recently Viewed, All Issues* – and your favorite filters are shown on the left of the issue navigator. You can also find and run filters using the *Manage Filters* page.

To perform a saved search:

1. Choose **Issues > Search for Issues**.
2. Choose any filter from the list on the left:
   - System filter — *My Open Issues, Reported by Me, Recently Viewed, All Issues*
   - Favorite filters (listed alphabetically)
   - Find filters lets you search for any filter that’s been shared, which you can then subscribe to (adding it to your Favorite Filters).
3. After selecting a filter, the search results are displayed. The search criteria for the filter is also displayed. **If you choose the Recently Viewed system filter, this switches you to advanced search due to the fact that basic search cannot represent the ORDER BY clause in this filter.**
4. You can add, remove or modify the search criteria to refine the search results. You can also save the modified search criteria (if you are updating your own filter), or save a copy of the search criteria as a new filter.

For instructions on adding, removing or modifying filters, see the following sections.

Saving a search as a filter

To save a search as a filter:

1. Define and run your search as described in **Searching for Issues**.
2. Click the **Save as** link above the search results. The **Save Filter** dialog is displayed.

3. Enter a name for the new filter and click **Submit**. Your filter is created.

About saved filters:

- Your new filter is added to your favorite filters.
- Your new filter is shared according to the sharing preference in your user profile. **If you have not specified a personal preference, then the global default for sharing applies (i.e. Private, unless it's been changed by your JIRA Administrator).**
- You need the Create Shared Object global permission to be able to share your filter. If you do not see the Share field, contact your JIRA Administrator to have this permission added to your profile.
- To change how your new filter is shared, refer to the instructions on sharing filters.

Managing your existing 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. You can access this page by selecting **Find filters** or through the **Issues > Manage Filters** menu option.
In this section:

- Searching for a filter
- Updating a filter
- Deleting a filter
- Cloning a Filter
- Adding a filter as a favorite
- Sharing a filter
- Defining a filter-specific column order
- Subscribing to a Filter

Searching for a filter

Filters that you have created or that have been shared by other users can be found using the Manage Filters page. If the filter has been added as a favorite by many users, you also may be able locate it on the Popular tab of the Manage Filters page. This tab lists the top 20 most popular filters.

To search for an existing filter:

1. Choose Issues > Manage Filters.
2. Click the Search tab. The issue filter Search will display.
   Tip: You can navigate to this screen via the Find filters link on the issue navigator.
3. Enter your search criteria and click Search to run the search.

4. Your search results are displayed on the same page. Click the name of any issue filter to run it. You can also sort the search results by clicking the column headers.
Updating a filter

You can update the details – name, description, sharing, favorite – as well as the search criteria of an existing filter.

Please note that you can only update filters that you have created. To edit a filter that was shared with you, either clone (aka copy) the shared filter or ask your JIRA administrator to change the filter’s ownership.

To update an existing filter:

1. Choose **Issues > Manage Filters.**
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, click the cog icon and select **Edit** from the dropdown menu.
4. The **Edit Current Filter** page displays, where you can update the filter details as required. To change the sharing or favorite settings, refer to the relevant instructions below.
If the filter sharing functions shown 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.

5. Click Save to save your changes.

To update the search criteria of one of your existing filters:

1. Choose Issues > Manage Filters.
2. Click the My tab. This tab displays the filters that you’ve created.
3. Click the cog icon and select Edit from the dropdown menu.
4. Update the search criteria as desired. You will see the word Edited displayed next to your filter name.
5. Click Save to overwrite the current filter with the updated search criteria. If you want discard your changes instead, click the arrow next to the save button and select Discard changes.

Deleting a filter

You can delete any filter that you’ve created using this procedure.

To delete an existing filter:

1. Choose Issues > Manage Filters.
2. Click the My tab. This tab displays the filters that you’ve created.
3. Locate the filter you wish to update, click the cog icon and select Delete from the dropdown menu.

Cloning a Filter

You can clone any filter – which is just a way of making a copy that you own – that was either created by you or shared with you.

To clone an existing filter:

1. Choose Issues > Manage Filters.
2. Locate the filter you wish to clone, and click the cog icon and select Edit from the dropdown menu.
3. Update the search criteria as desired. Click the name of the issue filter to run it and display the search results.
4. Click the arrow next to the Save button, and select Save > Save as to create a new filter from the existing filter:

To discard your changes instead, click Save > Discard changes.

Adding a filter as a favorite

Filters that you’ve created or that have been shared by others can be added to your favorite filters. Favorite filters are listed in the menu under Issues > Filters and in the left panel of the issue navigator. You can also add them to your dashboard using the Favorite Filters gadget.

Keep in mind:

- Filters that are already favorites are shown with a star.
- Filters that are not currently your favorites are shown without a star.

There are two ways you can add an existing shared filter as a favorite, described below.

To add a filter as a favorite using the Manage Filters page:

1. Choose Issues > Manage Filters.
2. Locate the filter you wish to add as a favorite. If you created the filter, it is listed under the My tab,
otherwise use the Search tab to find it.

3. Click the star icon next to the filter name to add it to your favorites.

**To add the filter that you are currently viewing as a favorite:**

- Click the star icon next to the filter name.

**Sharing a filter**

Filters that you have created can be shared with other users via user groups, projects and project roles. They can also be shared globally. Any filter that is shared is visible to users who have the JIRA Administrators global permission via the Shared Filters feature. See Managing Other User’s Shared Filters below for details.

**To share an existing filter using the Details button:**

1. Choose Issues > Manage Filters.
2. Click the My tab. This tab displays all the filters that have been created by you.
3. Locate the filter you wish to share and click on it.
4. Click the Details link to the right of the filter’s name, as shown here:

![Filter Details](image)

5. Click Edit permissions to open the Edit Current Filter screen:

![Edit Current Filter](image)

6. Update the Add Shares field by selecting the group, project or project role that you want to share the filter with, and clicking 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.*

   If the filter sharing functions shown 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.
7. Click **Save** to save your changes.

**To share an existing filter using Manage Filters:**

1. Choose **Issues > Manage Filters.**
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, click the cog icon \(\text{⚙} \downarrow\) and select **Edit** from the dropdown menu.
4. Follow **steps 6 and 7** in the previous procedure.

**Defining a filter-specific column order**

You can add a defined column order to a saved filter, which displays the filter results according to the saved column order. 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, select HTML for the *Outgoing email format* in your **User Profile**. If you receive text emails from JIRA, you won't be able to see your configured column order.

**To add a column layout to a saved filter:**

1. Choose **Issues > Manage Filters.**
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, click the cog icon \(\text{⚙} \downarrow\) and select **Columns** from the dropdown menu.
4. Configure the column order as desired. You can configure the column order the same way you would configure your personal Issue Navigator column order.

**To remove a filter’s saved column layout:**

1. Choose **Issues > Manage Filters.**
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, click the cog icon \(\text{⚙} \downarrow\) and select **Columns** from the dropdown menu.
4. Click the **Remove Filter’s Column Order** link near the top of the page. The default column order will be restored.
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 use your default column order for a search:

1. Run the issue filter. The search will be displayed in the issue navigator.
2. Click the cog icon above the search results and select Use your default Column Order.

To go back to using the filter's own column order, follow the instructions above but 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. For instructions, see Managing Shared Filters in the JIRA Administrator's Guide.

Working with Search Result Data

- Displaying Search Results as a Chart
- Displaying Search Results in XML
- Exporting Search Results to Microsoft Excel
- Exporting Search Results to Microsoft Word
- Receiving Search Results as an RSS Feed
- Receiving Search Results via Email
- Sharing a Search Result

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.
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 issue has been in a status.
- 'Time to First Response' — displays the average number of times an issue has been in a status.

¹ This particular chart will only be available if your JIRA administrator has installed the JIRA Charting.
Displaying a chart on your dashboard

To view your search results as a chart:

2. Refine your search, as described in Searching for Issues, then choose the Export menu.
3. Choose Charts from the dropdown menu.
4. Your search results will be displayed as the default chart.

5. Click Save to Dashboard.
6. Select a project dashboard and name your gadget in the Filter Name field.
7. Click Save. Your new dashboard gadget is displayed.

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:

2. Refine your search, as described in Searching for Issues, then choose the Export menu.
3. Choose XML from the dropdown menu.
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:

<table>
<thead>
<tr>
<th>Value</th>
<th>Sample XML output</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td><code>&lt;title&gt;[TEST-4] This is a test&lt;/title&gt;</code></td>
<td></td>
</tr>
<tr>
<td>link</td>
<td><code>&lt;link&gt;https://extranet.atlassian.com:443/jira/browse/TEST-4&lt;/link&gt;</code></td>
<td>This is a 'permalink' to the issue. For links between issues, see <code>* is suelinks *</code> (below).</td>
</tr>
<tr>
<td>project (or <em>pid</em>)</td>
<td><code>&lt;project id=&quot;10330&quot; key=&quot;TST&quot;&gt;Test&lt;/project&gt;</code></td>
<td></td>
</tr>
<tr>
<td>description</td>
<td><code>&lt;description&gt;This is a detailed description of the issue.&lt;/description&gt;</code></td>
<td></td>
</tr>
<tr>
<td>environment</td>
<td><code>&lt;environment&gt;Sydney network&lt;/environment&gt;</code></td>
<td></td>
</tr>
<tr>
<td>key</td>
<td><code>&lt;key id=&quot;22574&quot;&gt;TEST-4&lt;/key&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>summary</strong></td>
<td><code>&lt;summary&gt;This is a test&lt;/summary&gt;</code></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>type (or issuetype)</strong></td>
<td><code>&lt;type id=&quot;3&quot; iconUrl=&quot;https://extranet.atlassian.com:443/jira/images/icons/task.gif&quot;&gt;Task&lt;/type&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>parent</strong></td>
<td><code>&lt;parent id=&quot;22620&quot;&gt;TEST-5&lt;/parent&gt;</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Only relevant if the issue is a subtask.</td>
<td></td>
</tr>
<tr>
<td><strong>priority</strong></td>
<td><code>&lt;priority id=&quot;4&quot; iconUrl=&quot;https://extranet.atlassian.com:443/jira/images/icons/priority_minor.gif&quot;&gt;Minor&lt;/priority&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>status</strong></td>
<td><code>&lt;status id=&quot;5&quot; iconUrl=&quot;https://extranet.atlassian.com:443/jira/images/icons/status_resolved.gif&quot;&gt;Resolved&lt;/status&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>resolution</strong></td>
<td><code>&lt;resolution id=&quot;1&quot;&gt;Fixed&lt;/resolution&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>labels</strong></td>
<td><code>&lt;labels&gt;&lt;label&gt;focus&lt;/label&gt;&lt;label&gt;focus&lt;/label&gt;&lt;labels&gt;</code></td>
<td></td>
</tr>
<tr>
<td><strong>assignee</strong></td>
<td><code>&lt;assignee username=&quot;jsmith&quot;&gt;John Smith&lt;/assignee&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>reporter</td>
<td>&lt;assignee username=&quot;jsmith&quot;&gt;John Smith&lt;/assignee&gt;</td>
<td></td>
</tr>
<tr>
<td>security</td>
<td>&lt;security id=&quot;10021&quot;&gt;Private&lt;/security&gt;</td>
<td>Only relevant if a security level has been applied to the issue.</td>
</tr>
<tr>
<td>created</td>
<td>&lt;created&gt;Mon, 1 Sep 2008 17:30:03 -0500 (CDT)&lt;/created&gt;</td>
<td></td>
</tr>
<tr>
<td>updated</td>
<td>&lt;updated&gt;Mon, 1 Sep 2008 17:30:03 -0500 (CDT)&lt;/updated&gt;</td>
<td></td>
</tr>
<tr>
<td>resolved (or resolutiondate)</td>
<td>&lt;resolved&gt;Mon, 1 Sep 2008 17:30:03 -0500 (CDT)&lt;/resolved&gt;</td>
<td></td>
</tr>
<tr>
<td>due (or duedate)</td>
<td>&lt;due&gt;Mon, 1 Sep 2008 17:30:03 -0500 (CDT)&lt;/due&gt;</td>
<td></td>
</tr>
<tr>
<td>version (or versions)</td>
<td>&lt;version&gt;2.4.7&lt;/version&gt;</td>
<td></td>
</tr>
<tr>
<td>fixfor (or fixVersions)</td>
<td>&lt;fixVersion&gt;2.6&lt;/fixVersion&gt;</td>
<td></td>
</tr>
<tr>
<td>component (or components)</td>
<td>&lt;component&gt;Documentation&lt;/component&gt;</td>
<td></td>
</tr>
<tr>
<td>votes</td>
<td>&lt;votes&gt;1&lt;/votes&gt;</td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
</tbody>
</table>
| comments | `<comments>
  <comment
    id="39270"
    author="jsmith"
    created="Tue, 24 Feb 2009 16:45:02 -0600 (CST)">
    this looks familiar
  </comment>
  <comment
    id="39273"
    author="jbrown"
    created="Tue, 24 Feb 2009 16:48:16 -0600 (CST)">
    to me too
  </comment>
</comments>`<br>Only available if your administrator has enabled comments. |
| attachments | `<attachments>
  <attachment
    id="30318"
    name="Issue Navigator - Atlassian JIRA-2.png"
    size="16161"
    author="yoz"
  </attachment>
  <attachment
    id="30323"
    name="Windows XP (with Firefox 3.0).jpg"
    size="5802"
    author="vbhararara"
    created="Tue, 10 Feb 2009 00:30:11 -0600 (CST)"
  </attachment>
</attachments>`<br>Only available if your administrator has enabled attachments. |
<p>| timeoriginalestimate | <code>&lt;timeoriginalestimate seconds=&quot;600&quot;&gt;10 minutes&lt;/timeoriginalestimate&gt;</code>&lt;br&gt;Only available if your administrator has enabled 'time-tracking'. |
| timeestimate | <code>&lt;timeestimate seconds=&quot;300&quot;&gt;5 minutes&lt;/timeestimate&gt;</code>&lt;br&gt;Only available if your administrator has enabled 'time-tracking'. |</p>
<table>
<thead>
<tr>
<th>Timespent</th>
<th><code>&lt;timespent seconds=&quot;300&quot;&gt;5 minutes&lt;/timespent&gt;</code></th>
<th>Only available if your administrator has enabled 'time-tracking'.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregate Time Original Estimate</td>
<td><code>&lt;aggregatetimeoriginalestimate seconds=&quot;36000&quot;&gt;10 hours&lt;/aggregatetimeoriginalestimate&gt;</code></td>
<td>(ie. aggregate time for the issue plus all of its sub-tasks.) Only available if your administrator has enabled 'time-tracking'.</td>
</tr>
<tr>
<td>Aggregate Time Estimate</td>
<td><code>&lt;aggregatetimeestimate seconds=&quot;18000&quot;&gt;5 hours&lt;/aggregatetimeestimate&gt;</code></td>
<td>(ie. aggregate time for the issue plus all of its sub-tasks.) Only available if your administrator has enabled 'time-tracking'.</td>
</tr>
<tr>
<td>Aggregate Time Spent</td>
<td><code>&lt;aggregatetimespent seconds=&quot;18000&quot;&gt;5 hours&lt;/aggregatetimespent&gt;</code></td>
<td>(ie. aggregate time for the issue plus all of its sub-tasks.) Only available if your administrator has enabled 'time-tracking'.</td>
</tr>
<tr>
<td>Time Tracking</td>
<td><code>&lt;timeoriginalestimate seconds=&quot;600&quot;&gt;10 minutes&lt;/timeoriginalestimate&gt; &lt;timeestimate seconds=&quot;300&quot;&gt;5 minutes&lt;/timeestimate&gt; &lt;timespent seconds=&quot;300&quot;&gt;5 minutes&lt;/timespent&gt; &lt;aggregatetimeoriginalestimate seconds=&quot;36000&quot;&gt;10 hours&lt;/aggregatetimeoriginalestimate&gt; &lt;aggregatetimeestimate seconds=&quot;18000&quot;&gt;5 hours&lt;/aggregatetimeestimate&gt; &lt;aggregatetimespent seconds=&quot;18000&quot;&gt;5 hours&lt;/aggregatetimespent&gt;</code></td>
<td>This is a convenient shorthand way of specifying all of the above six time-tracking fields. (Only available if your administrator has enabled 'time-tracking'.)</td>
</tr>
</tbody>
</table>
### issuelinks

```xml
<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)

```xml
<subtasks>
    <subtask id="22623">TEST-8</subtask>
</subtasks>
```

### customfield_xxxx

```xml
<customfields>
    <customfield id="customfield_10112" key="com.atlassian.jira.plugin.system.customfieldtypes:select">
        <customfieldname>Department</customfieldname>
        <customfieldvalues>
            <customfieldvalue>Administration</customfieldvalue>
        </customfieldvalues>
    </customfield>
</customfields>
```

(Where "xxxxx" is the id of a given custom field. E.g. this output is the result of specifying `&field=customfield_10112`.)
### 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.
Viewing a Chart
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 customized reports, graphs and charts.

**Related topics:**
- Exporting Search Results to Microsoft Word

### Exporting to Microsoft Excel

**Before you begin:**
- 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.

**To export search results to Microsoft Excel:**

1. Choose **Issues > Search for Issues**.
2. Refine your search, as described in **Searching for Issues**, then choose the **Export** menu.
3. Choose one of the following from the dropdown menu:
   - **'Excel (All fields)'** — this will create a spreadsheet column for every issue field (excluding comments).
     - **Note:** This will only show the custom fields that are available for all of the issues in the search results. For example, if a field is only available for one project and multiple projects are in the search results then that field will not appear in the Excel document. The same goes for fields that are only available for certain issue types.
   - **'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:
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 customized format.

**Related topics:**
- Exporting Search Results to Microsoft Excel

**Exporting to Microsoft Word**

**Before you begin:**

- 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.

**To export search results to Microsoft Word:**

1. Choose **Issues > Search for Issues.**
2. Refine your search, as described in **Searching for Issues**, then choose the **Export** menu.

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
3. Choose **Word** from the dropdown menu.
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:

![Sample Exported File](image)

**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](http://jira.atlassian.com).
Subscribing to an RSS Feed

Before you begin:

- Tip: If you only want to receive current comments in an RSS feed, use the Date Updated field when doing a search. For example, to only receive comments created in the last week, add the Date Update field and set it to updated within the last 1 week.
- The tempMax parameter in the RSS URL can be used to control the maximum number of issues returned in your RSS feed.
- If you are getting an empty RSS feed, make sure you have logged in to JIRA — see Accessing protected data (below).

To subscribe to an RSS feed:

2. Refine your search, as described in Searching for Issues, then choose the Export menu.
3. Choose one of the following from the dropdown menu:
   - 'RSS (Issues)' — this will create an RSS feed that contains just issue data.
   - 'RSS (Comments)' — this will create an RSS feed that contains comments.
4. Copy the URL that is currently displayed on your Issue Navigator screen.
5. Paste the URL into your RSS reader.

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.

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. The filter results will only send the first 200 results of a filter.
Subscribing to a Filter

1. Choose **Issues > Manage Filters.**
2. A list of available filters is displayed:

3. Locate the filter you are interested in and click on its **Subscribe** link. The **Filter Subscription** form is 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 dropdown list.  
   **Note:** You cannot select a group unless your JIRA administrator has granted you the 'Manage Group Filter Subscriptions' global permission.
5. 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.

6. Click **Subscribe**.
7. You will now be shown a subscription summary page. If you wish, 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:

<table>
<thead>
<tr>
<th>Field</th>
<th>Allowed values</th>
<th>Allowed special characters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second</td>
<td>0-59</td>
<td>, - */</td>
</tr>
<tr>
<td>Minute</td>
<td>0-59</td>
<td>, - */</td>
</tr>
<tr>
<td>Hour</td>
<td>0-23</td>
<td>, - */</td>
</tr>
<tr>
<td>Day-of-month</td>
<td>1-31</td>
<td>, - */ ? L W C</td>
</tr>
<tr>
<td>Month</td>
<td>1-12 or JAN-DEC</td>
<td>, - */</td>
</tr>
<tr>
<td>Day-of-week</td>
<td>1-7 or SUN-SAT</td>
<td>, - */ ? L C #</td>
</tr>
<tr>
<td>Year (optional)</td>
<td>1970-2099</td>
<td>, - */</td>
</tr>
</tbody>
</table>

The special characters operate as follows:

<table>
<thead>
<tr>
<th>Special character</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>,</td>
<td>Specifies a list of values. For example, in the <strong>Day-of-week</strong> field, ‘MON,WED,FRI’ means ‘every Monday, Wednesday, and Friday’.</td>
</tr>
<tr>
<td>-</td>
<td>Specifies a range of values. For example, in the <strong>Day-of-week</strong> field, ‘MON-FRI’ means ‘every Monday, Tuesday, Wednesday, Thursday and Friday’.</td>
</tr>
<tr>
<td>*</td>
<td>Specifies all possible values. For example, in the <strong>Hour</strong> field, ‘*’ means ‘every hour of the day’.</td>
</tr>
<tr>
<td>/</td>
<td>Specifies increments to the given value. For example, in the <strong>Minute</strong> field, ‘0/15’ means ‘every 15 minutes during the hour, starting at minute zero’.</td>
</tr>
<tr>
<td>?</td>
<td>Specifies no particular value. This is useful when you need to specify a value for one of the two fields <strong>Day-of-month</strong> or <strong>Day-of-week</strong>, but not the other.</td>
</tr>
<tr>
<td>L</td>
<td>Specifies the last possible value; this has different meanings depending on context. In the <strong>Day-of-week</strong> 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 <strong>Day-of-month</strong> field, ‘L’ on its own means ‘the last day of the month’, or ‘LW’ means ‘the last weekday of the month’.</td>
</tr>
</tbody>
</table>
Specifications:

- **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:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 15 8 ? * *</td>
<td>Every day at 8.15 pm.</td>
</tr>
<tr>
<td>0 15 8 * * ?</td>
<td>Every day at 8.15 am.</td>
</tr>
<tr>
<td>0 * 14 * * ?</td>
<td>Every minute starting at 2.00 pm and ending at 2:59 pm, every day.</td>
</tr>
<tr>
<td>0 0/5 14 * * ?</td>
<td>Every 5 minutes starting at 2.00 pm and ending at 2:55 pm, every day.</td>
</tr>
<tr>
<td>0 0/5 14,18 * * ?</td>
<td>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.</td>
</tr>
<tr>
<td>0 0-5 14 * * ?</td>
<td>Every minute starting at 2.00 pm and ending at 2:05 pm, every day.</td>
</tr>
<tr>
<td>0 0/10 * * * * *</td>
<td>Every 10 minutes, forever.</td>
</tr>
<tr>
<td>0 10,44 14 ? 3 WED</td>
<td>2:10 pm and 2:44 pm every Wednesday in the month of March.</td>
</tr>
<tr>
<td>0 15 8 ? * MON-FRI</td>
<td>8:15 am every Monday, Tuesday, Wednesday, Thursday and Friday.</td>
</tr>
<tr>
<td>0 15 8 15 * ?</td>
<td>8:15 am on the 15th day of every month.</td>
</tr>
<tr>
<td>0 15 8 L * ?</td>
<td>8:15 am on the last day of every month.</td>
</tr>
<tr>
<td>0 15 8 LW * ?</td>
<td>8:15 am on the last weekday of every month.</td>
</tr>
<tr>
<td>0 15 8 ? * 6L</td>
<td>8:15 am on the last Friday of every month.</td>
</tr>
<tr>
<td>0 15 8 ? * 6#2</td>
<td>8:15 am on the second Friday of every month.</td>
</tr>
</tbody>
</table>

**Subscription Context**

Each time that a subscription is activated, the saved filter for the subscription is used to find the relevant issues. This search is executed for each user in the group that will receive the subscription email. This means that any filter that uses JQL function such as `currentUser()` will be evaluated with a different user each time. This also means that if one user doesn't have permission to see some of the issues returned by the search then the list of issues that they receive in email will be different from other users’ email. This has two other consequences:

- If you share a subscription with a group with many members it can take a long time to generate the emails to be sent. This can make JIRA very slow for minutes. This is a good reason be careful with allowing the Manage Group Filter Subscriptions permission.
• If a group contains a user whose email address is actually a distribution list (i.e., a group email alias), then everyone on that distribution list will receive the same email, since JIRA doesn't know who the users on the list are.

**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.

**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.

**Related topics:**

- Searching for Issues

To share a search result with one or more JIRA users or any email addresses:

2. Refine your search, as described in Searching for Issues, then choose the share icon.
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. You can access these reports directly from your project browser screen:

![Project Overview Summary Screen](image)

Reports might also be located in the project Overview Summary screen below the issues graph.

You can access these reports directly from your project Overview Summary screen below the issues graph:

The following reports are included with JIRA:

- **Average Age Report** — Shows the average age (in days) of unresolved issues.
- **Created vs Resolved Issues Report** — Shows the number of issues created vs number of issues resolved over a given period of time.
- **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.
- **Resolution Time Report** — Shows the average time taken to resolve issues.
- **Recently Created Issues Report** — Shows the rate at which issues are being created.
- **Single Level Group By Report** — Shows the search results from an issue filter, grouped by a field of your choice.
- **Time Since Issues Report** — Shows the number of issues for which your chosen date field (e.g. ‘Created’)

Creative Commons Attribution 2.5 Australia License.
was set on a given date.

- **Time Tracking Report** — Shows time tracking information on issues for a particular version of a project.
- **User Workload Report** * — Shows how much work a user has been allocated, and how long it should take.
- **Version Time Tracking Report** * — Shows progress towards completing a given version, based on issues' work logs and time estimates.
- **Version Workload Report** * — Shows how much outstanding work there is (per user and per issue) before a given version is complete.
- **Workload Pie Chart Report** * — Shows the relative workload for assignees of all issues in a particular project or issue filter.

* 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 Marketplace.
- JIRA administrators can also create new reports with the plugin API — see our Plugin Tutorial – Creating a JIRA Report. If you don't want 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 the report:

1. Navigate to the desired project.
2. Choose Summary (tab) > Reports section > Workload Pie Chart Report.
3. Choose the project or issue filter for which you wish to generate a Workload Pie Chart report.
4. In the Statistic drop-down list, select the field on which the pie chart will be based (this will usually be Assignee).
5. 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:

![Internet Options menu](image)

2. The 'Internet Options' window will display. Click the 'Settings' button in the 'Temporary Internet files' (i.e. cache) section:

![Settings window](image)

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 number of issues and the total workload remaining for this user.

Generating a User Workload report

To generate the report:

1. Navigate to the desired project.
3. Select or type the name of the user for whom you wish to generate a User Workload report.
4. 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.
5. Click Next to generate the report.

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.
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 the report:

1. Navigate to the desired project.
3. 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.
4. 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 spe
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 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*

Or, if sub-tasks are enabled, the report will look something like this:
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:

```
Progress: 40%
```

- 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.

```
Progress: 40%
Accuracy: 0%
```

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.

```
Progress: 42%
Accuracy: -4%
```

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.

```
Progress: 47%
Accuracy: 8%
```
Generating a Time Tracking report

To generate a Time Tracking Report:

1. Navigate to the desired project.
2. Choose Summary (tab) > Reports section > Time Tracking Report.

3. 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.
4. 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.
5. 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.
6. 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. Navigate to the desired project.
2. Choose **Summary** (tab) > **Reports** section > **Single Level Group By Report**.
3. Select the desired issue filter.
4. 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?

The report generated will look something like this:

**Screenshot: 'Created vs Resolved Issues' report**

- **Chart**
  - This chart shows the number of issues created vs the number of issues resolved in the last 30 days.

- **Data Table**
  - 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. Navigate to the desired project.
2. Choose **Summary** (tab) > **Reports** section > **Created vs Resolved Issues Report**.
3. Click Select Filter or Project.
4. 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.
5. In the Period field, select the timeframe on which the report will be based:
6. In the Days Previously field, enter the number of days' worth of data (counting backwards from today) to be included in the report.
7. 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.
8. In the Display the Trend of Unresolved field, choose either:
   - Yes to show the number of unresolved issues over time in a subplot; or
   - No otherwise.
9. 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:

Generating a Resolution Time report

1. Navigate to the desired project.
3. Choose the desired filter or project.
4. In the Period field, choose the timeframe on which the report will be based.
5. In the Days Previously field, enter the number of days 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
Generating a Pie Chart report

1. Navigate to the desired project.
2. Choose **Summary** (tab) > **Reports** section > **Pie Chart Report**.

3. Choose the desired filter/project.
4. 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

<table>
<thead>
<tr>
<th>Period</th>
<th>Issues Unresolved</th>
<th>Total Age</th>
<th>Avg. Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-January-2009</td>
<td>17</td>
<td>2082</td>
<td>157</td>
</tr>
<tr>
<td>8-January-2009</td>
<td>17</td>
<td>2099</td>
<td>158</td>
</tr>
<tr>
<td>9-January-2009</td>
<td>15</td>
<td>2563</td>
<td>170</td>
</tr>
<tr>
<td>10-January-2009</td>
<td>15</td>
<td>2568</td>
<td>171</td>
</tr>
<tr>
<td>11-January-2009</td>
<td>15</td>
<td>2583</td>
<td>172</td>
</tr>
<tr>
<td>12-January-2009</td>
<td>15</td>
<td>2598</td>
<td>173</td>
</tr>
<tr>
<td>13-January-2009</td>
<td>14</td>
<td>2545</td>
<td>181</td>
</tr>
<tr>
<td>14-January-2009</td>
<td>15</td>
<td>2569</td>
<td>179</td>
</tr>
</tbody>
</table>

Generating an 'Average Age' report

1. Navigate to the desired project.
2. Choose Summary (tab) > Reports section > Average Age Report.
3. Choose the desired filter/project.
4. In the Period field, select the timeframe on which the report will be based.
5. 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:

   ![Internet Options window]

   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 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 (i.e. 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
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. Navigate to the desired project.
2. Choose Summary (tab) > Reports section > Recently Created Issues Report.
3. Choose the desired filter/project.
4. In the **Period** field, select the timeframe on which the report will be based:
5. In the **Days Previously** field, enter the number of days 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:

![Internet Options menu](image)

2. The 'Internet Options' window will display. Click the 'Settings' button in the 'Temporary Internet files' (i.e. cache) section:

![Internet Options window](image)

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.

![Internet Explorer cache settings](image)

### 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 (i.e. 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. Navigate to the desired project.

3. Choose the desired project/filter.
4. In the **Date Field** field, select the desired date. Note, only available if time tracking has been **enabled** by your JIRA administrator.

5. In the **Period** field, select the timeframe on which the report will be based.

6. In the **Days Previously** field, enter the number of days of data (counting backwards from today) to be included in the report.

7. 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:

   ![Internet Options menu](image)

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 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, unreleased versions and a chart showing issue activity. Reports are also located here.
  - 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.
• **Reports** — Shows reports on statistics for particular people, projects, versions, or other fields within issues.
• **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. Your administrator must also integrate Bamboo with JIRA ()

** Only available if your organisation uses Atlassian FishEye. Your administrator must also integrate FishEye with JIRA ()

**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.

Browsing a project's summary

**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 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.

**Screenshot: ‘Summary’ page for a Project**
If you are concerned about screen real estate, you can display the Project Summary page as a single column. Simply resize your browser window and content automatically reformats into a single column display.

<table>
<thead>
<tr>
<th>Function</th>
<th>Instructions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reports</td>
<td>Click the relevant report in the Reports section.</td>
<td>See Generating Reports for more information.</td>
</tr>
<tr>
<td>Versions</td>
<td>Click the relevant version in the Versions section.</td>
<td>See Browsing a Project's Versions for more information.</td>
</tr>
</tbody>
</table>
Activity Stream

- 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, unless you are using Atlassian OnDemand.
- Activity from remote applications. Your administrator will need to set this up via the REST API or the provider plugin API 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 Marketplace for more information.

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 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
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 provides an overview of progress made towards releasing a version; therefore, the versions appear in the opposite order of the way they appear on the Versions screen for the project.

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 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 Road Map in the left column. The road map for your project is displayed (see screenshot below).

   • Click the issues link next to any version to expand the list of issues related to that version.
   • Click the release notes link to see the tasks, bugs, and other information for that version.

   Screenshot: Viewing a project's road map

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

If your administrator has hidden the 'Fix For Version' field, the Road Map report is not available.
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 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 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 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 Versions on the left of the page. A list of versions for your project is displayed (see screenshot below).
   - If specified, the Start Date is used by the Version Report. This gives you a more accurate report in cases where you might plan a version many weeks (or even months) in advance, but not actually commence work until closer to the release date.
   - Click the link for a version to browse that version.

Screenshot: ‘Versions’ page for a Project
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
- Version Report

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 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 Versions on the left of the page. Click the name of the version you wish to view.
4. Click Summary to display the summary for your version (see screenshot below), which contains recently updated issues related to the version.
   - Click the Release Notes link to view the release notes for the version (if released).
   - Click View Issues to see the full list of issues.

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 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 Versions on the left to display the list of versions. Click the name of the version you wish to view.

4. Click Issues on the left of the page. The issues summary for your version is displayed (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
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 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 Versions on the left to display the list of versions. Click the name of the version you wish to view.

4. Click Popular Issues to display the unresolved popular issues for your version. To see resolved popular issues, click Resolved Issues instead.

**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 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:

---

Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](http://creativecommons.org/licenses/by/2.5/au/).
• 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 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 Versions on the left to display the list of versions. Click the name of the version you wish to view.
4. Click Builds to view the following information:
   - Builds related to the Project (default)
   - Status of Build Plans related to the Project (click Latest plan status at the top of the Builds page)

   **Tip:** You will need the 'View Development Tools' permission in the appropriate projects, if you want to view the 'Builds' tab.

**Viewing the Builds related to the Project Version**

By default, Builds displays the related builds, ordered by build date in descending order.

**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 📰. 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 Latest plan status. The build plans listed 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.

**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 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 Components 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.
Viewing a project 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 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.

Related Topics

- 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

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

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.

A live version of this example can be seen online.
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 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**
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.

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 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**

If your administrator has hidden the 'Fix For Version' field, the Road Map report will not be available.
JIRA 6.1 Documentation

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 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

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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 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
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 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. The bigger the text, the more popular the label.

**Screenshot: 'Labels' view for a Project**
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 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 Builds on the left to view the following information:
   - Builds related to the Project (default)
   - Status of Build Plans related to the Project (click Latest plan status at the top of the Builds page)

You will need the 'View Development Tools' permission in the appropriate projects, if you want to view the 'Builds' tab.

Builds related to the Project

By default, Builds displays the related builds, ordered by build date in descending order.

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. Build plans are sorted by the plan name.

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.

Your JIRA administrator must have configured the FishEye plugin on your JIRA server, if you want to view...
the ‘Changeset’ report. You will also need the ‘View Development Tools’ 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 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 Source 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**

Click Statistics to view the Activity Statistics on LOC, Files or Commits for the project.

**Screenshot: Viewing the activity statistics for a project**

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.

⚠️ Your JIRA administrator must have configured the FishEye plugin on your JIRA server, if you want to view the 'Reviews' report. You will also need the 'View Development Tools' 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 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 Reviews on the left of the page. The recent changesets for your project will display. By default, you will see a listing of the most recent changesets 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 JIRA Agile plugin.

⚠️ Your JIRA administrator must have configured the JIRA Agile add-on appropriately, if you want to view the Agile report.

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 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 Agile on the left of the page. The backlog for your project will display.

ℹ️ On this page, you can:

• Use the Version dropdown to display the backlog for a different project version.
• Use the Context dropdown to select a different JIRA Agile context.
• Select Info 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 JIRA Agile chart for your selected project version and context.

Velocity Charts are also known more generically as Value Charts.

Customizing 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 Dashboards at the
top of your screen.

The information boxes on the dashboard are called Gadgets:

![Image of Gadgets]

You can easily customize 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 favorites pages, as described in Managing Multiple Dashboard Pages. Each page can be configured independently, as per the instructions below.

### Available Gadgets

<table>
<thead>
<tr>
<th>Gadget</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Stream Gadget</td>
<td>The Activity Stream gadget displays a summary of your recent activity.</td>
</tr>
<tr>
<td>Administration Gadget</td>
<td>The Administration (Guide for JIRA Administrators) gadget displays checklist of common administration tasks and links to administrative functions and documentation.</td>
</tr>
<tr>
<td>Assigned To Me Gadget</td>
<td>The Assigned To Me gadget displays all open issues in all projects assigned to the current user viewing the dashboard.</td>
</tr>
<tr>
<td>Average Age Gadget</td>
<td>The Average Age gadget displays a bar chart showing the average number of days that issues have been unresolved.</td>
</tr>
<tr>
<td>Bamboo Charts Gadget *</td>
<td>The Bamboo Charts gadget displays various charts and plan statistics from a particular Bamboo server.</td>
</tr>
<tr>
<td>Bamboo Plan Summary Chart Gadget *</td>
<td>The Bamboo Plan Summary gadget displays a graphical summary of a build plan.</td>
</tr>
<tr>
<td>Bamboo Plans Gadget *</td>
<td>The Bamboo Plans gadget displays a list of all plans on a Bamboo server, and each plan's current status.</td>
</tr>
<tr>
<td>Gadget Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Bugzilla ID Search Gadget</td>
<td>The <strong>Bugzilla ID Search</strong> gadget allows the user to search all JIRA issues for references to Bugzilla IDs.</td>
</tr>
<tr>
<td>Calendar Gadget *</td>
<td>The <strong>Issue Calendar</strong> 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.</td>
</tr>
<tr>
<td>Clover Coverage Gadget *</td>
<td>The <strong>Clover Coverage</strong> gadget displays the Clover coverage of plans from a particular Bamboo server.</td>
</tr>
<tr>
<td>Created vs Resolved Gadget</td>
<td>The <strong>Created vs Resolved</strong> gadget displays a difference chart showing the issues created vs resolved over a given period.</td>
</tr>
<tr>
<td>Crucible Charts Gadget *</td>
<td>The <strong>Crucible Charts</strong> gadget displays various charts showing statistical summaries of code reviews.</td>
</tr>
<tr>
<td>Favorite Filters Gadget</td>
<td>The <strong>Favorite Filters</strong> gadget displays a list of all the issue filters that have currently been added by you as a favorite filter.</td>
</tr>
<tr>
<td>Filter Results Gadget</td>
<td>The <strong>Filter Results</strong> gadget displays the results of a specified issue filter.</td>
</tr>
<tr>
<td>FishEye Charts Gadget *</td>
<td>The <strong>FishEye Charts</strong> gadget displays two charts showing showing statistics about a given sourcecode repository.</td>
</tr>
<tr>
<td>FishEye Recent Changesets Gadget *</td>
<td>The <strong>FishEye Recent Changesets</strong> gadget displays a number of recent changesets from a FishEye repository.</td>
</tr>
<tr>
<td>In Progress Gadget</td>
<td>The <strong>In Progress</strong> gadget displays all issues that are currently in progress and assigned to the current user viewing the dashboard.</td>
</tr>
<tr>
<td>Introduction Gadget</td>
<td>The <strong>Introduction</strong> gadget displays a configurable introduction message on the dashboard.</td>
</tr>
<tr>
<td>Issue Statistics Gadget</td>
<td>The <strong>Issue Statistics</strong> gadget displays the collection of issues returned from a specified filter, broken down by a specified field.</td>
</tr>
<tr>
<td>JIRA: News Gadget</td>
<td>The <strong>JIRA:News</strong> gadget displays recent Atlassian news about JIRA.</td>
</tr>
<tr>
<td>Pie Chart Gadget</td>
<td>The <strong>Pie Chart</strong> 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).</td>
</tr>
<tr>
<td>Projects Gadget</td>
<td>The <strong>Projects</strong> gadget provides information and various filters related to a specified project(s).</td>
</tr>
<tr>
<td>Quick Links Gadget</td>
<td>The <strong>Quick Links</strong> gadget displays a number of useful links to issues associated with the current user.</td>
</tr>
<tr>
<td>Recently Created Issues Gadget</td>
<td>The <strong>Recently Created Issues</strong> 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.</td>
</tr>
<tr>
<td>Resolution Time Gadget</td>
<td>The <strong>Resolution Time</strong> gadget displays a bar chart showing the average resolution time (in days) of resolved issues.</td>
</tr>
</tbody>
</table>
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 customize 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 customize 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:

<table>
<thead>
<tr>
<th>Gadget Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Road Map Gadget</td>
<td>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.</td>
</tr>
<tr>
<td>Text Gadget</td>
<td>The Text gadget displays a configurable HTML text on the dashboard.</td>
</tr>
<tr>
<td>Time Since Issues Gadget</td>
<td>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.</td>
</tr>
<tr>
<td>Two Dimensional Filter Statistics Gadget</td>
<td>The Two Dimensional Filter Statistics gadget displays statistical data based on a specified filter in a configurable table format.</td>
</tr>
<tr>
<td>Voted Gadget</td>
<td>The Voted Issues gadget shows issues for which you have voted.</td>
</tr>
<tr>
<td>Watched Gadget</td>
<td>The Watched Issues gadget shows issues which you are watching.</td>
</tr>
</tbody>
</table>

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.
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 organize related information by context. You can also share your dashboard pages with other users, as well as adding dashboards shared by other users as favorites.

You can view a dashboard page by simply clicking its name.

On this page:

- Managing your Dashboard
- Creating new dashboard pages
- Displaying a dashboard page on your dashboard ('Favorite 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 favorite.
   - 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 reorder 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 'favorite' 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 favorite after it has been created. Read more about adding an existing dashboard page as a favorite.

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.

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 favorite, it will also appear under the 'Favorite' tab and will be displayed as a tab on your JIRA dashboard.

6. You can now customize your new dashboard page, and add gadgets to it, as described in Customizing the Dashboard.

Displaying a dashboard page on your dashboard ('Favorite Dashboards')

Dashboard pages that you have created, or that have been shared by other people, can be added as a 'favorite'. 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 'favorite' 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 search for dashboards shared by other users via the ‘Search’ tab.

- Your favorite dashboards are shown with a gray star.
- Dashboards that are not currently your favorites are shown with the outline of a star.

3. Click the gray star icon next to the name of the desired dashboard page to add it as a favorite.

Please note, if you have added another user’s shared dashboard as a favorite 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 ‘favorites’ 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 favorites, 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 favorite to stop the error message from showing. You may need to search for the ‘System Default’ dashboard to add it as a favorite.

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 favorite.

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 favorite, 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 favorite 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 favorite' link.

Editing an existing Dashboard Page’s details

You can always update the details, i.e. Name, Description, Sharing, favorite, 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 favorite 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 favorite 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 delete 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 favorite. 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 favorite 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.

Changing the Look and Behavior of a Gadget

On this page:

- Hiding or Changing the Color of the Gadget's Frame
- Minimising and Expanding a Gadget
- Opening the Maximized or Canvas View of a Gadget
- Editing a Gadget's Settings

Hiding or Changing the Color of the Gadget's Frame

You can change the color 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 color 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 color you want for your gadget's frame. To hide the gadget's frame, select the white color box with the red line through it.

Screenshot: Hiding or changing the color of a gadget's frame

Minimising and Expanding a Gadget

You can shrink (minimize) a gadget on your dashboard so that it displays only the top bar of the gadget frame.

- If you minimize 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 color of the gadget frame.
- You can minimize/expand a gadget even if you do not have update permissions on the dashboard.
- The minimize/expand setting is stored in a cookie, and is not saved to the dashboard server.

To minimize 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 ‘Minimize’.

**To expand a gadget that has been minimized,**

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 minimized gadget (Introduction Gadget)**

---

**Opening the Maximized 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 maximized 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 maximized or canvas view.
- You can open the canvas view of a gadget even if you do not have update permissions on the dashboard.
- The maximized/canvas view setting is stored in a cookie, and is not saved to the dashboard server.

**To open the maximized 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 maximize icon at top right of the gadget frame. This icon will appear only if the gadget provides a maximized or canvas view.
3. The gadget's maximized 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 maximized 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 customize 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.
5. Adjust the settings as required then click ‘Save’.

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, unless you are using Atlassian OnDemand.
- Activity from remote applications. 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:
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. JIRA, 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 behavior of the gadget.

### Adding the Administration Gadget

The Administration (Guide for JIRA Administrators) gadget displays handy checklist of common tasks. Click on any task and you’ll be taken to the relevant administration screen to complete it. The gadget also provides other helpful information for new administrators, such as links to the JIRA administrator documentation. This gadget displays on the default dashboard 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:

The Administration gadget is automatically displayed on the **default dashboard**. If you have removed it and want to restore it to the default dashboard, or you want to add it to a new dashboard, follow the instructions below:

1. Go to your JIRA dashboard and click **Add Gadget**. The 'Gadget Directory' will appear.
2. Locate the **Guide for JIRA Administrators** gadget and click the **Add it Now** button. Then click the **Finish** 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 behavior 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 Watched Issues 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:

   **Assigned to Me**
   
<table>
<thead>
<tr>
<th>T Key</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>QA-23</td>
<td>QA version 2.0 bacon issue fixed 2.0 - Would I get notified????</td>
</tr>
<tr>
<td>BULK-43</td>
<td>Sub-Task of Death!!!!!!</td>
</tr>
<tr>
<td>QA-40</td>
<td>would u show up in my Inbox?</td>
</tr>
<tr>
<td>BULK-70</td>
<td>Description testing</td>
</tr>
<tr>
<td>BULK-118</td>
<td>BULK-116 / Cross-Site Request Forgery</td>
</tr>
<tr>
<td>BULK-3</td>
<td>Unknown Unknown issue fixed 2.0</td>
</tr>
<tr>
<td>BULK-4</td>
<td>Version 1.1.1 cheese issue fixed 1.1.1why did summary vanish? saving</td>
</tr>
<tr>
<td>BULK-5</td>
<td>BULK-4 / version 2.0 bacon issue fixed 2.0</td>
</tr>
<tr>
<td>BULK-8</td>
<td>asdasdasd as</td>
</tr>
<tr>
<td>QA-24</td>
<td>QA bacon 1.1.1 unknown</td>
</tr>
</tbody>
</table>

   1–10 of 124

   1 2 3 4 5 6 7 8 9 ▸

**Note:** This gadget only displays issues that are *unresolved*.

### 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 Watched Issues 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:

   **Assigned to Me**
   
   **Number of Results:**
   
   Number of results to display (maximum of 50).

   **Fields to display**
   
   - Issue Type
   - Key
   - Priority
   - Summary

   Drag-drop to reorder the fields.

   **Add a field**

   Add fields to the list above by selecting them and clicking "Add".

   **Refresh Interval:**

   How often would you like this gadget to update

   - Never
   - Every 15 minutes
   - Every 30 minutes
   - Every hour
   - Every two hours

   **Save**
   
   **Cancel**

   **To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behavior 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:

![Average Age Chart: Book Request](image)

This chart shows the average number of days issues were unresolved for over a given period:

Period: last 30 days (grouped Daily)

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 behavior 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*

![Bamboo Charts Gadget](image)

Add 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'.
   - '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 behavior 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 administrator to have it approved.*

To fix this problem, you will need to configure your Bamboo site to allow JIRA to draw information from it.
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:

   ![Plan Summary Chart](image)

   Connected to [https://jira.bamboo.atlassian.com](https://jira.bamboo.atlassian.com)

   ![Graph Example](image)

   - ![Graph Example](image)

   Your JIRA administrator must have configured the Bamboo plugin on your JIRA server, if you want to add the Bamboo Plan Summary gadget to your dashboard. If you have added multiple Bamboo servers in JIRA there will be one Bamboo Plan Summary gadget available per server, e.g. 'Bamboo Plan Summary Gadget from http://172.20.5.83:8085', 'Bamboo Plan Summary 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 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.

   - ![Graph Example](image)

   Connected to [https://jira.bamboo.atlassian.com](https://jira.bamboo.atlassian.com)
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 behavior 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 administrator 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*
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 behavior 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 administrator 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:

![Bugzilla Issue ID Search gadget](image)

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:

![Bugzilla Issue ID Search gadget](image)

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 behavior 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.

Please note the JIRA Calendar plugin is required for this Gadget to be available.

What does it look like?

The **JIRA Issues Calendar** gadget should appear as follows on the dashboard:
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 Add it Now. Then click Finished.
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.

e. **Refresh Interval** — select how often you would like this calendar to be updated.

4. Click **Save**.

To move the gadget to a different position on the dashboard, simply drag-and-drop. You can also change the look and behavior 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*
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 behavior 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 administrator 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:
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 report showing this information is also available.
Created vs Resolved Chart: Book Request

Project or Saved Filter: Book Request

Quick Find

Advanced Search

Period: Daily

The length of periods represented on the graph.

Days Previously: 100

Days (including today) to show in the graph.

Cumulative Totals: Yes

Progressively add totals (1..2..3), or show individual values (1..1..1).

Display the Trend of Unresolved: No

Show the number of unresolved issues over time in a subplot.

Display Versions: Only major versions

Show when versions were released on the chart.

Refresh Interval: Never

How often would you like this gadget to update

Save  Cancel

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 behavior 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:

<table>
<thead>
<tr>
<th>Review Age (days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day+</td>
</tr>
<tr>
<td>Week+</td>
</tr>
<tr>
<td>Month+</td>
</tr>
</tbody>
</table>

Day+ 2 - 6
Week+ 7 - 29
Month+ 30+

Your JIRA administrator must have configured the FishEye plugin on your JIRA server, if you want to add the Crucible Charts gadget to your dashboard.

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 behavior of the gadget.

**Adding the Favorite Filters Gadget**

The Favorite Filters gadget displays a list of all the issue filters that have currently been added by you as a 'favorite' filter.

[i] Read more about adding an issue filter as a favorite filter in the issue filters documentation.

What does it look like?

The Favorite Filters gadget should appear as follows on the dashboard:
Adding the ‘Favorite Filters’ gadget to your Dashboard

1. Go to your JIRA dashboard and click ‘Add Gadget’.
2. The ‘Gadget Directory’ will appear. Locate the ‘Favorite Filters’ gadget and click the ‘Add it Now’ button. Then click the ‘Finished’ button at the bottom of the Gadget Directory.
3. The Favorite Filters gadget will appear on your dashboard as follows, ready for you to configure:

   a. ‘Show issue counts’ — select whether, for each of your favorite 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 behavior of the gadget.

The Favorite Filters gadget is added by default to the 'System Default' dashboard.

The ‘Favorite 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:
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:
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 favorite 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. **Fields to display** — select the issue fields to display as columns. Drag and drop to re-order.

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 behavior of the gadget.

### 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:
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:

Your JIRA administrator must have configured the FishEye plugin on your JIRA server (Not applicable to JIRA OnDemand), if you want to add the FishEye Charts gadget to your dashboard.
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 behavior 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:
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:

   - **FishEye URL** – type the URL of your FishEye server.
   - **Repository** – type the name of your FishEye repository.
   - **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.
   - **Number of Results** – type the number of commits that you want the gadget to display.
   - **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).

   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 behavior of the gadget.

Adding the ‘FishEye Recent Changesets’ gadget to your Dashboard

Your JIRA administrator must have configured the FishEye plugin on your JIRA server (Not applicable to JIRA OnDemand), if you want to add the FishEye Recent Changesets gadget to your dashboard.
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:

![Issues in Progress gadget image]

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:

![Issues in Progress gadget configuration image]

   a. **Number of Results** — type the maximum number of issues that you want the gadget to display per page.
   b. **Fields to display** — select the fields that you want the gadget to display.
   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 behavior 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 when logged into JIRA:

![Introduction gadget example](image)

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 behavior of the gadget.

Configuring the Introduction gadget

The text/html displayed in the **Introduction** gadget is configured by your JIRA administration, through the **JIRA configuration** page. You may also click on the link in the displayed text box (shown above) to display this page:

![Introduction configuration](image)

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 behavior 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:

![JIRA: News gadget image]

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 behavior 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:
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 behavior 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.
1. Select 'Internet Options' from the 'Tools' menu:

![Internet Options](image)

2. The 'Internet Options' window will display. Click the 'Settings' button in the 'Temporary Internet files' (i.e. cache) section:

![Settings](image)

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:
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:

```
Quick Links

My Unresolved Reported Issues | Votes | Watches
Browse Projects | Search for Issues | Create Issue | Administration
```

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 behavior of the gadget.

**Adding the Recently Created Chart Gadget**

The *Recently Created Chart* 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 Chart’ gadget will appear as follows on the dashboard:

![Recently Created Chart](image)

Click the *more detail* link to go to the full-size report and data table.

Adding the ‘Recently Created Chart’ gadget to your Dashboard

1. Go to your JIRA dashboard and click ‘Add Gadget’.
2. The ‘Gadget Directory’ will appear. Locate the ‘Recently Created Chart’ gadget and click the ‘Add it Now’ button. Then click the ‘Finished’ button at the bottom of the Gadget Directory.
3. The Recently Created Chart gadget will appear on your dashboard as follows, ready for you to configure:

![Recently Created Chart Configuration](image)

   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' — type 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 behavior 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:
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:

   - '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.
   - 'Period' — select the timeframe on which the chart will be based: 'Hourly' / 'Daily' / 'Weekly' / 'Quarterly' / 'Yearly'
   - 'Days Previously' — enter the number of days' worth of data (counting backwards from today) to be included in the chart.

If you maximise the gadget, you can also view the data table on which the graph is based.
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 behavior** 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:

![Internet Options window](image1)

2. The **Internet Options** window will display. Click the **Settings** button in the **Temporary Internet files** (i.e., cache) section:

![Internet Options settings](image2)
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.

![](image.png)

**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:

![](image.png)

You can:

- Click the name of a project to browse the project.
- Click the name of a version 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:
Adding the Text Gadget

The Text gadget displays your specified HTML text on the dashboard.

This gadget is only available if your JIRA administrator has enabled it. 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.

To enable the text gadget: Choose Add-ons. The 'Find add-ons' screen shows add-ons available via the Atlassian Marketplace. Choose Manage Add-ons to view the plugins currently installed on your JIRA site. Enable the Text module in the Atlassian JIRA - Plugins - Gadgets Plugin (You need to select the System add-ons from the drop-down).
What does it look like?

The **Text** gadget should appear as follows on the dashboard:

![Text gadget example](image)

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:

![Text gadget configuration](image)
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 behavior 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](https://example.com), 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:

![Time Since Chart](https://example.com/)

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](https://example.com) 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 behavior 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:

![Two Dimensional Filter Statistics gadget](image_url)

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 field¹ on which the X-axis will be based.
c. ‘Y Axis’ — select an issue field¹ 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 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 behavior 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 **Watched 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:

   - **Number of results** — specify the maximum number of issues you wish the gadget to display per page.
   - **Fields to display** — select the **issue fields** to display as columns. Drag and drop to re-order.
   - **Show total votes** — select this if you wish the gadget to display the number of people who have voted for each issue.
   - **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.
   - **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 behavior** 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:
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. **Fields to display** — select the issue fields to display as columns. Drag and drop to re-order.
   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 behavior 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:

![Heat Map Gadget Example](image)

**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 behavior 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:

![Labels Gadget Example](image)
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:

![Labels Gadget](image)

   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.
   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 behavior 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:

- Choose your user name at top right of the screen, then choose Profile.
On this page:
- Using the Summary tab
- Using the Roadmap tab
- Using the Tools menu

Related topics:
- Customizing the Dashboard
- Saving Searches ('Issue Filters')

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.
  - Password – you are required to enter your password to save changes to your Full Name and Email address.
  - Click the Update button to save your changes.

- Click the Avatar icon to add an avatar to your user profile.
  - If you have already added a user avatar to your JIRA profile, that (current) avatar will appear instead of the icon above. Clicking your current avatar allows you to change it.
- Click the Administrator User link to view or edit your user details in JIRA’s administration area.
  - This option is only available to you 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:

  ![Update User Preferences](image)

  - 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 / Sharing (in dialog box)** – Choose the default 'sharing' setting for when you create new filters and dashboards, which can be either shared with all other users (Public) or restricted to your viewing only (Private).

• **Keyboard Shortcuts** – Choose between making JIRA's Keyboard Shortcuts feature either Enabled or Disabled.

• **Autowatch** – Choose between allowing JIRA to automatically make you a watcher of any issues that you create or comment on.

  • Click **View Navigator Columns** to choose which fields appear in your Issue Navigator.

**Tip:** The global defaults for most of the user preferences above can be set by your JIRA administrator (Administration > User Preferences). Your preferences will override the global preferences, but if you haven’t explicitly changed any settings they will be inherited from the global settings.

**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)
    ![Tip: This requires your administrator to configure a two-way Application Link, unless you are using Atlassian OnDemand.]
  - Activity from remote applications. 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.

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.

Choosing a User Avatar

There are two methods for choosing a user avatar depending on whether your administrator has enabled Gravatar for user avatars for your JIRA installation. If Gravatar has been enabled, you will not be able to choose JIRA-specific user avatars and vice versa.

Gravatar enabled

If Gravatar has been enabled, your Gravatar (i.e. the Gravatar associated with the email address in your user profile) will automatically be set as your user avatar.

To change your Gravatar, log in to Gravatar.com and follow the instructions on that site.

Gravatar disabled

If Gravatar has been disabled, you can choose your user avatar from the ones pre-packaged with JIRA or upload your own.

To choose your user avatar:

1. Choose your user name at top right of the screen, then choose Profile.

2. In the Details section, click the Avatar icon to open the Select a User Avatar dialog box.
   - If you have already added a user avatar to your JIRA profile, that (current) avatar will appear instead of the icon above. Clicking your current avatar allows you to change it.
JIRA comes pre-packaged with its own set of user avatars, which appear in the first few rows of this dialog box.

3. From this point, you can choose one of JIRA's pre-packaged user avatars, upload your own custom avatar or choose a user avatar which you have previously uploaded:
   - **To choose one of JIRA’s pre-packaged user avatars or one which you have previously uploaded:**
     - Click the user avatar on this dialog box. Your JIRA user account will use this avatar immediately.
     - User avatars which you have previously uploaded to JIRA will appear after JIRA's pre-packaged user avatars on this dialog box.
   - **To upload a new or custom user avatar:**
     a. Click the *Browse* button and in the resulting dialog box, browse for and choose an image file.
     b. Click and drag the centre of the superimposed square, whose content will eventually be cropped to become your new user avatar.
     - If desired, drag the corners of the square to re-size the area of the superimposed square. (You may need to re-centre the square again.)
     c. Click the *Confirm* button to create your new custom user avatar. Your JIRA user account will use this avatar immediately.
   - **Please Note:**
     - Your cropped image is re-sized to 48x48 pixels before it is saved in JIRA as your new custom user avatar.
     - 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:

<table>
<thead>
<tr>
<th>Consumer</th>
<th>Consumer Description</th>
<th>Issued on</th>
<th>Expires on</th>
<th>Actions</th>
</tr>
</thead>
</table>

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. Choose your user name at top right of the screen, then choose Profile.
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

<table>
<thead>
<tr>
<th>Consumer</th>
<th>Consumer Description</th>
<th>Issued on</th>
<th>Expires on</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created vs Resolved Chart</td>
<td>Atlassian Refimpl at <a href="http://localhost:8080/dashboards">http://localhost:8080/dashboards</a></td>
<td>02/10/2009</td>
<td>09/10/2009</td>
<td>Revoke OAuth Access Token</td>
</tr>
</tbody>
</table>

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 page at http://localhost:8090 says:

If you revoke the access token, the application Activity Stream will no longer be able to access data using your account.

Hint: If this application accesses your data via a gadget, you can restore the permission later by clicking the lock icon on the gadget.

Click 'OK' to revoke the access token.

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

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consumer</td>
<td>The name of the JIRA gadget that was added on the consumer.</td>
</tr>
<tr>
<td>Consumer Description</td>
<td>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 customize this Consumer Info detail.</td>
</tr>
<tr>
<td>Issued On</td>
<td>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).</td>
</tr>
<tr>
<td>Expires On</td>
<td>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.</td>
</tr>
<tr>
<td>Actions</td>
<td>The functionality for revoking the access token.</td>
</tr>
</tbody>
</table>

### Changing your Password

To change your JIRA password:

1. Choose your user name at top right of the screen, then choose Profile.
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.
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. Choose your user name at top right of the screen, then choose Profile.
2. In the 'Preferences' section, click the edit icon at the top-right to open the Updated User Preferences dialog box.
3. Select your language from the Language drop-down list.
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.

If the time zone specified in your JIRA user profile doesn’t match the time zone of the computer you are working on, JIRA will prompt you to ask if you want to change the time zone setting in your JIRA user profile.

#### To choose a time zone:

1. Choose your user name at top right of the screen, then choose Profile.
2. In the ‘Preferences’ section, click the edit icon at the top-right to open the Updated User Preferences dialog box.
3. Select your region (or country) and time zone from the Time Zone drop-down list.
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.

### Changing your JIRA Home Page

Your JIRA home page is the JIRA page you are presented with immediately after you log in.

You can configure the following JIRA pages as your JIRA home page:

- The Dashboard
- The Issue Navigator
- The Rapid Board *(available if you have the JIRA Agile add-on installed in JIRA)*

#### To configure your JIRA home page:

1. Choose your user name at top right of the screen, then choose Profile.
2. Select the appropriate home page option within the My JIRA Home section:
   - Dashboard
   - Issue Navigator
   - Agile *(i.e. the JIRA Agile Rapid Board — this option is only available if you have the JIRA Agile add-on installed in JIRA)*

   - Your page will be reloaded the JIRA home page you selected.
3. *(Optional)* To verify that your JIRA home page has been reset, log out and log back in to JIRA again. You should be taken directly to the JIRA home page you selected in the previous step.

### Requesting Add-ons

The Atlassian Marketplace website offers hundreds of add-ons that the administrator of your Atlassian application can install to enhance and extend JIRA. If the add-on request feature is enabled for your JIRA instance, you can submit requests for add-ons from the Marketplace to your JIRA administrator.

The ‘Atlassian Marketplace for JIRA’ page presents an integrated view of the Marketplace website from within the JIRA user interface. The page offers the same features as the Marketplace website, such as add-on search and category filtering, but tailors the browsing experience to JIRA.

This in-product view of the Marketplace gives day-to-day users of the Atlassian applications, not just administrators, an easy way to discover the add-ons that can help them work. When you find an add-on of interest, you can submit a request with just a few clicks.

#### Submitting an add-on request

To browse for add-ons in the Atlassian Marketplace, follow these steps:
1. From anywhere in the application, open your profile menu and choose **Atlassian Marketplace**.
2. In the Atlassian Marketplace page, use the search box to find add-ons or use the category menus to browse or filter by add-ons by type, popularity, price or other criteria. You can see what your fellow users have requested by choosing the **Most Requested** filter.
3. When you find an add-on that interests you, click **Request** to generate a request for your administrator.
4. Optionally, type a personal message to your administrators in the text box. This message is visible to administrators in the details view for the add-on
5. When ready, click **Submit Request**.
6. Click **Close** to dismiss the 'Success!' message dialog box.

At this point, a notification appears in the interface your administrators use to administer add-ons. Also your request message will appear in the add-on details view, visible from the administrator's 'Find New Add-ons' page. From there, your administrator can purchase the add-on, try it out or dismiss requests.

**Updating an add-on request**

After submitting the request, you can update your message at any time. Click the **Update Request** button next to the listing in the 'Atlassian Marketplace' page to modify the message to your administrator.

The administrator is not notified of the update. However, your updated message will appear as you have modified it in the details view for the add-on immediately.

**Data collection policy**

**Why does JIRA collect anonymous data usage?**

When you enable collection of usage data for an instance of JIRA (and certain Atlassian add-ons for JIRA), a variety of information about events occurring in that instance is collected to help us improve the user experience. Our goal is to capture these events to help us understand the features that are being used and how best to improve them. Most events are triggered as a result of a user interacting with the software, though some events are collected during background processing. Collecting this data often also involves collecting information about the configuration related to the features being used. The common example is an event triggered when a user clicks to invoke a particular function, which in turn would include information about the configuration related to that function.

As part of our data collection, we typically capture the hostname from which an event originated, but we do not associate the event with an identifiable user. This is what we mean by "anonymous" data collection. We use the data we collect to determine the features that are being used and how best to improve them. All the data we collect is subject to the terms of our Privacy Policy and our End User Agreement.

**Enabling/disabling data collection in JIRA**

During JIRA setup, we ask you whether or not you wish to help us improve JIRA by allowing us to capture anonymous usage data. This choice can be changed at any time by logging in to JIRA with administration privileges and navigating to the following location:

**Administration > System > General Configuration > Enable Atlassian analytics**

This only applies to the downloaded version from JIRA 6.1 onwards. Previous versions of JIRA (in and of themselves) did not capture any usage data and thus do not have this administration setting.

As an important caveat, however, please note that some Atlassian add-ons that are compatible with JIRA versions prior to 6.1 have their own analytics settings (listed further below). The above JIRA setting does not change the analytics option for any these add-ons. The JIRA add-ons that collect analytics (apart from the data collection in JIRA itself) is described below.

Another important caveat is that this setting does not apply to our OnDemand service. Data collection is permitted and described in our Privacy Policy and End User Agreement and cannot be disabled for OnDemand.

**Atlassian add-ons for JIRA that collect anonymous data**

Atlassian add-ons for JIRA are also able to collect anonymous data in the same way that JIRA itself collects anonymous data. As Atlassian is always looking for ways to improve the total JIRA experience, this section also covers future add-ons that have not been built or released.
Prior to JIRA 6.1, the Atlassian add-ons for JIRA listed below had their own individual analytics setting to determine if they captured anonymous data or not. Instructions on how to disable them are described in the section below. Instructions on how to disable future add-ons that are compatible with releases prior to JIRA 6.1 will also be added to this section as applicable.

- **JIRA Agile** (formerly known as GreenHopper)
- **JIRA Capture** (formerly known as Bonfire)
- **JIRA Importers**
- **InProduct Translation**

### Enabling/disabling data collection in add-ons

Anonymous data collection can be disabled by logging in to JIRA with administration privileges and navigating to the following locations:

- **JIRA Agile** (formerly known as GreenHopper)
  Administration > Add-Ons > JIRA Agile > JIRA Agile Labs > Analytics
- **JIRA Capture** (formerly known as Bonfire)
  Administration > Add-Ons > JIRA Capture > JIRA Capture Settings > Analytics
- **JIRA Importers**
  Administration > System > Import & Export > External System Import > Stop collecting my information
- **InProduct Translation**
  Administration > Add-Ons > Translation Administration > Google Analytics

### Data Collection Methods

JIRA uses two different methods to collect usage data:

1. Atlassian Analytics
2. Google Analytics

### Atlassian Analytics

At this time, the Atlassian infrastructure to capture usage data is not yet built. We plan to have this ready by late 2013 or early 2014. Atlassian plans to store all the JIRA system and user events in a file under the JIRA home directory. This file will then be periodically sent to Atlassian and then emptied. If there is no internet connection available, then no data will be sent.

As the Atlassian Analytics infrastructure does not exist yet, this section will be updated with the most accurate information when our system becomes live.

### Google Analytics

Atlassian also uses a service provided by Google to capture anonymous usage data; this service is [Google Analytics](https://analytics.google.com).

The data captured from Google Analytics is stored within Google for Atlassian staff to access. By default Google Analytics captures (among other things) the number of page hits, how much time is spent on each page, where users come from and leave to, demographics, and browser information.

### JIRA Administrator's Guide

This manual contains information on administering your JIRA system:

- [Getting Help](#)
**Configuring the Layout and Design**
- Customizing 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
- Creating Links in the Application Navigator
- Configuring Issue Table Columns

**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
- Password Policy for JIRA
  - Clearing 'Remember my login' Tokens
  - Disabling Remember My Login on this Computer
- Enabling Public Signup and CAPTCHA

**Project Management**
- Defining a Project
  - Editing a Project Key
    - Changing the Project Key Format
  - Simple Issue Tracking project
  - Software Development project
- Managing Project Role Membership
- Defining a Component
- Managing Versions
  - Running a Bamboo Build when Releasing a Version
  - Creating Release Notes

**Configuring Security**
- Configuring Issue-level Security
- Managing Project Permissions
- Managing Global Permissions
- Configuring Secure Administrator Sessions
- Preventing Security Attacks
- JIRA Cookies
- JIRA Admin Helper

**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 Behavior
  • Associating Field Behavior with Issue Types
  • Configuring Renderers
• Defining a Screen
  • Associating a Screen with an Issue Operation
  • Associating Screen and Issue Operation Mappings with an Issue Type

Configuring Workflow
• Activating workflow
• Classic mode workflow designer
• Configuring workflow schemes
• Working in text mode
• Sharing your workflow
• Advanced workflow configuration
  • Adding a custom event
  • Configuring the initial status
  • Configuring JIRA with HipChat
  • Using validators with custom fields
  • Using XML to create a workflow
  • Workflow properties

Configuring Email
• Configuring Email Notifications
  • Configuring JIRA's SMTP Mail Server to Send Notifications
  • Creating a Notification Scheme
  • Customizing 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 FogBugz for Your Server
• Importing Data from FogBugz On Demand
• 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
• Importing Data from JSON (beta release)
• Importing Data from Redmine

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 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
  • Configuring Advanced Settings
• 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 Issue Cloning
• Configuring Issue Linking
• Configuring the Whitelist
• Configuring Sub-tasks
• Managing Shared Filters
• Managing Shared Dashboards
• Linking to Another Application

**Server Administration**
• Increasing JIRA Memory
• Using the Database Integrity Checker
• Precompiling JSP pages
• Logging and Profiling
  • Logging email protocol details
• Restoring Data
  • Restoring a Project from Backup
• Optimizing 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
• Monitoring Database Connection Usage
• Viewing JIRA's Instrumentation Statistics
• Generating a Thread Dump
• Finding the JIRA Support Entitlement Number (SEN)
• Performance Testing Scripts

**Appendix A - Extending JIRA**
• Managing Add-ons
• Using the Issue Collector
  • Advanced Use of the JIRA Issue Collector
• Listeners
• Services
• Jelly Tags
• JIRA Toolkit (Customer Support Extensions)
• Developer Guides
• Building JIRA from Source
  • How to Make a JIRA Patch
• API Documentation
• Managing Webhooks
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. Choose \( \text{\textbullet\ System} \). Select Atlassian Support Tools from the left panel in the System page.
3. Click Support Request to display this form:
Please provide as much information as possible, including any error messages that are appearing on the console or in the logs.

4. 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. Choose `system` > `System`. Select Atlassian Support Tools from the left panel in the System page.
3. Click Support Zip. 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:

- Customizing the Look and Feel
- Choosing a Default Language
Customizing the Look and Feel

This page tells you how to customize your JIRA installation to match your company's environment. One of the easiest things you can do to get started is to update your JIRA color scheme to match your company's logo (shown below).

- **Upload from File** – click *Browse* to search for and upload a new image for the logo.
- **Upload from URL** – use one of the following conventions:
  - 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.

**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.

If the JIRA logo 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.

If you don't like the change, simply click *Undo*.
Look and feel configuration

You can easily customize JIRA’s look and feel to suit your needs:

1. Log in as a user with the JIRA Administrators global permission.

2. Choose ⚙️ > System. Select 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

   Logo, Site Title and Favicon
Colours

To apply a color scheme that matches your logo, click here.

<table>
<thead>
<tr>
<th>Colour Name</th>
<th>Hex Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Background Colour</td>
<td>#209881</td>
</tr>
<tr>
<td>Header Highlight Background Colour</td>
<td>#296CA3</td>
</tr>
<tr>
<td>Header Separator Colour</td>
<td>#2E3D54</td>
</tr>
<tr>
<td>Header Text Colour</td>
<td>#FFFFFF</td>
</tr>
<tr>
<td>Header Text Highlight Colour</td>
<td>#0F0F0F</td>
</tr>
<tr>
<td>Menu Item Highlight Background Colour</td>
<td>#326ca6</td>
</tr>
<tr>
<td>Menu Item Highlight Text Colour</td>
<td>#FFFFFF</td>
</tr>
<tr>
<td>Button Background Colour</td>
<td>#3B7B45</td>
</tr>
<tr>
<td>Button Text Colour</td>
<td>#FFFFFF</td>
</tr>
<tr>
<td>Link Active Colour</td>
<td>#326ca6</td>
</tr>
<tr>
<td>Heading Colour</td>
<td>#292929</td>
</tr>
<tr>
<td>Link Colour</td>
<td>#326ca6</td>
</tr>
</tbody>
</table>

Gadget Colours

<table>
<thead>
<tr>
<th>Colour Name</th>
<th>Hex Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Colour 1 (Default)</td>
<td>#369</td>
</tr>
<tr>
<td>Colour 2</td>
<td>#900925</td>
</tr>
<tr>
<td>Colour 3</td>
<td>#D46270</td>
</tr>
<tr>
<td>Colour 4</td>
<td>#346006</td>
</tr>
</tbody>
</table>

Day/Time Formats

<table>
<thead>
<tr>
<th>Format Type</th>
<th>Format Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Format</td>
<td>h:mm a</td>
</tr>
<tr>
<td>Day Format</td>
<td>EEEE h:mm a</td>
</tr>
<tr>
<td>Complete Date/Time Format</td>
<td>dd/MMM/yyyy h:mm a</td>
</tr>
<tr>
<td>Day/Month/Year Format</td>
<td>dd/MMM/yyyy</td>
</tr>
</tbody>
</table>

Use ISO8601 standard in Date Picker

Turning it on will cause Monday to be the first day of the week in the Date Picker, as specified by the ISO8601 standard.

Refresh Client Resources

If you have applied patches or made other support changes it can be necessary to cause all browser clients of JIRA to refresh fresh copies of web resources.

Clicking this link will ensure users see those updates.

Logo and Favicon

The logo appears in the top left corner of every JIRA page while the favicon appears typically to the left of 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.

<table>
<thead>
<tr>
<th>Option</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preview (Logo)</td>
<td>A preview of your JIRA site's current logo is shown here.</td>
</tr>
<tr>
<td>Favicon Preview (Favicon)</td>
<td>A preview of your JIRA site's current favicon is shown here.</td>
</tr>
</tbody>
</table>

colors

The following options control the appearance of the entire JIRA user interface.

**Editing colors**

To edit the colors, click on the individual colors and follow this procedure.

1. Click on the color box for an element.
2. This opens up the color display where you can create customized colors or enter specific color values:

3. To save your changes, click **Update**.
4. If you are unhappy with a color change, click the **Revert** button that displays in the row where you've made the change:

**Usage Notes**

- The colors 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 color, you can use the pop-up color chooser, or specify your own (eg. `#FFFFFF`, 'red').
- To return to the original color scheme, just clear any values that you have set.

**Gadget colors**

These seven colors are the seven options from which users can select when changing the color of a gadget's frame on their JIRA dashboard. color 1 is the default frame color for newly-added gadgets.

**Please note:**

- The colors 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 color, you can use the pop-up color chooser, or specify your own (eg. `#FFFFFF`, 'red').
- To return to the original color scheme, just clear any values that you have set.
Date/Time Formats

The Look and Feel page allows you to customize the way times and dates are presented to users throughout the JIRA user interface.

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.

Relative time is used in date/time formats

Issue date/time fields show a relative instead of absolute date/time format (for example: Yesterday 12:00 PM instead of 20 May 2013 12:00 PM). You can still see the absolute date/time by hovering over the field.

The date/time format reverts to absolute after a week.

If you want to switch off this format, set the jira.lf.date.relativize application property to 'false'. See Advanced JIRA Configuration for more information.

Configuring date picker formats

JIRA system administrators can configure the format of date pickers used throughout the JIRA user interface via options on the Advanced Settings page.

Be aware that these options are different from the Date/Time Formats configuration options on the Look and Feel page, which only customize JIRA's presentation of times and dates to users.

The date or date/time formats for date pickers are defined by a pair of properties (one for Java and the other for JavaScript). The two properties in this Java/JavaScript pair must match in order for the date (or date/time) picker they define to function 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:

<table>
<thead>
<tr>
<th>Preferred Date</th>
<th>Value of the jira.datepicker.java.format property</th>
<th>Value of the jira.datepicker.jsformat property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-10-01</td>
<td>yyyy-MM-dd</td>
<td>%Y-%m-%d</td>
<td>ISO 8601 format</td>
</tr>
<tr>
<td>Oct/1/10</td>
<td>MMM/d/yy</td>
<td>%b/%e/%y</td>
<td></td>
</tr>
<tr>
<td>10/01/10</td>
<td>MM/dd/yy</td>
<td>%m/%d/%y</td>
<td></td>
</tr>
<tr>
<td>Oct 1, 2010</td>
<td>MMM d, yyyy</td>
<td>%b %e, %Y</td>
<td></td>
</tr>
<tr>
<td>10/01/2010</td>
<td>MM/dd/yyyy</td>
<td>%m/%d/%Y</td>
<td></td>
</tr>
</tbody>
</table>

Here are some examples of date/time configurations:

<table>
<thead>
<tr>
<th>Preferred Date/Time</th>
<th>Value of the jira.datepicker.java.format property</th>
<th>Value of the jira.datepicker.jsformat property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-10-15 08:50</td>
<td>yyyy-MM-dd HH:mm</td>
<td>%Y-%m-%d %H:%M</td>
<td>ISO 8601 format</td>
</tr>
<tr>
<td>15/Oct/10 8:50 AM</td>
<td>dd/MMM/yyyy h:mm a</td>
<td>%d/%b/%y %I:%M %p</td>
<td></td>
</tr>
<tr>
<td>10/15/10 08:50 AM</td>
<td>MM/dd/yyyy hh:mm a</td>
<td>%m/%d/%Y %I:%M %p</td>
<td></td>
</tr>
</tbody>
</table>

Choosing a Default Language
Overview

Most user-visible pages in JIRA are now internationalized. Chinese, Czech, Danish, English, French, German, Italian, Norwegian, 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:

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Choose System. Select General Configuration to open the Administration page.
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.

There is also a plugin currently in Beta release that allows you to translate most JIRA items on the fly: InProduct Translations.

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

JIRA lets you change the columns of the table of search results for any search results displayed using the List view (as opposed to the Detail View). Click Columns at top right of the issue table to open the column configuration dialog, shown below.
**Column Configuration Dialog**

This displays the list of the columns used in the current table of results. Choose the columns you want with checkboxes and click **Done** to finish. Notice that the Filter option is greyed out, this is because the the issue table results are not coming from a filter. See Changing the column configuration for your own filters for an example of using this dialog to set the displayed columns for your own filters.

**Sorting and rearranging columns**

- To sort issues, just click on a column header.
- To rearrange the column layout, press and hold the mouse button to enter “column drag mode.”

**My Defaults, Filter, and System**

If the currently selected button is **My Defaults**, this indicates that the columns you are seeing are from your user account preferences. **Filter** is an available option whenever the issue search results come from a saved filter. If you are a JIRA Admin, you will also see the **System** tab, where you can change the columns for all users who have not set their own defaults.

JIRA administrators can configure the columns that appear in the Issue Navigator for all users that do not have personal column filters defined. When administrators are configuring default 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. JIRA administrators can also select which views are available in the JIRA system, as views are configurable via plugins.

**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.

*Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
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.

Adding and Configuring Gadgets on the Default Dashboard

JIRA's default dashboard is limited to only one dashboard page. However, users can add multiple pages to their own dashboards if they wish.

1. Log in as a user with the JIRA Administrators global permission.
2. Choose System > System User Interface > System Dashboard to open the Configure System Dashboard page.
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 Behavior of a Gadget.
   - To choose a different layout for the default dashboard, please see Customizing the Dashboard.

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
- Customizing the Dashboard

Using Dashboard Gadgets

On this page:

- About gadgets
- Pre-installed 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
Pre-installed gadgets

JIRA provides a set of standard gadgets out-of-the-box:

<table>
<thead>
<tr>
<th>Gadget</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Stream Gadget</td>
<td>The Activity Stream gadget displays a summary of your recent activity.</td>
</tr>
<tr>
<td>Administration Gadget</td>
<td>The Administration (Guide for JIRA Administrators) gadget displays checklist of common administration tasks and links to administrative functions and documentation.</td>
</tr>
<tr>
<td>Assigned To Me Gadget</td>
<td>The Assigned To Me gadget displays all open issues in all projects assigned to the current user viewing the dashboard.</td>
</tr>
<tr>
<td>Average Age Gadget</td>
<td>The Average Age gadget displays a bar chart showing the average number of days that issues have been unresolved.</td>
</tr>
<tr>
<td>Bamboo Charts Gadget *</td>
<td>The Bamboo Charts gadget displays various charts and plan statistics from a particular Bamboo server.</td>
</tr>
<tr>
<td>Bamboo Plan Summary Chart Gadget *</td>
<td>The Bamboo Plan Summary gadget displays a graphical summary of a build plan.</td>
</tr>
<tr>
<td>Bamboo Plans Gadget *</td>
<td>The Bamboo Plans gadget displays a list of all plans on a Bamboo server, and each plan's current status.</td>
</tr>
<tr>
<td>Bugzilla ID Search Gadget</td>
<td>The Bugzilla ID Search gadget allows the user to search all JIRA issues for references to Bugzilla IDs.</td>
</tr>
<tr>
<td>Calendar Gadget *</td>
<td>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.</td>
</tr>
<tr>
<td>Clover Coverage Gadget *</td>
<td>The Clover Coverage gadget displays the Clover coverage of plans from a particular Bamboo server.</td>
</tr>
<tr>
<td>Created vs Resolved Gadget</td>
<td>The Created vs Resolved gadget displays a difference chart showing the issues created vs resolved over a given period.</td>
</tr>
<tr>
<td>Crucible Charts Gadget *</td>
<td>The Crucible Charts gadget displays various charts showing statistical summaries of code reviews.</td>
</tr>
<tr>
<td>Favorite Filters Gadget</td>
<td>The Favorite Filters gadget displays a list of all the issue filters that have currently been added by you as a favorite filter.</td>
</tr>
<tr>
<td>Filter Results Gadget</td>
<td>The Filter Results gadget displays the results of a specified issue filter.</td>
</tr>
<tr>
<td>FishEye Charts Gadget *</td>
<td>The FishEye Charts gadget displays two charts showing statistics about a given sourcecode repository.</td>
</tr>
<tr>
<td>FishEye Recent Changesets Gadget *</td>
<td>The FishEye Recent Changesets gadget displays a number of recent changesets from a FishEye repository.</td>
</tr>
<tr>
<td>Gadget Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>In Progress Gadget</td>
<td>The <strong>In Progress</strong> gadget displays all issues that are currently in progress and assigned to the current user viewing the dashboard.</td>
</tr>
<tr>
<td>Introduction Gadget</td>
<td>The <strong>Introduction</strong> gadget displays a configurable introduction message on the dashboard.</td>
</tr>
<tr>
<td>Issue Statistics Gadget</td>
<td>The <strong>Issue Statistics</strong> gadget displays the collection of issues returned from a specified filter, broken down by a specified field.</td>
</tr>
<tr>
<td>JIRA: News Gadget</td>
<td>The <strong>JIRA: News</strong> gadget displays recent Atlassian news about JIRA.</td>
</tr>
<tr>
<td>Pie Chart Gadget</td>
<td>The <strong>Pie Chart</strong> 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.).</td>
</tr>
<tr>
<td>Projects Gadget</td>
<td>The <strong>Projects</strong> gadget provides information and various filters related to a specified project(s).</td>
</tr>
<tr>
<td>Quick Links Gadget</td>
<td>The <strong>Quick Links</strong> gadget displays a number of useful links to issues associated with the current user.</td>
</tr>
<tr>
<td>Recently Created Issues Gadget</td>
<td>The <strong>Recently Created Issues</strong> 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.</td>
</tr>
<tr>
<td>Resolution Time Gadget</td>
<td>The <strong>Resolution Time</strong> gadget displays a bar chart showing the average resolution time (in days) of resolved issues.</td>
</tr>
<tr>
<td>Road Map Gadget</td>
<td>The <strong>Road Map</strong> 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.</td>
</tr>
<tr>
<td>Text Gadget *</td>
<td>The <strong>Text</strong> gadget displays a configurable HTML text on the dashboard.</td>
</tr>
<tr>
<td>Time Since Issues Gadget</td>
<td>The <strong>Time Since Issues</strong> 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.</td>
</tr>
<tr>
<td>Two Dimensional Filter Statistics Gadget</td>
<td>The <strong>Two Dimensional Filter Statistics</strong> gadget displays statistical data based on a specified filter in a configurable table format.</td>
</tr>
<tr>
<td>Voted Gadget</td>
<td>The <strong>Voted Issues</strong> gadget shows issues for which you have voted.</td>
</tr>
<tr>
<td>Watched Gadget</td>
<td>The <strong>Watched Issues</strong> gadget shows issues which you are watching.</td>
</tr>
</tbody>
</table>

See the [big list of all Atlassian gadgets](https://marketplace.atlassian.com) 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 [Atlassian Marketplace](https://marketplace.atlassian.com). If you wish to 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

**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/my-gadget.xml

For example:
http://mycompany.com/jira/rest/gadgets/1.0/g/com.atlassian.streams-jira-plugin:activitystream-gadget/gadgets/activitystream-gadget.xml

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. 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. 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.)

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.

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. 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: Gadget directory with 'Gadget Subscriptions' button
Screenshot: 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 level can be configured to display to all users or just logged-in users.

If you are using downloadable JIRA, the banner can be configured to contain HTML text. If you are using JIRA OnDemand, you can only use wiki markup in the banner.
Configuring an Announcement Banner

1. Log in as a user with the JIRA Administrators global permission.
2. Choose ➕ System. Select User Interface > Announcement Banner in the System panel below.
   - 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. Choose ➕ System. Select General Configuration to open the Administration page.
   - Keyboard shortcut: `g` + `g` + start typing 'general configuration'
3. Locate the 'Options' section:
By default, JIRA will not prompt users to confirm logging out by default. To change this, click the **Edit Settings** button at the top of the page.

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).

**Creating Links in the Application Navigator**

You can add custom links in the application navigator, to make it easier for users to navigate to frequently used information.

**What is the application navigator?**

The application navigator is the control in the top left of the JIRA header that displays a menu of links to other applications. It is only displayed to users if there is more than one link. You can customize the links that appear in the application navigator, as well as making certain links only visible for specific users.

**Screenshot: Application navigator**

Adding links to the application navigator

If applications are linked to your JIRA instance via application links, those applications will automatically appear...
in the application navigator. If you don't have any applications linked, the application navigator icon () will appear only for administrators. After links have been set up, the application navigator icon will automatically be visible to all users.

2. Create links by entering a name and the URL on the page.
   After you've created a link, it will appear in the application navigator for all your applications after a few minutes (up to 10). Or, if you want links to appear immediately, you can navigate to the application navigator administration page in each application and refresh the page.

If you want to make a link appear in the application navigator for only specific users, use the Groups box to specify which groups can see the link. To hide the link from all users, select the Hide check box (for example, if you want to temporarily hide the link without deleting it entirely).

When you make a link visible for a specific group, the link visibility is only set up in the application where you are configuring the link. For example, if you change the visibility in the JIRA administration screen and you also want it to be visible to the same users in Confluence, you must make the same changes in the Confluence administration settings.

To modify links that were created and are managed in other applications (for example, in a different JIRA application), edit the link in that application. You cannot delete links to linked applications, you must delete the application link instead.

**Configuring Issue Table Columns**

JIRA lets you change the columns of the table of search results for any search results displayed using the List view (as opposed to the Detail View). Click **Columns** at top right of the issue table to open the column configuration dialog, shown below.
Column Configuration Dialog

This displays the list of the columns used in the current table of results. Choose the columns you want with checkboxes and click **Done** to finish. Notice that the Filter option is greyed out, this is because the the issue table results are not coming from a filter. See Changing the column configuration for your own filters for an example of using this dialog to set the displayed columns for your own filters.

Sorting and rearranging columns

- To sort issues, just click on a column header.
- To rearrange the column layout, press and hold the mouse button to enter "column drag mode."

My Defaults, Filter, and System

If the currently selected button is **My Defaults**, this indicates that the columns you are seeing are from your user account preferences. **Filter** is an available option whenever the issue search results come from a saved filter. If you are a JIRA Admin, you will also see the **System** tab, where you can change the columns for all users who have not set their own defaults.

JIRA administrators can configure the columns that appear in the Issue Navigator for all users that do not have personal column filters defined. When administrators are configuring default 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. JIRA administrators can also select which views are available in the JIRA system, as views are configurable via plugins.
Changing the column configuration for your own filters

If you are searching using a saved filter and if the filter is owned by you, use the Filter button to customize the columns displayed when users see results from that filter. When sharing a filter with other users, it's sometimes helpful to choose the relevant columns for those results. For example, if your filter searches for issues that are open bugs, you may decide to remove the columns for status and issue type for that filter since they will all be the same. Filters don't always have columns configured, but when they do, those columns will be shown unless the user chooses to use their defaults using the My Defaults button.

For any JIRA filters that you own, you can change the displayed columns as follows.

1. Click on the name of a JIRA filter you own.
2. Click the Columns button at top right of the currently displayed columns. This opens the column configuration dialog.
3. Select or deselect checked items in the list.
4. Click Done when you are finished.

Troubleshooting

If you cannot find a column, please make sure that you haven’t run in to any of the following restrictions:

- You can only see columns for issue fields that have not been hidden and that you have permissions to see.
- It is possible to add any of the existing custom fields to the column list, as long as the fields are visible and you have the right permissions.
- Some custom fields, even if selected, do 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.

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
- Password Policy for JIRA
- Clearing ‘Remember my login’ Tokens
- Enabling Public Signup and CAPTCHA

Managing Users
On this page:

- Viewing users
- Adding users
  - Who can see a user?
  - Creating a user
  - Inviting users
- 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
- Changing a username
- Adding a property to a user
- Deactivating a user
- Deleting a user
- Notes

Viewing users

To view a list of JIRA users:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose User Management. 
   - Keyboard shortcut: g + g + start typing users

3. To restrict the list of users shown, use the Filter form at the top of the page.
4. To view details and login information about a user in the list, click their Username or Email Address.

Adding users

Users can be created via any of the following methods:

- Add the user directly into JIRA — see Creating a user below. You can create one user at a time, using this method.
- Invite users via email — see Inviting Users below. You can invite multiple users at the same time, using this method.
- Allow users to sign up — see Enabling Public Signup.
- Automatically create users when issue/comment creation emails are received from unknown email addresses — You can use a mail handler to allow JIRA to create issues or comments via emails received. The handler can also be configured to create new users based on the sender's email address. See Creating Issues and Comments from Email.
- Connect 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 created will not have permission to log in to JIRA

Who can see a user?
Any JIRA user can see another JIRA user’s full name and username, for example, see Mark Lassau. The email address visibility is controlled by a configuration item; see “User email visibility” on the Configuring JIRA Options documentation page.

Group membership and the ability to edit users is only available to administrators.

Any JIRA user can see their own details, including group memberships, update their own password, and change certain user preferences (for example, time zone and language).

Creating a user

To create a user:

1. Open the User browser (see Viewing Users above) and click the Create User button to open the 'Create New User' dialog box.
2. Enter the Username, 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.

Inviting users

You can invite one or more users to JIRA via email. Note, JIRA’s SMTP mail server must be configured to send notifications before you can invite users via email.

To invite users to JIRA:

1. Open the User browser (see Viewing Users above) and click the Invite Users button to open the 'Invite Users' dialog box.
2. Enter the email addresses of the users that you want to invite. Enter each address on a new line or separate addresses using commas.
   Note, you cannot invite users by sending an invitation to a mailing list.
3. Click the Send button to send the invitations.
   - Each invitation can only be used to create a user under the email address that it was sent to, and can only be used once.
   - Each invitation will expire seven days after the day it was sent.
   - Your user license count will not be affected until users accept the invitation and the users are created.
   - Users that are created via the invitation will be added to the 'jira-users' group.

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.
   This will display two lists; the one on the left shows all Available Groups, and the one on the right shows the Current Groups to which the user currently belongs.
2. Choose a group(s) and click the Join selected groups or Leave selected groups buttons to add/remove the user from the selected groups.

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 assign a user to a project role:

1. 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:

   ![Project Roles Table]

2. 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

To change 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.
2. In the resulting form, make the required changes the user's Full Name and/or Email address.
   - Do not clear the Active check box unless you want to deactivate this user.
3. Click Update to confirm the change.

Changing a user's password

To change 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. Choose Actions > Set Password.
3. Enter and confirm the new password.
4. Click the Update button.

Changing a username

ℹ️ This feature is only available for downloadable instances of JIRA. It is not available in JIRA On Demand.

JIRA Administrators can edit any Username in the JIRA Internal Directory (this is often referred to as the "rename user" feature). This ability is important to have if you wish to connect JIRA to an LDAP directory that does not follow the same username conventions. You also may want to do this if a staff member wishes to change their surname. Once the Username is changed in the internal directory, all parts of the JIRA interface display the updated Username.

Note: The Username is for the JIRA Internal Directory, and should not be confused with the user's display name, or Full Name, in the JIRA system.

There are some important exceptions that will prevent you from using this feature, of which you should be
aware:

- Only **JIRA Administrators** can perform this function.
- JIRA cannot update **external** users – for example, users that are coming from an LDAP server or Crowd instance – it can only update users stored in the JIRA Internal Directory. (However, JIRA can update JIRA users stored in an "Internal Directory with LDAP Authentication.")
- If you are using your JIRA instance as a JIRA User Server for other applications, e.g., Confluence, you will not be able to use this feature. If you aren't sure about this, check under **User Management > JIRA User Server** to confirm that no external applications have been configured to use JIRA as a Crowd Server.

**Using a JIRA User Server for other applications**

When JIRA is being used as a **User Server** for other applications, e.g. Confluence, we don't allow a user rename on the JIRA server as this would not be recognised on the other server. (And the other application would think that user was deleted and a new user was added.)

If you are happy to accept this behavior, then you can set a flag to allow the rename, as documented in the Knowledge Base article: **Cannot rename users despite upgrading/installing JIRA 6**

We are hoping to add the ability to detect renames from a remote Crowd or JIRA server sometime soon, see **JRA-32200**.

**To change a username:**

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. Choose **Actions > Edit Details**.
3. Edit the **Username**.
4. Click the **Update** button.

**Adding a property to a user**

A 'Property' is an extra piece of information about a user that you can store in JIRA. A Property consists of a **Key** of your choice (eg. 'Phone number', 'Location') plus a corresponding **Value** (eg. '987 654 3210', 'Level Three'). Other than adding property data to the specified user, User Properties do not have an effect anywhere else in the project. Plugins, however, can frequently use this data.

**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. Choose **Actions > Edit Properties**. The **Edit User Properties** screen will be displayed:
3. Enter the new **Key** and its **Value**, then click the **Add** button.

**Deactivating a user**

JIRA administrators can 'deactivate' a JIRA user, which disables that user's access to JIRA. This avoids the need for a JIRA administrator to **delete the user's account** from the system.

This feature is useful when a JIRA user leaves an organisation because a deactivated user's history of JIRA activity is preserved on the system. If a user with a deactivated JIRA account rejoins the organisation at some point in the future, their JIRA user account can be re-activated.

**To deactivate a user account:**

1. Locate the user in the **User** browser (see **Viewing Users** above) and click their **Edit** link in the **Operations** column.
2. In the resulting form, clear the **Active** check box.
3. Click **Update** to confirm the change.

    - To re-enable the user again, repeat the steps above but instead, select the **Active** check box.

**While a JIRA user account has been deactivated, that user:**

- Will no longer be able to **log in to JIRA**.
- Cannot be assigned issues or added as a **watcher** to issues (whenever issues are **created** or **edited**).
However:

- A user who was assigned, was watching or had reported any issues in JIRA before their account is deactivated, will still appear as the respective assignee, watcher or reporter of those issues. This situation remains until another user is specified as the assignee or reporter of these issues, or the deactivated user is removed as a watcher from them.
- A user who voted on any issues in JIRA before their account is deactivated, will continue to appear as a voter on these issues.
- Will continue to appear on the JIRA user interface with '(Inactive)' displayed after their name, where applicable.
- Can still be used to filter issues in a JIRA search query.
- Will not receive any email notifications from JIRA, even if they continue to remain the assignee, reporter, or watchers of issues.
- Will not count towards your JIRA user license limit. Refer to the JIRA Users global permission explanation on Managing Global Permissions for more information.

Please Note:

- Users who are project or component leads cannot be deactivated. To deactivate these users, assign other users as the relevant project or component leads first.
- Any JIRA site’s users who are configured in an external Atlassian Crowd user directory and deactivated in Crowd, will be deactivated in JIRA.
- With the exception of JIRA users configured with 'delegated LDAP authentication', JIRA does not deactivate users who are configured and deactivated/disabled in an external Microsoft Active Directory or LDAP-based user directory.

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 history of activity on JIRA.

To delete a user:

1. Locate the user in the User browser (see Viewing Users above) and click the Delete link in the Operation column. 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.
2. Take any actions required to disassociate the user with JIRA. These may include:
   - Reassigning any issues assigned to the user.
   - Bulk-editing the issues created by the user and change the 'Reporter' to someone else. You'll need the 'Modify Reporter' permission to do this. 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.
   - Changing the owner of shared dashboards owned by the user. See Managing Shared Dashboards.
   - Changing the project lead for any projects that the user is a lead of.
3. If there are no issues assigned to, or reported by the user, and the user has not commented on any issues, the confirmation screen will display a Delete button. Click this to proceed with the deletion.

Please Note:

- You cannot delete a user from JIRA if they have performed any of the following actions:
  - reported or been assigned to any issues
  - commented on any issues
- 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...
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.

<table>
<thead>
<tr>
<th>JIRA's default groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewing groups</td>
</tr>
<tr>
<td>Adding a group</td>
</tr>
<tr>
<td>Deleting a group</td>
</tr>
<tr>
<td>Editing group membership</td>
</tr>
<tr>
<td>Automatic group membership</td>
</tr>
<tr>
<td>Notes</td>
</tr>
</tbody>
</table>

JIRA's default groups

When you install JIRA, three groups are automatically created:

<table>
<thead>
<tr>
<th>Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira-administrators</td>
<td>Typically contains people who are JIRA system administrators. By default, this group:</td>
</tr>
<tr>
<td></td>
<td>is a member of the 'Administrators' project role.</td>
</tr>
<tr>
<td></td>
<td>has the 'JIRA Administrators' and the 'JIRA System Administrators' global permissions.</td>
</tr>
<tr>
<td></td>
<td>i f 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''.</td>
</tr>
</tbody>
</table>
### 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. Choose 🔄 > User Management. Select Groups to open the Groups page.
3. **Keyboard shortcut:** `g + g + start typing groups`
4. To see which permission schemes, email notification schemes, issue security levels and saved filters are using this group, click the group name.

**Screenshot: the Group Browser**

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Users</th>
<th>Permission Schemes</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-team</td>
<td>1</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>administrators</td>
<td>9</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>angry-dets</td>
<td>16</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>argnments-deveads</td>
<td>1</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>argnments-qa</td>
<td>1</td>
<td>Default Permission Scheme</td>
<td>Delete</td>
</tr>
</tbody>
</table>

**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** and click the **Add Group** button.

### Deleting a group

Before deleting a group it is recommended that you check whether the group is being used by any permission schemes, email notification schemes, issue security levels or saved filters. See 'Viewing groups' (above).
To delete a group, click the Delete link for that group in the Group Browser. 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.

Editing group membership

To edit a group's membership, click the Edit Members link in the row for that group in the Group Browser. 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. See Granting global permissions for instructions.
- Specify the group name in the 'Default Group Memberships' option when Connecting to an LDAP Directory. See Adding Users to Groups Automatically for instructions.

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:

- JIRA 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).

  A project administrator is someone who has the project-specific 'Administer Project' permission, but not necessarily the global 'JIRA Administrator' permission.

Project roles can be used in:

- permission schemes
- email notification schemes
- issue security levels
- comment visibility
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 altered by project administrators.

**About project leads and component leads** — Every project has a project lead and every project component has a component lead. These individual roles can be used in schemes, issues and workflows, just like project roles. You assign project/component leads when Defining a Project or Defining a Component respectively.

### 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:

<table>
<thead>
<tr>
<th>Project Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrators</td>
<td>Typically contains people who administer a given project.</td>
</tr>
<tr>
<td>Developers</td>
<td>Typically contains people who work on issues in a given project.</td>
</tr>
<tr>
<td>Users</td>
<td>Typically contains people who log issues in a given project.</td>
</tr>
</tbody>
</table>

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. Choose User Management. Select Roles to open the Project Role Role Browser page.
3. You will then see the Project Role Browser, which contains a list of all the project roles in your JIRA system:

**Screenshot: 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.

5. Click any of the **View** links on the 'View Usage for Project Role' screen to see which users/groups are associated with a project role for a particular project.

**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.

1. Click on Manage Default Members in the **Operations** column for the newly created Project Role.
2. Click **Edit** under Default Users to open this dialog:

3. Select the User Picker icon to the right of the **Add user(s) to project role** field to open up this dialog:
4. Click the Select button at the bottom of this dialog when you are finished adding users and then click the Add button. You now see a list of users on the right that are now included in this Project Role.

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.

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

Before you begin: Back up your existing JIRA data.

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Choose 📚 > System. Select 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'

   ![Scheme Tools](image)

   The tools below may be used to efficiently manage existing permission schemes and notification schemes. These tools also provide the ability to update schemes to use project roles instead of groups, as project roles can be easier to manage than multiple, project-specific groups.

   **Note:** Please perform a full backup before running any of these tools.

   **Scheme Comparison Tool**
   This tool identifies the differences between several selected schemes. This is useful for identifying similar schemes, which can then be edited to make them identical. Identical schemes can then be merged using the Scheme Merge Tool described below.

   **Group to Project Role Mapping Tool**
   This tool helps you to migrate from group-based schemes to role-based schemes. It provides a quick way to bulk edit schemes such that group-based recipients (for notification schemes) and group-based permissions (for permission schemes) are replaced by project roles. Your existing schemes will be backed up.

   ![Group to Project Role Mapping Tool](image)

4. This will display the 'Map Groups to Project Roles: Select Schemes' page:
   Screenshot 2: 'Map Groups to Project Roles: Select Schemes'

   ![Map Groups to Project Roles: Select Schemes](image)

   Select the schemes for which you would like to replace groups with project roles. By default, only schemes with a project association are shown. To view all schemes, click the 'All' tab.

   **Note:** When this tool maps groups to project roles it will also unpack users into the mapped project role. This is done to preserve the membership of the resulting scheme.

   - **Step 1: Select a scheme type**
   - **Step 2: Select the schemes to work with**

   ![Map Groups to Roles](image)

   - 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'

   You may want to run the 'Scheme Merge Tool' to slim down some of your new schemes. You may also run the 'Bulk Delete Schemes Tool' to clean up your backup schemes once you are satisfied that the new schemes are working correctly.
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. Choose ‣ System. Select 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)**

   ![Scheme Comparison: View Scheme Differences](image)

   Then repeat the steps above, and verify that you have achieved a batch of 2 or more identical permission schemes, e.g.

6. 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'**

   ![Merge Schemes: Choose Schemes to Merge](image)

   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.

8. You will now see the 'Merge Schemes: Preview Results' page:

   **Screenshot 11: 'Merge Schemes: Preview Results'**

   ![Merge Schemes: Preview Results](image)

   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).

10. 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.

### Configuring User Directories in JIRA

**To configure your user directories:**

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Choose > User Management > User Directories.
   ![Keyboard shortcut]('g' + 'g' + '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.
- 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

<table>
<thead>
<tr>
<th>Directory Name</th>
<th>Type</th>
<th>Order</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>JIRA Internal Directory</td>
<td>Internal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDAP server</td>
<td>OpenLDAP (Read-Write)</td>
<td></td>
<td>Disable Edit Synchronise</td>
</tr>
</tbody>
</table>

Last synchronised at 17/01/11 10:31 AM (took 72s).

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 another directory server.

On this page:
- Settings
- Diagram of Possible Configuration

Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>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.</td>
</tr>
</tbody>
</table>

Diagram of Possible Configuration

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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.

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. Choose User Management > 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.

Notes:

- For this configuration, every time user logs in (i.e. first and subsequent times), the user's data in JIRA will be updated from the user's data in LDAP. This includes username, display name, email and group memberships. However for group memberships, only the following applies:
  - direct groups only (i.e. not nested groups) are synchronised from LDAP.
  - only groups that are already present in JIRA are synchronised, i.e. groups are not added/removed, and group hierarchies are not synchronised.

Server Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| **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  
  - opensds.example.com |
| **Port** | The port on which your directory server is listening. Examples:  
  - 389  
  - 10389  
  - 636 (for example, for SSL) |
| **Use SSL** | Check this 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**

<table>
<thead>
<tr>
<th><strong>Setting</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
</table>
| **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**

**Note:** You can only assign LDAP users to local groups when 'External Management User Management' is not selected.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Only</td>
<td>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.</td>
</tr>
<tr>
<td>Read Only, with Local Groups</td>
<td>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.</td>
</tr>
<tr>
<td>Read/Write</td>
<td>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.</td>
</tr>
</tbody>
</table>

**Adding Users to Groups Automatically**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>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.</td>
</tr>
<tr>
<td>Use Paged Results</td>
<td>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.</td>
</tr>
<tr>
<td>Configuration Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Follow Referrals</td>
<td>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.</td>
</tr>
<tr>
<td>Naive DN Matching</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>• If this check box is selected, 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.</td>
</tr>
<tr>
<td></td>
<td>• If this check box is not selected, the application will parse the DN and then check the parsed version.</td>
</tr>
<tr>
<td>Enable Incremental Synchronisation</td>
<td>Enable incremental synchronisation if you only want changes since the last synchronisation to be queried when synchronising a directory.</td>
</tr>
<tr>
<td></td>
<td>⚠️ Please be aware that when using this option, the user account configured for synchronisation must have read access to:</td>
</tr>
<tr>
<td></td>
<td>• The uSNChanged attribute of all users and groups in the directory that need to be synchronised.</td>
</tr>
<tr>
<td></td>
<td>• The objects and attributes in the Active Directory deleted objects container (see Microsoft's Knowledge Base Article No. 892806 for details).</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>This setting is only available if the directory type is set to &quot;Microsoft Active Directory&quot;.</td>
</tr>
<tr>
<td>Synchronisation Interval (minutes)</td>
<td>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.</td>
</tr>
<tr>
<td>Read Timeout (seconds)</td>
<td>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.</td>
</tr>
<tr>
<td>Setting</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Search Timeout (seconds)</td>
<td>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.</td>
</tr>
<tr>
<td>Connection Timeout (seconds)</td>
<td>This setting affects two actions. The default value is 0.</td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td>• 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</td>
</tr>
<tr>
<td></td>
<td>may be several minutes.</td>
</tr>
</tbody>
</table>

### User Schema Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Object Class</td>
<td>This is the name of the class used for the LDAP user object. Example:</td>
</tr>
<tr>
<td></td>
<td>• user</td>
</tr>
<tr>
<td>User Object Filter</td>
<td>The filter to use when searching user objects. Example:</td>
</tr>
<tr>
<td></td>
<td>• {objectCategory=Person} {sAMAccountName=*}</td>
</tr>
<tr>
<td>User Name Attribute</td>
<td>The attribute field to use when loading the username. Examples:</td>
</tr>
<tr>
<td></td>
<td>• cn</td>
</tr>
<tr>
<td></td>
<td>• sAMAccountName</td>
</tr>
<tr>
<td></td>
<td>NB: In Active Directory, the 'sAMAccountName' is the 'User Logon Name (pre-Windows 2000)' field. The User Logon Name field is referenced by 'cn'.</td>
</tr>
<tr>
<td>User Name RDN Attribute</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td>• cn</td>
</tr>
<tr>
<td>User First Name Attribute</td>
<td>The attribute field to use when loading the user's first name. Example:</td>
</tr>
<tr>
<td></td>
<td>• givenName</td>
</tr>
<tr>
<td>User Last Name Attribute</td>
<td>The attribute field to use when loading the user's last name. Example:</td>
</tr>
<tr>
<td></td>
<td>• sn</td>
</tr>
<tr>
<td>User Display Name Attribute</td>
<td>The attribute field to use when loading the user's full name. Example:</td>
</tr>
<tr>
<td></td>
<td>• displayName</td>
</tr>
<tr>
<td>User Email Attribute</td>
<td>The attribute field to use when loading the user's email address. Example:</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• mail</td>
</tr>
<tr>
<td>User Password Attribute</td>
<td>The attribute field to use when loading a user's password. Example:</td>
</tr>
<tr>
<td></td>
<td>• unicodePwd</td>
</tr>
<tr>
<td>User Unique ID Attribute <em>(JIRA only)</em></td>
<td>The attribute used as a unique immutable identifier for user objects. This is used to track username changes and is optional. If this attribute is not set (or is set to an invalid value), user renames will not be detected — they will be interpreted as a user deletion then a new user addition.</td>
</tr>
<tr>
<td></td>
<td>This should normally point to a UUID value.</td>
</tr>
<tr>
<td></td>
<td>Standards-compliant LDAP servers will implement this as 'entryUUID' according to RFC 4530. This setting exists because it is known under different names on some servers.</td>
</tr>
</tbody>
</table>

**Group Schema Settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Object Class</td>
<td>This is the name of the class used for the LDAP group object. Examples:</td>
</tr>
<tr>
<td></td>
<td>• groupOfUniqueNames</td>
</tr>
<tr>
<td></td>
<td>• group</td>
</tr>
<tr>
<td>Group Object Filter</td>
<td>The filter to use when searching group objects. Example:</td>
</tr>
<tr>
<td></td>
<td>• (&amp;(objectClass=group)(cn=*))</td>
</tr>
<tr>
<td>Group Name Attribute</td>
<td>The attribute field to use when loading the group's name. Example:</td>
</tr>
<tr>
<td></td>
<td>• cn</td>
</tr>
<tr>
<td>Group Description Attribute</td>
<td>The attribute field to use when loading the group's description. Example:</td>
</tr>
<tr>
<td></td>
<td>• description</td>
</tr>
</tbody>
</table>

**Membership Schema Settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Members Attribute</td>
<td>The attribute field to use when loading the group's members. Example:</td>
</tr>
<tr>
<td></td>
<td>• member</td>
</tr>
<tr>
<td>User Membership Attribute</td>
<td>The attribute field to use when loading the user's groups. Example:</td>
</tr>
<tr>
<td></td>
<td>• memberOf</td>
</tr>
<tr>
<td>Use the User Membership Attribute, when finding the user's group membership</td>
<td>Use the User Membership Attribute, when finding the members of a group</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Check this if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.)</td>
<td></td>
</tr>
<tr>
<td>• If this checkbox is selected, 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 retrieval.</td>
<td></td>
</tr>
<tr>
<td>• If this checkbox is not selected, your application will use the members attribute on the group ('member' by default) for the search.</td>
<td></td>
</tr>
<tr>
<td>• If the Enable Nested Groups checkbox is selected, your application will ignore the Use the User Membership Attribute option and will use the members attribute on the group for the search.</td>
<td></td>
</tr>
</tbody>
</table>

**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

There’s a Confluence SSL plugin that facilitates this process.

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.

<table>
<thead>
<tr>
<th>Required Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Information Services (IIS)</td>
<td>This is required before you can install Windows Certificate Services.</td>
</tr>
<tr>
<td>Windows Certificate Services</td>
<td>This installs a certification authority (CA) which is used to issue certificates. Step 1, below, explains this process.</td>
</tr>
<tr>
<td>Windows 2000 Service Pack 2</td>
<td>Required if you are using Windows 2000</td>
</tr>
</tbody>
</table>

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.
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.

Once the certificate has been imported as per the below instructions, you will need to restart the application to pick up the changes.

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 keystor password: changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 155636677a4e9e4582d8a84be683f9
Certificate fingerprints:
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:

```plaintext
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:
   SHA1:
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:

   ```bash
   sudo keytool -import -keystore ./jre/lib/security/cacerts -file
d   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:

```plaintext
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:
   SHA1:
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)
  - Group Schema Settings (Used when enabling Synchronise Group Memberships)
- Diagrams of Possible Configurations

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. Choose ☯ > User Management > 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.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A descriptive name that will help you to identify the directory. Examples:</td>
</tr>
<tr>
<td></td>
<td>• Internal directory with LDAP Authentication</td>
</tr>
<tr>
<td></td>
<td>• Corporate LDAP for Authentication Only</td>
</tr>
</tbody>
</table>
### 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
- opensds.example.com

### Port
The port on which your directory server is listening. Examples:
- 389
- 10389
- 636 (for example, for SSL)

### Use SSL
Select 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. It currently copies the data across whenever a user logs in, as per the bug

[JRA-27541 - Delegated LDAP copy user on first login problem](https://jira.atlassian.com/browse/JRA-27541) Resolved.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copy User on Login</td>
<td>This option affects what will happen when a user attempts to log in. If this check box is selected, the user will be created automatically in the internal directory that is using LDAP for authentication when the user first logs in and their details will be synchronised on each subsequent log in. If this check box is not selected, the user's login will fail. If you select this check box the following additional fields will appear on the screen, which are described in more detail below:</td>
</tr>
</tbody>
</table>
|                          | - Default Group Memberships  
|                          | - Synchronise Group Memberships  
|                          | - User Schema Settings (described in a separate section below) |
### Default Group Memberships

This field appears if you select the **Copy User on 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
- bamboo-users,jira-users,jira-developers

### Synchronise Group Memberships

This field appears if you select the **Copy User on Login** check box. If this check box is selected, group memberships specified on your LDAP server will be synchronised with the internal directory each time the user logs in.

If you select this check box the following additional fields will appear on the screen, both described in more detail below:

- Group Schema Settings (described in a separate section below)
- Membership Schema Settings (described in a separate section below)

### Schema Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The root distinguished name (DN) to use when running queries against the directory server. Examples:</td>
</tr>
<tr>
<td></td>
<td>- o=example,c=com</td>
</tr>
<tr>
<td></td>
<td>- cn=users,dc=ad,dc=example,dc=com</td>
</tr>
<tr>
<td></td>
<td>- 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.</td>
</tr>
<tr>
<td>User Name Attribute</td>
<td>The attribute field to use when loading the username. Examples:</td>
</tr>
<tr>
<td></td>
<td>- cn</td>
</tr>
<tr>
<td></td>
<td>- sAMAccountName</td>
</tr>
</tbody>
</table>
### 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.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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`                                                                                              |

### Group Schema Settings (Used when enabling Synchronise Group Memberships)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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:  
  - `(&(objectClass=group)(cn*))`                                                                         |
<table>
<thead>
<tr>
<th>Group Name Attribute</th>
<th>The attribute field to use when loading the group's name. Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• cn</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Group Description Attribute</th>
<th>The attribute field to use when loading the group's description. Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• description</td>
</tr>
</tbody>
</table>

**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

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.
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.

Notes:

- If you have JIRA-Crowd-LDAP, every time user logs in (i.e. first and subsequent times), the user's data in JIRA/Crowd will be updated from the user's data in LDAP. This includes username, display name, email and group memberships. However for group memberships, only the following applies:
  - direct groups only (i.e. not nested groups) are synchronised from LDAP.
  - only groups that are already present in JIRA are synchronised, i.e. groups are not added/removed, and group hierarchies are not synchronised.

Settings in JIRA for the Crowd Directory Type

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A meaningful name that will help you to identify this Crowd server amongst your list of directory servers. Examples:</td>
</tr>
<tr>
<td></td>
<td>- Crowd Server</td>
</tr>
<tr>
<td></td>
<td>- Example Company Crowd</td>
</tr>
<tr>
<td>Server URL</td>
<td>The web address of your Crowd console server. Examples:</td>
</tr>
<tr>
<td></td>
<td>- <a href="http://www.example.com:8095/crowd/">http://www.example.com:8095/crowd/</a></td>
</tr>
<tr>
<td></td>
<td>- <a href="http://crowd.example.com">http://crowd.example.com</a></td>
</tr>
<tr>
<td>Application Name</td>
<td>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.</td>
</tr>
<tr>
<td>Application Password</td>
<td>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.</td>
</tr>
</tbody>
</table>

Crowd Permissions

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
### Advanced Crowd Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>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.</td>
</tr>
<tr>
<td>Synchronisation Interval (minutes)</td>
<td>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.</td>
</tr>
</tbody>
</table>

---

### 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.
   - 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.
   - Choose 🏛️ > User Management > 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

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Only</td>
<td>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.</td>
</tr>
</tbody>
</table>

### Advanced Settings for the JIRA Directory Type

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>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.</td>
</tr>
</tbody>
</table>
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.
Avoid duplicate usernames across directories. If you are connecting to more than one user directory, we recommend that you ensure the usernames are unique to one directory. For example, we do not recommend that you have a user jsmith in both 'Directory1' and 'Directory2'. The reason is the potential for confusion, especially if you swap the order of the directories. Changing the directory order can change the user that a given username refers to.

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.

### 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.

<table>
<thead>
<tr>
<th>Directory Name</th>
<th>Type</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>JIRA Internal Directory</td>
<td>Internal</td>
<td>🔄 🔄</td>
</tr>
<tr>
<td>LDAP server</td>
<td>OpenLDAP (Read-Write)</td>
<td>🔄 🔄</td>
</tr>
</tbody>
</table>

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.
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 external directory.

On this page:
- Affected Directory Types
- How it Works
- Finding the Time Taken to Synchronise
- Manually Synchronising the Cache
- Configuring the Synchronisation Interval

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 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.

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

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

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*
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

### JIRA Internal Directory

![Diagram of JIRA Internal Directory](image)

*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.
General Recommendations

- **Avoid duplicate usernames across directories.** If you are connecting to more than one user directory, we recommend that you ensure the usernames are unique to one directory. For example, we do not recommend that you have a user `jsmith` in both 'Directory1' and 'Directory2'. The reason is the potential for confusion, especially if you swap the order of the directories. Changing the directory order can change the user that a given username refers to.

- **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.

<table>
<thead>
<tr>
<th>Your environment</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up to 10 000 (ten thousand) users, 1000 (one thousand) groups, and 20 (twenty) groups per user</td>
<td>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.</td>
</tr>
<tr>
<td>More than the above</td>
<td>Use LDAP filters to reduce the number of users and groups visible to the synchronisation task.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Your environment</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>If all the following are true:</td>
<td></td>
</tr>
<tr>
<td>• You have fewer than 500 users.</td>
<td></td>
</tr>
<tr>
<td>• You want to share user and group management across just a few applications,</td>
<td></td>
</tr>
<tr>
<td>such as one JIRA server and one Confluence server, or two JIRA servers.</td>
<td></td>
</tr>
<tr>
<td>• You do not need single sign-on (SSO) between JIRA and Confluence, or between</td>
<td></td>
</tr>
<tr>
<td>two JIRA servers.</td>
<td></td>
</tr>
<tr>
<td>• You do not have custom application connectors. Or, if you do have them,</td>
<td></td>
</tr>
<tr>
<td>you are happy to convert them to use the new REST API.</td>
<td></td>
</tr>
<tr>
<td>• You are happy to shut down all your servers when you need to upgrade JIRA.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Your environment meets the optimal requirements for using JIRA for user management.</td>
</tr>
<tr>
<td>If one or more of the following are true:</td>
<td>We recommend that you install Atlassian Crowd for user management and SSO.</td>
</tr>
<tr>
<td>• You have more than 500 users.</td>
<td></td>
</tr>
<tr>
<td>• You want to share user and group management across more than 5 applications.</td>
<td></td>
</tr>
<tr>
<td>• You need single sign-on (SSO) across multiple applications.</td>
<td></td>
</tr>
<tr>
<td>• You have custom applications integrated via the Crowd SOAP API, and you cannot</td>
<td></td>
</tr>
<tr>
<td>convert them to use the new REST API.</td>
<td></td>
</tr>
<tr>
<td>• You are not happy to shut down all your servers when you need to upgrade JIRA.</td>
<td></td>
</tr>
</tbody>
</table>
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

---

**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. Choose <icon> > **User Management**. Select **JIRA User Server**.
   c. **Keyboard shortcut**: 'g' + 'g' + start typing 'jira user'.
   d. **Add** an application.
   e. Enter the **application name** and **password** that the application will use when accessing your JIRA server.
   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.

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. Choose System > Security > User Sessions to open the Current User Sessions in JIRA.

Keyboard shortcut: 'g' + 'g' + start typing 'user sessions'

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.
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:

1. Choose 

2. 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:

```properties
log4j.additivity.com.atlassian.jira.web.filters = false
```

to:

```properties
log4j.category.com.atlassian.jira.web.filters.AccessLogFilter = INFO, console, filelog
log4j.additivity.com.atlassian.jira.web.filters = false
```
Usage in JIRA 4.x

URL logging is disabled in JIRA by default. To turn it on:

1. Choose 🛠️ > System. Select Troubleshooting and Support > Logging & Profiling to open the Logging page, which lists all defined log4j categories (as package names) and their current logging levels.
2. 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>`:

```xml
<Valve className="org.apache.catalina.valves.AccessLogValve" pattern="%h %l %u %t %r %s %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:

```log
log4j.category.com.atlassian.jira.web.filters.AccessLogFilter = INFO, console, filelog
log4j.additivity.com.atlassian.jira.web.filters = false
```

and then restart JIRA.
Password Policy for JIRA

Overview

The JIRA password policy enables JIRA Administrators to set limits and restrictions on the types of passwords their users can create. You can use this feature to create a more secure JIRA system for your company.

Note: The JIRA password policy is disabled by default. To turn it on and configure it, follow the instructions below. Also, this policy is only useful when JIRA users can change their own passwords. If JIRA is connected to an Active Directory, this policy should not be used.

Enabling the password policy

1. Log in as a user with the JIRA Administrators global permission.

   Choose the cog icon at top right of the screen, then choose User Management. Next, select Password Policy on the left.
2. Select one of the following options:
   a. **Disabled** – The equivalent of having no password policy.
   b. **Basic** – Requires passwords to be at least 8 characters long and use at least 2 character types. Rejects passwords that are very similar to the previous password or the user's public information.
   c. **Secure** – Requires passwords to be at least 10 characters long and use at least 3 character types including at least 1 special character. Rejects passwords that are even slightly similar to the previous password or the user's public information.
   d. **Custom** – Lets you use your own settings.
3. Configure the following fields:
   a. **Password Length** – Set a minimum and maximum length for your passwords. Currently, you must set a maximum length if you enable the password policy and the maximum value allowed is 255.
   b. **Character Variety** – Use these fields to set requirements around the types of characters – uppercase letters, lowercase letters, special characters, and so on – that are required.
   c. **Similarity Checks** – See the section below for details on this feature.
4. Click the **Update** button at the bottom of the screen when you are ready.

**Similarity Checks**
This is a system check to make sure that your users aren't creating a new password that is too similar to the current password, the user's name or email address. It can be set to **Ignored**, **Lenient**, or **Strict**.
What is the difference between lenient and strict?

- **Lenient** checks for obvious similarities, like reversing the username or moving the front letter to the end.
- **Strict** checks for more subtle variations, like mixing up the letters or adding just one new character. It also performs a character frequency analysis.

Password FAQ

**Question:** Why would you ever want a maximum password length?

**Answer:** Maybe you shouldn't, but you may want to do this for security or other reasons. For example, if you are using a writable external user directory, then that external directory may have its own restrictions on the maximum password length that it allows.

**Question:** What is Character Variety and why should I use this?

**Answer:** Character variety refers to the different types of characters you can create on a keyboard: lowercase letters, uppercase letters, numbers, and special characters. Requiring different character types makes passwords harder to guess, but it might also make them harder to remember. Use your best judgment when setting these fields, keeping in mind your company's requirements as well as your user base.

**Question:** Does this policy affect existing passwords?

**Answer:** The policy is only enforced as passwords are changed; there is no way to detect whether or not existing passwords satisfy the policy or to force the users to update their passwords if the policy has been changed. As a workaround, you can use this Crowd REST resource to forcibly change the users' passwords to something they won't know, thereby requiring them to reset it to get back in, and the password reset enforces the policy rules.

**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

ℹ️ If you are a JIRA administrator and wish to disable this feature, see 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]

   When you log in and select 'Remember my login...', JIRA remembers this using a token so you do not have to enter your login details again from the same browser.

   Click Clear All Tokens to remove them.

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. Choose ☰ > User Management.
    
   ✔ 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]
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. Choose System > System. Select Security > Remember My Login to open the Remember My Login for All Users page.

   Keyboard shortcut: 'g' + 'g' + start typing 'remember my login'

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
Enabling Public Signup and CAPTCHA

About Public Signup

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.

Enabling Public Signup

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Choose System. Select General Configuration to open the Administration page.
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. Choose System. Select General Configuration to open the Administration page.
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:

![SignUpForm]

### Project Management
- Defining a Project
  - Editing a Project Key
  - Changing the Project Key Format
  - Simple Issue Tracking project
  - Software Development project
- Managing Project Role Membership
- Defining a Component
- Managing Versions
  - Running a Bamboo Build when Releasing a Version
  - Creating Release Notes

### Defining a Project
This page tells you how to add a new project or configure an existing project, if you are a JIRA administrator. You’ll need to navigate to the project administration screen for this:

Choose **Projects.**

- To create a new project, choose the **Add Project** button and follow the wizard. See Creating a project below for details.
- To configure an existing project, choose the project name from the **Project list** and configure the settings. See Configuring a project below for details.
Creating a project

To create a new project in JIRA:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Projects > Create Project.
3. Choose the type of project that you want to create.
4. Enter the project details. Note, choose the project key carefully, as it is not a simple task to change it after the project is created.
5. Choose Submit to add the new project.

Notes:

- Choosing JIRA Classic or Project Management creates the default JIRA project.
- The project key will be used as the prefix of this project’s issue keys (e.g. ‘TEST-100’). Choose one that is descriptive and easy to type.
- The project lead is a unique project role. Choose the person who manages the project as the project lead. 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.

Simple Issue Tracking project

This project provides you with a quick and easy way to get JIRA up and running for simple issue tracking. For details on working with this project, see Simple Issue Tracking project.

Software Development project
This project provides you with a template to use for software development. For details on working with this project, see Software Development project.

Configuring a project

To configure a project in JIRA:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > Projects. ✅ Keyboard shortcut g + g and then start typing the name of your project.
3. Use the tabs on the left to navigate between the different project settings. Read the sections below for a description of each setting.
   - **Project details**
   - **Issue types**
   - **Workflows**
   - **Screens**
   - **Fields**
   - **Settings**
   - **Roles**
   - **Versions**
   - **Components**
   - **Permissions**
   - **Notifications**
   - **Development tools**

Project details

To edit the project’s details:

1. Click Edit Project at the top of the Project Summary page.
2. In the resulting Edit Project dialog box, edit the following fields:
   - **Name** — type a descriptive name. This can be changed later if you wish.
   - **Key** — Click edit key to change the project key. Read this page before you edit the project key: Editing a Project Key.
   - **URL** — an optional URL associated with this project, e.g. 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. The process for choosing a project avatar is similar to that for choosing a user avatar. If you prefer not to use an image for your project, 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.

To edit the project Category, enter the project Administration section by clicking a project name and then click the link next to the **Category** field (located under the project name). JIRA can search for all the issues in a particular project category, and can display projects sorted by the project category, but a project category is not part of a project hierarchy. JIRA does not support sub-projects or parent projects. In addition a JIRA project can only belong to one category.

If no categories exist, click the **Add** link on the following **No Project Category** 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 behavior: each field can be required/optional, rich text/plain text, hidden/visible. You define this behavior 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

- **Application Links** (Configure Project Links) — if you have linked your JIRA instance to other Atlassian
applications, like Confluence, FishEye or other JIRA instances, you will be able to link this JIRA project to areas of those applications that contain information relating to your project or team. For example, Confluence spaces, FishEye repositories, JIRA projects (in another JIRA instance), etc. This allows you to take advantage of integration points between these applications. See Linking to Another Application for information about application links and project links.

Roles
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.

By default, new projects also have their 'Default Assignee' set to 'Unassigned.' You can change this here if you want to set it to a specific role, i.e. 'Project Lead.'
- **Project Roles** — members are users/groups who fulfil particular functions for this project. Project roles ar e 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.

⚠ Please note, the Default Notification Scheme (shipped with JIRA) is associated with all new projects by default. This means that if you have an outgoing (SMTP) mail server set up, that email notifications will be sent as soon as there is any activity (e.g. issues created) in the new project.

Development tools

The Development tools section gives you an overview of the development tools that are connected and which users can use the integration features between them:

• View permission - This section lists which users can see the development tools integration features (like the Create Branch link) on the view issue screen, as well as other development-related information, like commits, reviews and build information. This ability is controlled by the “View Development Tools” project permission.
• Applications - This section shows which development tools are connected to JIRA via application links and are eligible to use the development tool features in JIRA.

A note about project administrators

A project administrator in JIRA is someone who has the project-specific Administer Projects project permission, but not necessarily the JIRA Administrator global permission.

Without the JIRA Administrator global permission, however, project administrators can do the following:

• Edit the project name
• Edit the project description
• Edit the project avatar image
• Edit the project URL
• Edit the project lead
• Edit project role membership
• Define project components
• Define project versions
• View, but not select nor edit the project's schemes (notification scheme, permission scheme, etc)

Editing a Project Key

Editing a project key is not a trivial task. You should choose key that will suit your long-term needs when creating a project, rather than rely on editing the project key after the project is created. However, there are situations where you need to change the key for an existing project, e.g. change of product name.

The instructions on this page show you how to change the project key and describe the implications of such a change. If you are a plugin developer, you should also read the developer notes below.

Before you begin

• Your desired project key must confirm to the project key format restrictions specified in JIRA. By default, the project key format must be at least 2 characters long and contain only uppercase letters. You can change the project key format to enforce different restrictions. See Changing the Project Key Format for instructions.
• Perform this change during a low usage period — JIRA will start a background re-index when you save your updated project key. This can have a performance impact on your instance. Note, you cannot choose a 'Lock JIRA and rebuild index'. The background index will be faster anyway, as it is limited to issues for the project.
• Communicate changes to your users — Ensure that you are aware of the consequences of changing the project key, and have adequately prepared your users for the changes. See the Changes section below.

Related topics

• Before you begin
• Editing the project key
• Notes for change management
• Related topics
• Notes for developers
Editing the project key

1. Navigate to the desired project in JIRA.
2. Choose Administration (tab) > Edit Project (button).
3. Choose edit key next to the Key field.
4. Update the key and choose Update.

![Edit Project: Sample Scrum Project](image)

Note:

- If you updated other fields on the 'Edit Project' page as well, you will see the changes after you save your changes. You won't need to wait for the re-index to finish.
- Don't cancel the background re-index. If you cancel it, then you will have problems searching for issues related to the project. If you do need to cancel it, you can run it again later to fix these problems.

Post-update tasks

- Fix the project entity links — When you connected JIRA to another Atlassian application, entity links would have been automatically created between your JIRA projects and the relevant "projects" in other applications, e.g. FishEye/Crucible projects. If you change the key of a JIRA project, you will need to fix the project entity links as described on Creating Links Between Projects.

Notes for change management

While editing the project key is a major change, in most cases, your JIRA project will work as you'd expect with a new key. There are a few cases that you should be aware of, which are listed below. We recommend reviewing these and advising your users accordingly.

- The old project key can be used in JQL queries — Users won't have to update issue filters that reference the old project key.
- If you use Confluence with JIRA, the JIRA issue macros in Confluence will continue to work. Please note, if you don't see the change straight away, allow some time for the cache to refresh.
- You won't be able to create a new project with the old project key. However, you can change the renamed project back to the old project key. If you delete the project, all associated keys will be freed and you'll be able to re-use them.
- Links will work, whether they are inside JIRA or from external sources. However, link aliases will not be updated — For example, if you have a link to an issue ‘EXAMPLE-1’ in the description of an issue, and you change the project key ‘EXAMPLE’ to ‘DEMO’, then the alias ‘EXAMPLE-1’ will not be updated to ‘DEMO-1’. The link will still direct you to DEMO-1 though.
- If you are using the Activity Stream gadget with a global filter, you will need to update the filter after the project is renamed.
- All attachments will be accessible after the project key change. Please note however, that the directory that they are stored in (under the `<JIRA Home>\data\attachments` directory) will be retain the old project key. For example, if you change a project's key from TEST to DEMO, the attachments will be stored under `<JIRA Home>\data\attachments\TEST`.
- If you export a renamed project, then import it, it will have the updated project key, i.e. the original project key will not be retained. In fact, all historical keys for that project will be removed.

Related topics

Changing the maximum project key length — You can change the maximum characters allowed for a project
key. Navigate to the General Configuration page of the JIRA administration console, as described on Configuring JIRA Options, and change the Maximum project key size field.

Changing the project key format — You can change the format of a project key. This restricts the format of a project key when it is created or edited (as described above). For instructions, see Changing the Project Key Format.

Notes for developers

- REST API calls will still work with old project key — REST calls that specify an issue key will work with the old issue key after the project key has changed. For example, /rest/api/issue/EXAMPLE-100 will still work after the project key is changed from EXAMPLE to DEMO.
- We have created a new event, ProjectUpdatedEvent. This event is triggered any time a project's details are changed, including changing the project key.
- If you need to retrieve all issue keys and project keys (historical and current), you can do this via the following:
  - REST:
    - Get all project keys for a project: /rest/api/2/project/<project key>?expand=projectKeys
  - Java API:
    - Get all project keys: com.atlassian.jira.project.ProjectManager#getAllProjectKeys
    - Get all issue keys for an issue: com.atlassian.jira.issue.IssueManager#getAllIssueKeys

Changing the Project Key Format

JIRA provides the ability to specify the format of project keys within the system. This allows you to restrict the format of a project key, when a project key is created or edited.

A project key format is defined via a regular expression 'rule' that governs the valid project key format. By default, the JIRA project key configuration requires two or more uppercase alphabetical characters — based on the regular expression ([A-Z][A-Z]+).

On this page:
- Before you begin
- Configuring the project key format
- Related topics

Before you begin

- Ensure that you choose a supported project key format. Only formats that meet all of the following rules are supported:
  - The first character must be a letter,
  - All letters used in the project key must be from the Modern Roman Alphabet and upper case, and
  - Only letters, numbers or the underscore character can be used.
  Examples:
    - Examples of supported keys: PRODUCT_2013, R2D2, MY_EXAMPLE_PROJECT.
    - Examples of unsupported keys: 2013PROJECT (first character is not a letter), PRODUCT-2012 (hyphens are not supported).
- You cannot configure the issue key pattern, as JIRA expects this key to conform to specific rules. By default, JIRA issue keys (or issue IDs) are of the format <project key>-<issue number>, e.g. ABC-123. For example, you can't show the issue number before the project key.
- If a number of issues have already been created in your JIRA installation, then changing the project key format 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 (e.g. use a regular expression which will still be valid for existing project keys defined in your JIRA installation).
- If you use JIRA Agile and you are using the Classic Boards, do not change JIRA's default project key format as the Classic Boards only support this key format. If you are not using the Classic Boards, you can change to a supported project key format.
- If you have integrated JIRA with Bamboo, do not change JIRA's default project key format as Bamboo only supports this key format.
Configuring the project key format

The `jira.projectkey.pattern` property allows JIRA administrators to specify a Perl5 regular expression value that defines the rule for a valid project key. Further information on Perl5 is available [here](#).

This property and its regular expression value can be defined through the Advanced Settings page. This is described below.

**Step 1. Configure a pattern for your project key syntax**

1. Navigate to the JIRA Advance settings page, as described on Configuring Advanced Settings.
2. Find the `jira.projectkey.pattern` property and click its value to modify it. Below is a list of common examples and patterns:

<table>
<thead>
<tr>
<th>Pattern Requested</th>
<th>Expression needed</th>
<th>Resulting Issue IDs</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXYY, where X indicates two fixed letters, Y represents two fixed digits</td>
<td><code>([A-Z]{2}[0-9]{2})</code></td>
<td>TQ09-01, TQ09-02, etc.</td>
<td><strong>[A-Z]</strong> Any character from A to Z &lt;br&gt; {2} Matches the preceding character 2 times exactly&lt;br&gt; <strong>[0-9]</strong> Any character (i.e. digit) from 0 to 9</td>
</tr>
<tr>
<td>XZ+, where X indicates one fixed letter, Z+ represents one or more letters, digits or underscore characters</td>
<td><code>([A-Z][A-Z_0-9]+)</code></td>
<td>ACAT_51-1, AAA5-1330, A_20_A091-15, etc.</td>
<td><strong>[A-Z]</strong> Any characters from A to Z &lt;br&gt; <strong>[A-Z_0-9]</strong> Any character from A to Z, 0 to 9 or the underscore character. &lt;br&gt; + specifies one or more characters from A to Z, 0 to 9 or the underscore character.</td>
</tr>
</tbody>
</table>

Please Note:

- JIRA prepends the regular expression specified with `'` and closes it with `'$' for an exact matching rule within the system.
- The project key only supports uppercase characters, as stated above. Hence, for simplicity, use uppercase characters in your expressions as JIRA will convert any lowercase characters to uppercase ones.

**Step 2. Test 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.

(Optional) **Step 3. Customize the project key description and warning**

In addition to the project key format, you can also customize 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` — 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 this error message, so that when a user keys in the wrong format, they will be informed of the correct pattern to use.

Related topics:

- **Changing the maximum project key length** — You can change the maximum characters allowed for a project key. Navigate to the General Configuration page of the JIRA administration console, as described...
on Configuring JIRA Options, and change the **Maximum project key size** field.

- Editing a Project Key
- Defining a Project
- Configuring Advanced Settings

### Simple Issue Tracking project

This project provides you with a quick and easy way to get JIRA up and running for simple issue tracking.

This project is aimed at users who want to track their issues through a simple three-step workflow, while taking advantage of JIRA's other powerful features. Later on, it's easy to extend to other, more complex workflows provided by JIRA.

**On this page:**
- Getting started
- Details of this project template

**Related pages:**
- Defining a Project
- Migrating from Other Issue Trackers

### Getting started

To create a Simple Issue Tracking project in JIRA, follow these instructions and be sure to select **Simple Issue Tracking** in Step 3.

1. Log in as a user with the JIRA Administrators global permission.
2. Choose **Projects > Create Project**.
3. Choose the **type of project** that you want to create.
4. Enter the **project details**. Note, choose the project key carefully, as it is not a simple task to change it after the project is created.
5. Choose **Submit** to add the new project.

### Details of this project template

**Issue Types**

You get the following three issue types:

- Parent types: **New Feature**, **Task**
- Sub-task type: **Sub-Task**

JIRA auto-creates an issue type scheme called `<project key>`: **Simple Issue Tracking issue type scheme**.

### Simple Issue project workflow

This is what the project workflow looks like:
Workflow description

The Simple Issue project workflow operates as follows:

- Three statuses: **To Do, In Progress** and **Done**
- Uses the resolution: **Done**

JIRA auto-creates a workflow scheme called `<project key>: Simple Issue Tracking workflow scheme`.

Workflow transitions

Transitions have the following properties:

- When the 'Start Progress' transition is used the issue gets **assigned** to the **current user**.
- When the 'Reopen and Start Progress' transition is used the issue gets **assigned** to the **current user**.
- When any 'Done' transition occurs the **resolution** is set to **Done**.
- When the 'Reopen' transition occurs the **resolution** field is **cleared**.
- There are no transition screens.
- No Assignee transition conditions (apart from the permission scheme).

Software Development project

This project provides you with a template to use for software development. It includes a simple workflow with statuses representing tasks that go through a development and review process. You can later customize this further to suit your team's evolving processes.

Getting started

To create a Software Development project in JIRA, follow these instructions and be sure to select **Software Development** in Step 3.

1. Log in as a user with the JIRA Administrators **global permission**.
2. Choose **Projects > Create Project**.
3. Choose the **type of project** that you want to create.
4. Enter the **project details**. Note, choose the project key carefully, as it is not a simple task to change it after the project is created.
5. Choose **Submit** to add the new project.
Details of this project template

Issue Types

You get the following issue types:

- Parent types: **Story**, **Task**, **New Feature** and **Bug**
- Sub-task types: **Sub-Task**

JIRA auto-creates an issue type scheme called `<project key>: Software Development issue type scheme`.

Software Development project workflow

This is what the project workflow looks like:

![Workflow diagram]

**Workflow description**

The Software Development project workflow operates as follows:

- Four statuses: **To Do**, **In Progress**, **In Review** and **Done**
- Uses the resolution: **Done**
- **All** indicates that you can transition from any status to the associated status

JIRA auto-creates a workflow scheme called `<project key>: Software Development workflow scheme`.

**Workflow transitions**

Transitions have the following properties:

- When the ‘Start Progress’ or ‘Start Review’ transition is used the issue gets **assigned** to the **current user**
When the 'Reopen and Start Progress' or 'Reopen and start review' transition is used the issue gets assigned to the current user and the resolution field is cleared.

When any 'Done' transition occurs the resolution is set to Done.

When the 'Reopen' transition occurs the resolution field is cleared.

There are no transition screens.

No Assignee transition conditions (apart from the permission scheme).

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.

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

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.
2. Choose 🏛️ > Projects. The 'Project Summary' page (see Defining a Project) for your selected project is shown. 
   Keyboard shortcut: g + g + start typing project
3. Choose Roles in the left menu. The Roles page is displayed, from where you can manage the project role membership as described below.

Assigning a user to a project role

1. Open the Roles page as described in 'Viewing project role members' above.
2. Hover over the Users column for the project role in which you are interested and click the yellow box which appears. The users and groups will become editable.
3. Type the name of the user(s) you wish to add to this project role. 
   The Browse Users global permission is required to search for users in the Users column. If you do not have this permission, you will need to specify the exact name of the user(s), followed by the 'Enter' key after each user specified.
4. Click the Update button.

Removing a user from a project role

1. Open the Roles page, as described in 'Viewing project role members' above.
2. Hover over the Users column for the project role in which you are interested and click the yellow box which appears. The users and groups will become editable.
3. Click the x next to the name of the user(s) you wish to remove from this project role.
4. Click the Update button.

Assigning a group to a project role

1. Open the Roles screen, as described in 'Viewing project role members' above.
2. Hover over the Groups column for the project role in which you are interested and click the yellow box which appears. The users and groups will become editable.
3. Type the name of the group(s) you wish to add to this project role.
   - The Browse Users global permission is required to search for groups in the Groups column. If you do not have this permission, you will need to specify the exact name of the group(s), followed by the 'Enter' key after each group specified.
4. Click the Update button.

Since group membership can only be edited by users 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 Roles page, as described in 'Viewing project role members' above.
2. Hover over the Groups column for the project role in which you are interested and click the yellow box which appears. The users and groups will become editable.
3. Click the x next to the name of the group(s) you wish to remove from this project role.
4. 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.

On this page:
- Managing a project's components
- Adding a new component
- Editing a component's details
- Deleting a component

Managing a project's components

1. Log in to JIRA as a project administrator.
2. Choose 🌐 > Projects. The Project Summary page is displayed (see Defining a Project).
   - Keyboard shortcut: g + g + start typing project
3. Choose Components in the left menu. The Components page is displayed, showing a list of components and each component's details. From here you can manage the project's components as described below.

Screenshot: The 'Components' screen
Adding a new component

1. The Add Component form is located at the top of the 'Components' screen.
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.

<table>
<thead>
<tr>
<th>Default Assignee Option</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Default</td>
<td>Issues matching this component will have the assignee set to the same default assignee as the parent project.</td>
<td></td>
</tr>
<tr>
<td>Project Lead</td>
<td>The assignee will be set to the project leader.</td>
<td>If the project leader is not permitted to be assigned to issues in the permission scheme this option will be disabled and will say &quot;Project Lead is not allowed to be assigned issues&quot;.</td>
</tr>
<tr>
<td>Component Lead</td>
<td>The assignee will be set to the component leader.</td>
<td>If the component leader is not permitted to be assigned to issues in the permission scheme this option will be disabled and will say &quot;Component Lead is not allowed to be assigned issues&quot;. 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 &quot;Component does not have a lead.&quot;.</td>
</tr>
<tr>
<td>Unassigned</td>
<td>The assignee of the issue will not be set on the creation of this issue.</td>
<td>This option will only be available if &quot;Allow unassigned issues&quot; is enabled in the General Configuration.</td>
</tr>
</tbody>
</table>

Editing a component's details
1. On the 'Components' screen, hover over the relevant component to display the pencil icon.
2. 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, 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 organize 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** and **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.

---

Managing a project's versions

1. Log in to JIRA as a **project administrator**.
2. Choose ☑️ > **Projects**. The **Project Summary** page is displayed (see Defining a Project).
   
   **Keyboard shortcut:** g + g + start typing **project**
3. Choose **Versions** in the left menu. The **Versions** page is displayed, showing a list of versions and each version's status. From here you can manage the project's versions as described below.

   **Screenshot:** 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').

Add a new version

1. The Add Version form is located at the top of the 'Versions' screen.
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. You can drag the new version to a different position by hovering over the 'drag' icon at the left of the version name.

Add a start date

If specified, the **Start Date** is used by the Version Report. This gives you a more accurate report in cases where you might plan a version many weeks (or even months) in advance, but not actually commence work until closer to the release date.

Release 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 in JIRA. The 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, 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.

Archive a version
1. On the 'Versions' screen, 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.

Merge multiple versions

Merging multiple versions allows you to move the issues from one or more versions to another version.

1. On the 'Versions' screen, 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.

Edit a version's details

1. On the 'Versions' screen, 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.

Delete a version

1. On the 'Versions' screen, 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.

Reschedule a version

Rescheduling a version changes its place in the order of versions.

• On the 'Versions' screen, click the icon for the relevant version, and drag it to its new position in the version order.

See also

• Setting Up a Version Hierarchy
• Viewing the Version Report

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.

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 > the desired project > the desired version > Release.
   - **Warning**: 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 dialog will be displayed.
4. Enter the build details for the release:

<table>
<thead>
<tr>
<th>'No Build'</th>
<th>Choose this option if you do not want to run a Bamboo build, i.e. you only want to release the version in JIRA.</th>
</tr>
</thead>
</table>
| 'Release new Build'    | Choose this option, if you want to run a Bamboo build that has not been started:  
    * 'Using Plan' — You can select any Plan in the linked Bamboo instance that you have permission to view (unless your administrator has configured basic HTTP authentication).  
    * 'Stages' — You can select the Stages that you want to run for this release. Note, you cannot skip Stages.  
    * 'Build Variables' — You can override any global variables or plan variables with your own parameters. See Running 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 can select any Plan in the linked Bamboo instance that you have permission to view (unless your administrator has configured basic HTTP authentication).  
    * 'Stages' — You can 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 override any global variables or plan variables with your own parameters. See Running 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.

### 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. Choose ![Projects](Projects) on the left of the screen.
2. Click **Overview > Road Map** on the left of the screen.
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`

   **Notes:**
   a. Corresponding entries in both of these properties must be in the same order.
   b. If these properties do not exist in your `jira-config.properties` file, then:
      i. For each of these properties, add the property's name,
      ii. followed by an `='`,
      iii. followed by the content of the property's corresponding `<default-value/>` element copied from your JIRA installation's `jpm.xml` file.
      iv. 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 Security**

When configuring security for your JIRA instance, there are two areas to address:

- permissions within JIRA itself
- security in the external environment

**Configuring permissions 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 permissions 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
- JIRA Admin Helper

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.)

- **SSL** — if you are running your JIRA instance over the Internet, you may want to consider using SSL.

- **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
Other security resources
Security Addendum 2010-04-16 - Preventing security attacks

- JIRA Cookies
- Securing JIRA with Apache HTTP Server
- User and Group Management
- How to Report a Security Issue
- Severity Levels for Security Issues
- Security Patch Policy
- Security Advisory Publishing Policy
- Configuring Security
- Configuring project specific security
- Tomcat security best practices
- Security Advisories

**Configuring Issue-level Security**

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 'Reported', '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:**
- Why use issue security levels?
- Creating an issue security scheme
- Assigning an issue security scheme to a project
- Deleting an issue security scheme
- Copying an issue security scheme

**Related topics:**
- Setting Security on an Issue

**Why use issue security levels?**

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. Choose ➞ Issues. Select **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

   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.
4. 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. Choose ➞ Issues. Select **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 open the **Edit Issue Security Levels** page.
4. In the **Add Security Level** box, enter a name and description for your new security level and 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. Choose ⚙ > Issues. Select 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 to open 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 from 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. Choose ⚙ > Issues. Select 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 to open the Edit Issue Security Levels page (above).

4. Locate the appropriate security level and click its Add link (in the Operations column), which opens the Add User/Group/Project Role to Issue Security Level page.

5. Select the appropriate user, group or project role, then click the Add button.

6. Repeat steps 4 and 5 until all appropriate users and/or groups and/or project roles have been added to the security level.

Assigning an issue security scheme to a project

1. Log in as a user with the JIRA Administrators global permission.

2. Choose ⚙ > Projects. Select the name of the project of interest. The Project Summary page is displayed.
3. In the Permissions section of the Project Summary page, click the link corresponding to the Issues label to open the Associate Issue Security Scheme to Project page.

   This will either be the name of the project's current issue security scheme, or the word None.

4. Select the issue security scheme that you want to associate with this project.

5. If there are no previously secured issues (or if the project did not previously have an issue security scheme), skip the next step.

6. 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.

7. 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. Choose Issues. Select 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 Delete link (in the Operations column) for the scheme that you want to delete.

   You cannot delete an issue security scheme if it is associated with a project. To do so, you must first remove any associations between the issue security scheme and projects in your JIRA installation — 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. Choose Issues. Select 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 Copy link (in the Operations column) for the scheme that you want to copy. 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.

This table lists the different global permissions and the functions they secure:

<table>
<thead>
<tr>
<th>Global Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>JIRA System Administrators</td>
<td>Permission to perform all JIRA administration functions.</td>
</tr>
<tr>
<td></td>
<td>This does not include the JIRA Users permission.</td>
</tr>
<tr>
<td></td>
<td>A user with JIRA System Administrators will be able to log in to JIRA</td>
</tr>
<tr>
<td></td>
<td>without the JIRA Users permission, but may not be able to perform all</td>
</tr>
<tr>
<td></td>
<td>regular user functions (e.g. edit their profile) unless they also belong</td>
</tr>
<tr>
<td></td>
<td>to a group that has the JIRA Users permission.</td>
</tr>
<tr>
<td>JIRA Administrators</td>
<td>Permission to perform most JIRA administration functions (see list of</td>
</tr>
<tr>
<td></td>
<td>exclusions below).</td>
</tr>
<tr>
<td></td>
<td>This does not include the JIRA Users permission.</td>
</tr>
<tr>
<td></td>
<td>A user with JIRA Administrators will be able to log in to JIRA without the</td>
</tr>
<tr>
<td></td>
<td>JIRA Users permission, but may not be able to perform all regular user</td>
</tr>
<tr>
<td></td>
<td>functions (e.g. edit their profile) unless they also belong to a group</td>
</tr>
<tr>
<td></td>
<td>that has the JIRA Users permission.</td>
</tr>
</tbody>
</table>
### 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](#).

ℹ️ Granting the JIRA Users permission to a group results in all newly created users being automatically added to that group. The exception to this are groups that also have either the JIRA System Administrators or JIRA Administrators permission, since JIRA prevents groups with these administrative-level global permissions from being granted the JIRA Users permission. Furthermore, it would be unwise to automatically grant these administrative-level global permissions 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. Enables auto-completion of user names in most 'User Picker' menus and popups.

Note that the Assign User permissions also allows a limited version of this on a per-project basis.

### Create Shared Objects

Permission to share a filter or dashboard globally or with groups of users. Also used to control who can create a JIRA Agile board.

### 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).

## Managing Project Permissions

Project permissions are created within Permission Schemes, which are then assigned to specific projects.

Project permissions are able to be granted based on:

- 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.

<table>
<thead>
<tr>
<th>Permission Schemes</th>
<th>Creating a Permission Scheme</th>
<th>Associating a Permission Scheme with a Project</th>
<th>Deleting a Permission Scheme</th>
<th>Copying a Permission Scheme</th>
<th>Global Permissions</th>
</tr>
</thead>
</table>

### Project Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administer Projects</td>
<td>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').</td>
</tr>
<tr>
<td>Browse Projects</td>
<td>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.</td>
</tr>
<tr>
<td>View Development Tools</td>
<td>Permission to view the related source code commits (e.g. CVS, Subversion, FishEye, etc) for an issue, in a 'Source' tab. Note that for CVS, to view the related source code commits, the project needs to be associated with at least one Repository.</td>
</tr>
<tr>
<td>View (Read-Only) Workflow</td>
<td>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.</td>
</tr>
</tbody>
</table>

### Issue Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign Issues</td>
<td>Permission to assign issues to users. Also allows autocompletion of users in the Assign Issue dropdown. (See also Assignable User permission below)</td>
</tr>
<tr>
<td>Assignable User</td>
<td>Permission to be assigned issues. (Note that this does not include the ability to assign issues; see Assign Issue permission above).</td>
</tr>
<tr>
<td>Close Issues</td>
<td>Permission to close issues. (This permission is useful where, for example, developers resolve issues and testers close them). Also see the Resolve Issues permission.</td>
</tr>
<tr>
<td>Create Issues</td>
<td>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).</td>
</tr>
<tr>
<td>Permission</td>
<td>Explanation</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Delete Issues</td>
<td>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.</td>
</tr>
<tr>
<td>Edit Issues</td>
<td>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.</td>
</tr>
<tr>
<td>Link Issues</td>
<td>Permission to link issues together. (Only relevant if Issue Linking is enabled).</td>
</tr>
<tr>
<td>Modify Reporter</td>
<td>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.</td>
</tr>
<tr>
<td>Move Issues</td>
<td>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.</td>
</tr>
<tr>
<td>Resolve Issues</td>
<td>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.</td>
</tr>
<tr>
<td>Schedule Issues</td>
<td>Permission to schedule an issue — that is, to edit the 'Due Date' of an issue. In older versions of JIRA this also controlled the permission to view the 'Due Date' of an issue.</td>
</tr>
<tr>
<td>Set Issue Security</td>
<td>Permission to set the security level on an issue to control who can access the issue. Only relevant if issue security has been enabled.</td>
</tr>
<tr>
<td>Voters &amp; Watchers Permissions</td>
<td>Explanation</td>
</tr>
<tr>
<td>Manage Watcher List</td>
<td>Permission to manage (i.e. view/add/remove users to/from) the watcher list of an issue.</td>
</tr>
<tr>
<td>View Voters and Watchers</td>
<td>Permission to view the voter list and watcher list of an issue. Also see the Manage Watcher List permission.</td>
</tr>
<tr>
<td>Comments Permissions</td>
<td>Explanation</td>
</tr>
<tr>
<td>Add Comments</td>
<td>Permission to add comments to issues. Note that this does not include the ability to edit or delete comments.</td>
</tr>
</tbody>
</table>
### Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete All Comments</td>
<td>Permission to delete any comments, regardless of who added them.</td>
</tr>
<tr>
<td>Delete Own Comments</td>
<td>Permission to delete comments that were added by the user.</td>
</tr>
<tr>
<td>Edit All Comments</td>
<td>Permission to edit any comments, regardless of who added them.</td>
</tr>
<tr>
<td>Edit Own Comments</td>
<td>Permission to edit comments that were added by the user.</td>
</tr>
<tr>
<td>Attachments Permissions</td>
<td></td>
</tr>
<tr>
<td>Create Attachments</td>
<td>Permission to attach files to an issue. (Only relevant if attachments are enabled). Note that this does not include the ability to delete attachments.</td>
</tr>
<tr>
<td>Delete All Attachments</td>
<td>Permission to delete any attachments, regardless of who added them.</td>
</tr>
<tr>
<td>Delete Own Attachments</td>
<td>Permission to delete attachments that were added by the user.</td>
</tr>
<tr>
<td>Time Tracking Permissions</td>
<td></td>
</tr>
<tr>
<td>Work On Issues</td>
<td>Permission to log work against an issue, i.e. create a worklog entry. (Only relevant if Time Tracking is enabled).</td>
</tr>
<tr>
<td>Delete All Worklogs</td>
<td>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.</td>
</tr>
<tr>
<td>Delete Own Worklogs</td>
<td>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.</td>
</tr>
<tr>
<td>Edit All Worklogs</td>
<td>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.</td>
</tr>
<tr>
<td>Edit Own Worklogs</td>
<td>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.</td>
</tr>
</tbody>
</table>

### 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. Choose **> Issues**. Select **Permission Schemes** to open the Permission Schemes page, which
displays a list of all permission schemes in your JIRA system and the projects that use each scheme. 

**Keyboard shortcut:** `g` + `g` + start typing *permission schemes*

3. Click the 'Add Permission Scheme' link.
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: The 'Add Permission Scheme' form*

<table>
<thead>
<tr>
<th>Add Permission Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Add</td>
</tr>
<tr>
<td>Cancel</td>
</tr>
</tbody>
</table>

5. You will return to the 'Permission Schemes' page which now contains 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. Choose **> Issues**. Select Permission Schemes to open the Permission Schemes page, which displays a list of all permission schemes in your JIRA system and the projects that use each scheme.

**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.

*Screenshot: Project Permissions*

<table>
<thead>
<tr>
<th>Edit Permissions — Demo permission scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>On this page you can edit the permissions for the 'Demo permission scheme' permission scheme.</td>
</tr>
<tr>
<td>• Grant permission</td>
</tr>
<tr>
<td>• View all permission schemes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Project Permissions</th>
<th>Users / Groups / Project Roles</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administer Projects</td>
<td></td>
<td>Add</td>
</tr>
<tr>
<td>Browse Projects</td>
<td></td>
<td>Add</td>
</tr>
<tr>
<td>View Issue Source Tab</td>
<td></td>
<td>Add</td>
</tr>
<tr>
<td>View Read-Only Workflow</td>
<td></td>
<td>Add</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Issue Permissions</th>
<th>Users / Groups / Project Roles</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create issues</td>
<td></td>
<td>Add</td>
</tr>
<tr>
<td>Edit issues</td>
<td></td>
<td>Add</td>
</tr>
<tr>
<td>Schedule issues</td>
<td></td>
<td>Add</td>
</tr>
<tr>
<td>Move issues</td>
<td></td>
<td>Add</td>
</tr>
</tbody>
</table>

4. Click the 'Add' link in the 'Operations' column, which displays the 'Add Permission' page.
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: 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. Choose **> Issues**. Select **Permission Schemes** to open the Permission Schemes page, which displays a list of all permission schemes in your JIRA system and the projects that use each scheme. **Keyboard shortcut:** `g + g +` permission schemes
3. Locate 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).
4. 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 as a user with the **JIRA Administrators** global permission.
2. Choose **> Projects**. **Keyboard shortcut:** `g + g +` projects
3. Select the project of interest to open the **Project Summary** administration page for that project. See **Defining a Project** for more information.
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'.
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.
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. Choose **> Issues**. Select **Permission Schemes** to open the Permission Schemes page, which displays a list of all permission schemes in your JIRA system and the projects that use each scheme. **Keyboard shortcut:** `g + g +` 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. Choose 🔄 ➤ **Issues**. Select **Permission Schemes** to open the Permission Schemes page, which displays a list of all permission schemes in your JIRA system and the projects that use each scheme.
   - **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.

### Global Permissions

This table lists the different global permissions and the functions they secure:

<table>
<thead>
<tr>
<th>Global Permission</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| **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 results in all newly created users being **automatically added** to that group. The exception to this are groups that also have either the **JIRA System Administrators** or **JIRA Administrators** permissions, since JIRA prevents groups with these administrative-level global permissions from being granted the **JIRA Users** permission. Furthermore, it would be unwise to automatically grant these administrative-level global permissions to all new users. |
<table>
<thead>
<tr>
<th>Global Permission</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Browse Users</strong></td>
<td>Permission to view a list of all JIRA user names and group names. Used for selecting users/groups in popup screens. Enables auto-completion of user names in most 'User Picker' menus and popups. Note that the Assign User permissions also allows a limited version of this on a per-project basis.</td>
</tr>
<tr>
<td><strong>Create Shared Objects</strong></td>
<td>Permission to share a filter or dashboard globally or with groups of users. Also used to control who can create a JIRA Agile board.</td>
</tr>
<tr>
<td><strong>Manage Group Filter Subscriptions</strong></td>
<td>Permission to manage (create and delete) group filter subscriptions.</td>
</tr>
</tbody>
</table>
| **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). |

**Managing Global Permissions**

Global permissions are system wide and are granted to groups of users.

See also project permissions, which apply to individual projects.

On this page:
- Granting global permissions
- Removing global permissions
- About 'JIRA System Administrators' and 'JIRA Administrators'
- Separating 'JIRA System Administrators' from 'JIRA Administrators' in default JIRA installations
- Troubleshooting permissions with the JIRA admin helper

This table lists the different global permissions and the functions they secure:
<table>
<thead>
<tr>
<th>Role</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>JIRA System Administrators</strong></td>
<td>Permission to perform all JIRA administration functions.</td>
</tr>
<tr>
<td></td>
<td>This does not include the JIRA Users permission.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td><strong>JIRA Administrators</strong></td>
<td>Permission to perform most JIRA administration functions (see list of exclusions below).</td>
</tr>
<tr>
<td></td>
<td>This does not include the JIRA Users permission.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td><strong>JIRA Users</strong></td>
<td>Permission to log in to JIRA.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>Granting the JIRA Users permission to a group results in all newly created users being automatically added to that group. The exception to this are groups that also have either the JIRA System Administrators or JIRA Administrators permissions, since JIRA prevents groups with these administrative-level global permissions from being granted the JIRA Users permission. Furthermore, it would be unwise to automatically grant these administrative-level global permissions to all new users.</td>
</tr>
<tr>
<td><strong>Browse Users</strong></td>
<td>Permission to view a list of all JIRA user names and group names. Used for selecting users/groups in popup screens. Enables auto-completion of user names in most 'User Picker' menus and popups.</td>
</tr>
<tr>
<td></td>
<td>Note that the Assign User permissions also allows a limited version of this on a per-project basis.</td>
</tr>
<tr>
<td><strong>Create Shared Objects</strong></td>
<td>Permission to share a filter or dashboard globally or with groups of users. Also used to control who can create a JIRA Agile board.</td>
</tr>
<tr>
<td><strong>Manage Group Filter Subscriptions</strong></td>
<td>Permission to manage (create and delete) group filter subscriptions.</td>
</tr>
</tbody>
</table>
**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).

**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).
   
   By default, the user account created during the JIRA Setup Wizard has both of these global permissions.

2. Choose ✉️ User Management. Select Global Permissions to open the Global Permissions page, which lists JIRA's global permissions.

   **Keyboard shortcut:** g + g + start typing global permissions

   The Add Permission box is shown at the bottom of the list (not displayed in the screen capture above).

3. In the Permission dropdown list, select the global permission you wish to grant.
4. In the **Group** dropdown 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**. This is **not** recommended for production systems, or systems that can be accessed from the public Internet such as OnDemand.

   **Please Note:**
   - 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.
   - The **JIRA Users** permission cannot be granted to groups that have the **JIRA System Administrators** or **JIRA Administrators** permissions.
   - If you have a user limited license (e.g. personal license) and have reached your user limit, you will not be able to grant the **JIRA Users** permission (i.e. permission to log in) to any further groups without first reducing the number of users with the **JIRA Users** permission.

### Removing 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 remove the **JIRA System Administrators** global permission. By default, the user account created during the **JIRA Setup Wizard** has both of these global permissions.

2. Choose **> User Management**. Select **Global Permissions** to open the Global Permissions page, which lists JIRA's global permissions.

   **Keyboard shortcut:** `g + g +` start typing **global permissions**

   For each global permission in JIRA (indicated on the left of this page), groups which currently have that permission are shown on the right (under the **Users / Groups** column).

3. Locate the global permission you want to remove from a group as well as the group you want to remove that permission from (under **Users / Groups** and click the **Delete** link next to that group.

### 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 separation 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.

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** that use an authentication type other than OAuth.
- 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.
permission.
- Upload and/or install an Add-on.

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.

Separating 'JIRA System Administrators' from 'JIRA Administrators' in default JIRA installations

By default, the jira-administrators group has both the JIRA Administrators permission and the JIRA System Administrators permission. Also by default, the user account created during the JIRA Setup Wizard is a member of this jira-administrators group.

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.

Troubleshooting permissions with the JIRA admin helper

The JIRA admin helper can help you diagnose why a user can or cannot see a certain issue. This tool is only available to JIRA administrators.

To diagnose why a user can or cannot see an issue:

1. Choose at the top right of the screen. Then choose Add-ons > Admin Helper > Permission Helper. (Keyboard shortcut: g + g + start typing 'Permission Helper'

2. Enter the username of the user (leave blank for anonymous users), an issue key (for example, an issue that the user can/cannot see) and the permission to check.

3. Click Submit.

Screenshot: Permissions helper (click to view larger image)

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.

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.

On this page:
- Manually ending a Secure Administrator Session
- Disabling Secure Administrator Sessions
- Changing the Timeout
- Developer Notes

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 administrator 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.

Use Strong Passwords

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.

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.

Apply JIRA Security Patches

Apply the patches found in any security advisories that we release for your version of JIRA.

These patches protect JIRA from recently detected privilege escalation and XSS vulnerabilities.

Protect Against Brute Force Attack

You can also actively protect your systems against repeated unsuccessful login attempts, known as “brute force” login attacks.

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.

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.

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.
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:
- Use Strong Passwords
  - Administrators should use Strong Passwords
  - Administrators should have Different Passwords for Different Systems
- Apply JIRA Security Patches
- Protect Against Brute Force Attack
  - Enable Brute Force Login Protection on your Web Server
- Restrict Network Access to Administrative Sections of Applications
- Restrict File System Access by the Application Server
- Disable Jelly
- Configuring Tomcat to use HttpOnly Session ID Cookies
- See Also

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:

```xml
...<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 random 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 (i.e. token). This token is generated by JIRA and is also stored for the user in the JIRA database.

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 (also if present), which is stored for the user in the JIRA database.

If the token in the cookie matches the token 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:

```xml
<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.
<table>
<thead>
<tr>
<th>Cookie Key</th>
<th>Purpose</th>
<th>Cookie Contents</th>
<th>Expiry</th>
</tr>
</thead>
<tbody>
<tr>
<td>atlassian.xsrf.token</td>
<td>Helps prevent XSRF attacks. Ensures that during a user's session,</td>
<td>Your JIRA server's Server ID, a securely-generated random string (i.e. token) and a flag indicating whether or not the user was logged in at the time the token was generated.</td>
<td>At the end of every session or when the browser is closed.</td>
</tr>
<tr>
<td></td>
<td>browser requests sent to a JIRA server originated from that JIRA server.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information about XSRF checking by JIRA, see Form Token Checking on the Atlassian Developers site.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jira.issue.navigator.type</td>
<td>Tracks which type of search view was last used (i.e. simple or advanced searching).</td>
<td>A string indicating the state of your last search view.</td>
<td>Approximately 10 years from the date it is set or was last updated.</td>
</tr>
<tr>
<td>AJS.conglomerate.cookie</td>
<td>Tracks which general tabs were last used (e.g. in JIRA's plugin manager) or expansion elements were last opened or closed.</td>
<td>One or more key-value strings which indicate the states of your last general tab views or expansion elements.</td>
<td>One year from the date it is set or was last updated.</td>
</tr>
<tr>
<td>UNSUPPORTED_BROWSER_WARNING</td>
<td>Acknowledges that the user has read a message displayed by JIRA indicating that the user's browser is not supported by JIRA.</td>
<td>A string which indicates that the user has clicked a button acknowledging they have read the message stating they are using an unsupported browser.</td>
<td>At the end of every session or when the browser is closed.</td>
</tr>
<tr>
<td>AJS.thisPage</td>
<td>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.</td>
<td>A string which indicates that the user's browser does not support local storage.</td>
<td>At the end of every session or when the browser is closed.</td>
</tr>
</tbody>
</table>

**JIRA Admin Helper**

The JIRA Admin Helper is a **free, bundled plugin** that answers questions like:

- Why isn't my field showing up on view/edit/create screens?
- Why can/can't a user see a certain issue?
- Why did/didn’t a user get a certain email notification?

The JIRA Admin Helper plugin is visible only to JIRA Administrators. When you are viewing an issue, it is available from the **Admin** menu.

**Field Helper**

If you’re logged in as a JIRA administration, you can use the Field Helper – displayed as a **Where is my field?** link – to help you determine why a field is not appearing on a specific screen. The Field Helper works with custom fields as well as JIRA system fields.

**Where is my field?** link is available on:
- Create Issue – in Configure Fields pop up
- Edit issue - in Configure Fields pop up
- View Issue- in More Actions menu
• Issue Navigator – in cog menu

Simply click on the link and then enter the field name in the search box!

Here's an example:

![Image of Issue Navigator in cog menu](image)

After you enter the name of the missing field, the Field Helper returns a form that explains why this field is not appearing:

![Image of Field Helper form](image)

You can then use this information to fix your screen by adding this field to your project and issue type.

Permission Helper
The JIRA Admin Helper can help you diagnose why a user can or cannot see a certain issue.

To diagnose why a user can or cannot see an issue:

1. Choose at the top right of the screen. Then choose Add-ons > Admin Helper > Permission Helper.
   - Keyboard shortcut: g + g + start typing 'Permission Helper'  
2. Enter the username of the user (leave blank for anonymous users), an issue key (for example, an issue that the user can/cannot see) and the permission to check.
3. Click Submit.

![Permission Helper](image)

Notification Helper

The Notification Helper can you help figure out why a user didn't get an email notification when a comment was added. It's available from the view issue page, the issue navigator, and from JIRA Administration.

To diagnose why a user didn't receive a notification:

1. Choose at the top right of the screen. Then choose Add-ons > Admin Helper > Notification Helper.
   - Keyboard shortcut: g + g + start typing 'Notification Helper'  
2. Enter the username of the user (leave blank for anonymous users) and select the Notification Event from the dropdown list.
3. Click Submit.

![Notification Helper](image)

Configuring Fields and Screens

Overview

To help you tailor JIRA to your organization's needs, JIRA enables you to manipulate the display and behavior 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’)

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’ ...
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 behaviors 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.

Info 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.

**Related topics**

- Configuring Built-in Fields
  - Defining 'Issue Type' Field Values
  - 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 Behavior
  - Associating Field Behavior with Issue Types
  - Configuring Renderers
- Defining a Screen
  - Associating a Screen with an Issue Operation
  - Associating Screen and Issue Operation Mappings with an Issue Type

**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 customized 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.

Info Reordering issue types changes the order in which they are displayed to the user who is creating an
Creating an issue type

When creating a new issue type in JIRA, you can create either a new standard or sub-task issue type. However, to create a sub-task issue type, you must Enable sub-tasks.

You can also create sub-tasks on the Sub-Tasks page. See Creating a sub-task issue type for details.

To create a new issue type:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 📚 > Issues. Select Issue Types to open the Issue Types page, which lists all issue types.
3. Click the Add Issue Type button to open the Add New Issue Type dialog box.
4. Complete the **Add New Issue Type** dialog box:
   - **Name** — enter a short phrase that best describes your new issue type.
   - **Description** — enter a sentence or two to describe when this issue type should be used.
   - **Type** — specify whether the issue type you are creating is a **Standard** issue type or a **Sub-Task** issue type. Sub-tasks are associated with individual **Standard** issues.
   - **Icon URL** — supply the path of an image that has been placed somewhere inside `<jira-application-dir>/images/icons` of your JIRA Installation Directory or from an accessible URL.

5. Click the **Add** button to create your new issue type.
   - Your new issue type 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**

**Warning**: 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. Choose > Issues. Select Issue Types to open the Issue Types page, which lists all issue types.
   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.
4. Complete the fields

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. Choose > Issues. Select Issue Types to open the Issue Types page, which lists all issue types.
   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.
4. Edit the Name, Description and/or Icon as described above for Creating an issue type.

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' defines a subset of issue types, which:

- restricts the set of available issue types for a project, and
- controls the order of available issue types and the default issue type shown to your users for a project.

The 'default issue type' is the issue type displayed in the selection-box when a user creates an issue.

A single issue type scheme can be 're-used' across multiple projects, so that a group of similar projects (i.e. projects which might be used for similar purposes) can share the same issue type settings.

For example, all projects in your company may fit one of two 'purpose' categories:

- Development-related projects or
- Support-related projects.

Hence, you could create one scheme called Development Issue Type Scheme (with issue types Bug and Feature) and another called Support Issue Type Scheme (with issue types Development Query and Support Request). You can then associate each of these schemes with the appropriate project(s), for which there may be a plethora.

This provides your users with a different set of issue types based on the project they decide to create issues in and furthermore reflects the purpose behind creating these issues.

Your future maintenance workload is minimised, because any change you make to an issue type scheme is made across all projects that are associated with the scheme. In the example above, adding a new issue type to all support-related projects only requires the simple step of adding the issue type to the Support Issue Type Scheme.
Managing issue type schemes

To access the 'Issue Type Schemes' page:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Issues > Issue Types > Issue Type Schemes to open the Issue Type Schemes page, which displays all existing issue type schemes, their related issue types and their associated projects.

Keyboard shortcut: g + g + start typing issue type schemes

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 Issue Type Scheme' as described below) and associate these 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).
2. Click the Add Issue Type Scheme button to open the Add Issue Type Scheme page.
3. Enter the Scheme 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.
4. To add issue types to your scheme, drag and drop an issue type from the **Available Issue Types** list on the right to the **Issue Types for Current Scheme** list on the left:

5. If you need an issue type that does not currently exist, you can easily add this by using the **Add New Issue Type** button and dialog box.
   - **This will add the issue type** to your JIRA system and also add it to **Issue Types for Current Scheme** list on the left.
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 in the selection-box when a user creates an issue.**
7. Set the **Default Issue Type** for the new scheme from the dropdown list.
   - **Please Note:**
     - The 'default issue type' is the issue type displayed in the selection-box when a user creates an issue.
     - The issue types in this list depend on the issues in the **Issue Types for Current Scheme** list on the left.
     - The **None** option means that there is no default value. If this option is selected, the system will show the first Issue Type listed in the **Issue Types for Current Scheme**.
     - The **Issue Type** is remembered as long as you keep creating issues in the same project. Once you change projects or log off the system, it goes back to the default value.
8. Click the **Save** button to create your issue type scheme.

---

**Editing an issue type scheme**

**To edit an Issue Type scheme:**

- Go to the **Issue Type Schemes** tab (see above).
- Click the **Edit** link (in the **Operations** column) to access and edit the relevant issue type scheme.
Please Note:

- The process of editing a scheme is identical to the creation process. While editing your issue type scheme, you can set the default default issue type and reorder, add or remove issue types.
- If an issue type scheme has been associated with one or more JIRA projects (below) and:
  - issues of the issue types (defined by this issue type scheme) already exist in any of these JIRA projects and
  - you then want to remove one or more of these issue types from this issue type scheme, you will be prompted to use the Issue Type Migration Wizard (below). This wizard will move your issues from the original issue type (which will no longer be applicable) to a valid one. If you cancel this process at any time, your changes will not be saved.

**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).
2. Click the Associate link (in the Operations column) for the relevant Issue Type scheme.
3. Using the multi-select Project box, choose the JIRA projects that you wish to apply your issue type scheme to.
4. Click the Associate button and all selected projects will change from their current scheme to the selected scheme.

**Please Note:** If a project you are attempting to associate your new issue type scheme with has issues with issue types which have not been added to this new issue type scheme, you will be asked to use the Issue Type Migration Wizard (below) to migrate the issues to a new issue type (made available by the new issue type scheme).

**Choosing a project’s issue type scheme**

You may want to change a project to use a different set of issue types.

**Please Note:** This is effectively the same as associating an issue type scheme with projects (above), but is performed from a project’s Project Summary administration page (and you cannot choose multiple projects in one action).

To change a project to use a different issue type scheme:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Projects. **Keyboard shortcut:** g + g + start typing projects
   In the Issue Types section, click the name of the current scheme to display the details of the project’s issue type scheme.
3. Click the Actions dropdown menu and choose Use a different scheme.
4. 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. 'Development Issue Type Scheme'), 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'** — Select this option 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. 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 cannot 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.

5. 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.

To translate your priorities into another language, please see Translating Resolutions, Priorities, Statuses and Issue Types.

Defining a new priority

1. Log in as a user with the JIRA Administrators global permission.

2. Choose Issues. Select Priorities to open the View Priorities page, which lists the currently-defined priorities and the Add New Priority form.

   Keyboard shortcut: g + g + start typing priorities

3. Complete the Add New Priority form towards the end of the page:
   - Name — specify a word or two to describe your new priority. (This name will appear in the dropdown field when a user creates or edits an issue).
   - Description — add a sentence or two to describe when this priority should be used.
   - Icon URL — supply the path of an image that has been placed somewhere inside <jira-applica
tion-dir>/images/icons of your JIRA Installation Directory or from an accessible URL.
• **Priority Color** — specify a color to represent this priority. You can either type the HTML color code, or click the box at the right of the field to select from a color chart.

4. Click the **Add** button.

**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.

**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. Choose **JIRA** > **Issues**. Select **Resolutions** to open the View Resolutions page, which lists the standard resolutions, along with a form for adding new resolutions.

   ✔ **Keyboard shortcut**: `g + g` + start typing **resolutions**
3. Complete the **Add New Resolution** form at the bottom of the page:
   - **Name** — enter a short phrase that best describes your new resolution.
   - **Description** — enter a sentence or two to describe when this resolution should be used.

   ! The **View Resolutions** page 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.

   ! **Don't create a Resolution named "Unresolved"/"None"**

   Any issue that has the Resolution field set is treated by JIRA as "resolved". The Issue Navigator displays Unresolved when no resolution is set for an issue. So adding a resolution named Unresolved/None and setting it in an issue will mean that the issue is seen by JIRA as resolved. This will lead to confusion and is not recommended.

### 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 customize 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. Choose **Issues** > **Issues**. Select **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

3. Complete the **Add New Status** form towards the end of the page:
   - **Name** — specify a short phrase that best describes your new status.
   - **Description** — add a sentence or two to describe what workflow step this status represents.
   - **Icon URL** — supply the path of an image that has been placed somewhere inside `<jira-application-dir>/images/icons` of your **JIRA Installation Directory** or from an accessible URL.
JIRA ships with a number of images that can be used as status icons. These images are located in the `<jira-application-dir>/images/icons` directory of your JIRA Installation:

- 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 internationalizable 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 (for either standard and sub-task issue types)
- 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.

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 translated issue type constants:

1. Log in as a user with the JIRA Administrators global permission.
2. Click the Translate link located on the issue constant management page — i.e.
   - the Manage Issue Types page (for standard issue types - click any of the Translate links),
   - the Sub Tasks page (for sub-task issue types),
   - the View Statuses page,
   - the View Resolutions page or
   - the View Priorities page.
   The relevant issue constant Translation page displays the translation set for the currently selected language.
3. To view/update a translation set for a specific language, select the required language from the View list at the top of the page and click the button.

A translated name and description set can be specified for each type of issue constant.

4. Once all translations have been entered, the translation set can be saved by clicking the Update button at the end of the page.

Note that:
- The process can be repeated for all of the issue constants — i.e. Issue Type, Status, Resolution and Priority fields.
- The translated issue constant name and description will be displayed throughout JIRA, e.g. in reports, gadgets and all issue views.

Adding a Custom Field

JIRA lets you add custom fields in addition to the built-in fields. When creating a custom field, you can choose between Standard and Advanced types. For standard types, a preview image is shown for each type, so you can see what you are creating in advance. This ensures that you get the custom field you want, much faster. To configure search templates or add contexts to custom fields, use the option on each custom field.

JIRA ships with over 20 custom field types and you can find many more in the Atlassian Marketplace (e.g. the JIRA Toolkit). To build your own custom field types, see the tutorial at the JIRA Developer Documentation.

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.

Adding a field directly to an issue

JIRA Admins can add an existing field or create a custom field while in View Issue with the Admin > Add field option. You can even configure the options for that custom field without having to leave the screens you are presented with.

See this page for instructions on adding a field from View Issue: Adding a field to an issue.

Adding a field using the Add Custom Field button

1. Log in as a user with the JIRA Administrators global permission.
Choose "Issues" > "Fields" > "Custom Fields" to open the Custom Fields page.

*Keyboard shortcut: g + g + start typing custom fields*

2. Click the Add Custom Field button. The following dialog is displayed:

By default, this dialog displays the Standard, or most common, choices for custom fields. Click on the Advanced option in the left navigation or use the search box if you don't see the field you are looking for right away.

3. Select a field and click the Next button.

4. Configure the selection criteria for your field, as shown in the example below:

```
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select an Option</td>
<td>Select a fruit</td>
</tr>
</tbody>
</table>
```

The Field Name will appear as the custom field's title in both entering and retrieving information on issues, whereas the Field Description is displayed beneath the data entry field when entering new issues and editing existing issues, but not when browsing issues.

5. Click the Create button when you are ready. You will be presented with a dialog that displays your field, in context, and you can select the display options at this point:
Anything you select here will be displayed in the issue you are editing.

6. Click **Submit** to finalize the process. You will now see your new custom field and selected criteria displayed in the issue, as shown here:

**Next steps**

If you wish to change the context or other variables in your custom field, 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 Behavior.
- **Hide/Show** — see Specifying Field Behavior.
- **Required/Optional** — see Specifying Field Behavior.
Viewing custom fields

To view the custom fields in your JIRA system:

1. Log in as a user with the [JIRA Administrators] global permission.
2. Choose [Issues] > [Fields] > [Custom Fields] to open the Custom Fields page.

*Keyboard shortcut: g + g + start typing custom fields*

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. Choose [Issues] > [Fields] > [Custom Fields] to open the Custom Fields page.

*Keyboard shortcut: g + g + start typing custom fields*
3. Locate the relevant custom field and choose [ cog icon ] > [Edit].
   - The Name is the label that appears to the left of the custom field when it is displayed to a user.
   - The Description is the Help text that appears below the custom field when it is displayed in the Simple Search column.
   - 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). Every custom field type has a preconfigured search template, but you may select a different template using this procedure.
4. Modify the fields as desired and click [Update].

Configuring a custom field

The custom field context – also known as custom field configuration scheme is not related to the field configuration scheme – specifies the following for the custom field:

- Default Value
- Options
- The issue types and projects to which the default values and options 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 is created automatically when you add your custom field.

Context

To change the project(s) and issue type(s) to which a given Default Value and Options apply:

1. Log in as a user with the [JIRA Administrators] global permission.
2. Choose [Issues] > [Fields] > [Custom Fields] to open the Custom Fields page.

*Keyboard shortcut: g + g + start typing custom fields*
3. Locate the relevant custom field and choose [ cog icon ] > [Configure].
4. Locate the context named Default Configuration Scheme for ... and click the [Edit Configuration] link.
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. Choose ![> Issues](issuesIcon) > **Issues**. Select **Fields** > **Custom Fields** to open the Custom Fields page.
   
   **Keyboard shortcut:** g + g + start typing **custom fields**
3. Locate the relevant custom field and choose **cog icon > Configure**.
4. Click the **Add new context** link. 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.

A custom field can only have one context per JIRA project. So you cannot have multiple contexts for different issue types in the same project.

### 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. Choose ![> Issues](issuesIcon) > **Issues**. Select **Fields** > **Custom Fields** to open the Custom Fields page.
   
   **Keyboard shortcut:** g + g + start typing **custom fields**
3. Locate the relevant custom field and choose **cog icon > Configure**.
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.
     
     **Tip:** 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](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).
   
   **Tip:** 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.
These options are case insensitive, so when using a select or multi-select list for a notification scheme, JIRA-ADMINISTRATORS will match the jira-administrators group. This means you cannot have both a JIRA-ADMINISTRATORS and a jira-administrators option, as they have the same name.

To edit a custom field's options:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > Issues. Select Fields > Custom Fields to open the Custom Fields page.
3. Locate the relevant custom field and choose cog icon > Configure.
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.)

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. Choose > Issues. Select Fields > Custom Fields to open the Custom Fields page.
3. Locate the relevant custom field and click the cog icon > Screens.
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 Behavior for more information.

Translating a custom field

You can translate the name and description of any custom field that you create into another language. You can only select from the language packs that are installed in JIRA.

To translate the field and description of a custom field to another language:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > Issues. Select Fields > Custom Fields to open the Custom Fields page.
3. Locate the relevant custom field and click the cog icon > Translate.
4. Choose the language pack that this custom field translation will belong to (e.g. French) and enter the translated strings for the Field Name and Description.

Troubleshooting custom fields

Using the JIRA admin helper

The JIRA admin helper can help you diagnose why a custom field is not showing on your screens. This tool is only available to JIRA administrators.

To diagnose why a custom field is not showing on the View Issue, Edit Issue or Create Issue screens:

1. Navigate to the View Issue, Edit Issue or Create Issue screen where the field is not showing.
2. If you are viewing an issue, click **More Actions > Where is my field?** If you are creating or editing an issue, click **Configure Fields > Where is my field?**
3. Enter the name of the field.
4. Click **Submit**.

**Tip:** You can also access the ‘Where is my field?’ dialog via the cog menu for each issue in the **issue navigator**.

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:

```html
<a href="http://www.mycompany.com/jirahelp/fieldhelp.html">get help</a>
```

Or using Javascript, you can have help text right in the field:

![ QA Contact field with help icon ]

where clicking the help icon makes hidden help text appear:

![ QA Contact field with help text ]

This can be done by entering the following as the field’s description:

```html
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;">\</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 Behavior**

A **field configuration** defines the behavior of **all fields** available in your JIRA installation, including JIRA’s own ‘fixed’/‘built in’ fields (known as ‘system’ fields) and **custom fields**.

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

When defining field behavior for one or more JIRA projects and the fields used by the issue types in these projects, you typically start by adding one or more new field configurations (see below). You then begin modifying the behavior of individual fields in these new field configurations.

✔ A new field configuration should be added for each project and issue type combination which requires specific fields to be present and/or fields that express unique behavior.

You can then associate each new field configuration with a different issue type through a ‘field configuration scheme’. A field configuration scheme can then be associated with one or more projects.

This process of association gives you the flexibility of defining field behavior on a per project, per issue type basis. For more information, please see the Overview Diagram.

**On this page:**
- 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
- Modifying field behavior
  - 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**

You can create multiple field configurations for use on separate projects and issue types.

- Multiple field configurations are organized 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 Behavior 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.

ℹ️ 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. Choose ![Issues] > Issues. Select [Fields] > [Field Configurations] to open the View Field Configurations page, which lists all your field configurations.
   ✔ Keyboard shortcut: `g` + `g` + start typing field configurations

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
3. Click the **Add New Field Configuration** button to open the **Add Field Configuration** dialog box.

4. Complete the **Add Field Configuration** dialog box:
   - **Name** — enter a short phrase that best describes your new field configuration.
   - **Description** (*optional but recommended*) — enter a sentence or two to describe when this field configuration should be used.

5. Click the **Add** button to add your new field configuration to JIRA. Once you have added your new field configuration, you can then begin modifying the behavior of its fields (below).

   **You will be taken directly to the View Field Configuration page, where you can modify the behavior of fields in your new field configuration. See Modifying field behavior (from step 4) below for details.**

**Editing a field configuration**

1. Log in as a user with the **JIRA Administrators** global permission.

2. Choose **Issues > Fields > Field Configurations** to open the View Field Configurations page, which lists all your field configurations.

   **Keyboard shortcut:** `g + g + field configurations`

3. Click the **Edit** link next to the field configuration you wish to edit.

4. On the **Edit Field Configuration** page, edit the field configuration’s **Name** and **Description**.

   **Please note:** The **Default Field Configuration** cannot be edited.

**Deleting a field configuration**

1. Log in as a user with the **JIRA Administrators** global permission.

2. Choose **Issues > Fields > Field Configurations** to open the View Field Configurations page, which lists all your field configurations.

   **Keyboard shortcut:** `g + g + 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**. The **Delete** link will not be available for field configurations which are associated with one or more field configuration schemes.

**Copying a field configuration**

1. Log in as a user with the **JIRA Administrators** global permission.

2. Choose **Issues > Fields > Field Configurations** to open the View Field Configurations page, which lists all your field configurations.

   **Keyboard shortcut:** `g + g + field configurations`

3. Click the **Copy** link next to the field configuration you wish to copy.

4. On the **Copy Field Configuration** page, specify the **Name** and **Description** for the field configuration to be copied.

   **The (initial) field settings between the original and copied field configurations will be identical.**
Please Note: a newly created field configuration will not take effect until you:

1. Associate your new field configuration to one or more issue types.
2. Associate that field configuration with one or more projects.

See Associating Field Behavior with Issue Types for more information.

Modifying field behavior

To modify the behavior of fields in JIRA, you need to modify the field configurations that those fields have been defined in.

To modify the behavior of a set of fields in a field configuration:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Issues > Field Configurations to open the View Field Configurations page, which lists all your field configurations.
   Keyboard shortcut: g + g + field configurations
3. Locate the field configuration of interest and click the Configure View Field Configuration page, which lists all system and custom fields in your JIRA installation for that field configuration.
   Please Note:
   - The Edit link only allows you to change the Name and Description of the field configuration, not of individual fields.
   - Note that the Edit link is not available for the Default Field Configuration on the View Field Configuration page (listing all field configurations defined in your JIRA installation).
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).

Please Note: a newly created field configuration will not take effect until you:

1. Associate your new field configuration to one or more issue types and then
2. Associate that field configuration with one or more projects.

See Associating Field Behavior with Issue Types for more information.

Editing a field's description

Fields can be given descriptions to better identify the meaning of the field. These descriptions are typically displayed under the fields they are associated through JIRA's user interface, for example, when creating an issue or editing it:

Screenshot: Sample description text shown beneath the ‘Assignee’ field

To edit the description of a field:

1. Follow the first three steps above (in Modifying field behavior) to access the field configuration whose field's description you wish to edit.
2. Click the Edit link next to the field you want to change and update the field's description.
3. Click the Update button to save your changes.

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 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 and 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 Version/s field, the Change Log and Road Map reports will not work.

To hide or show a field:

1. Follow the first three steps above (in Modifying field behavior) to access the field configuration whose fields you wish to hide or show.
2. Do either of the following:
   - If you no longer want to expose a field through JIRA's user interface, click the Hide link associated with that field.
     You can make this field visible again at any time by clicking the Show link.
   - If you want to show a field (which is currently hidden) through JIRA's user interface, click the Show link associated with that field.
     You can hide this field again at any time by clicking the Hide 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: Sample validation of the 'Fix Version/s' field

![Fix Version/s](https://example.com/fixversions.png)

To make a field required or optional:

1. Follow the first three steps above (in Modifying field behavior) to access the field configuration whose fields you wish to hide or show.
   When viewing a field configuration (see above), fields which are already required have that indication next to their name.
2. Do either of the following:
   - To make a field mandatory when used through JIRA's user interface, click the Required link associated with that field.
     The text Required will appear next to the field's name.
   - To make a field optional, click the Optional link associated with that field.
     The Required text next to the field's name will disappear.

Please note:

- Fields that are hidden cannot be set to required.
- If you make a field 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 behavior 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).
- Be aware that there is a feature request (JIRA-5783) to make a field required for only one transition. If you are interested, please watch that issue for status updates.

Changing a field's renderer

JIRA renderers affect how a JIRA field's content is either displayed to the user (for text fields) or how a user enters field data (for multi-select fields), thereby enabling you to choose a style which best suits your needs.
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 dropdown list of options; and
  - The **Select List Renderer**, which simply provides a dropdown list of options.

For custom fields of type **Multi Select**, only the Select List Renderer is available. Furthermore, when modifying a field configuration, you will not be able to configure a **Multi Select** custom field's renderer.

Before you change the renderer for a specific field, please read [Configuring Renderers](#), paying particular attention to the **Implications for JIRA operations** section.

**To change the renderer for a specific field:**

1. Follow the first three steps above (in **Modifying field behavior**) to access the field configuration whose field's renderer you wish to change.
   
   When viewing a field configuration (see above), 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.

2. 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.

3. 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**.

   Changing the renderer only affects how a JIRA field's content is displayed or how a user interacts with a multi-select field — it does not affect the issue data that exists in the system. Hence, you can therefore toggle between renderer types safely.

**Associating Field Behavior with Issue Types**

A **field configuration scheme** associates (or "maps") **field configurations** to issue types in a project. In turn, a field configuration scheme can be **associated** with one or more projects.

This means that you can define different **field configurations** for each issue type that is available in a given project. For example, it is possible to have separate field configurations for the **Bug** the **Improvement** issue types (whose associations are defined in a field configuration scheme) for a project called 'Test'. Refer to the [Overview Diagram](#) for more information.

Because a field configuration scheme can be associated with more than one project (and associations between field configurations and issue types in a field configuration scheme are flexible), you can minimize your administrative workload as you can reuse the same field configuration for the same (or different) issue types across multiple projects.

**Adding a field configuration scheme**

1. Log in as a user with the **JIRA Administrators** global permission.
2. Choose 📦 > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.

   ✔️ Keyboard shortcut: g + g + start typing field configuration schemes

3. Click the Add New Field Configuration Scheme button to open the Add New Field Configuration Scheme dialog box.

4. Complete the Add New Field Configuration Scheme dialog box:
   - Name — enter a short phrase that best describes your new field configuration scheme.
   - Description (optional but recommended) — enter a sentence or two to describe when this field configuration scheme should be used.

5. Click the Add button to add your new field configuration to JIRA.

   You will be taken directly to the Configure Field Configuration Scheme page, where you can start associating issue types with field configurations in your new field configuration scheme. See Modifying field behavior (from step 4) below for details.

Associating an issue type with a field configuration

1. Log in as a user with the JIRA Administrators global permission.

2. Choose 📦 > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.

   ✔️ Keyboard shortcut: g + g + start typing field configuration schemes

3. Click the Configure link for the field configuration scheme in which to create an association between an issue type and field configuration. 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 new field configurations since installing JIRA, you will only have JIRA’s Default Field Configuration to work with.

4. Click Associate an Issue Type with a Field Configuration. You will see this screen:

5. Select the desired issue type and field configuration and click the Add button.

   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 between an issue type and a field configuration

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Issues > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.
   ✔️ Keyboard shortcut: g + g + start typing field configuration schemes
3. Click the Configure link for the field configuration scheme that contains the association between a field configuration and issue type you want to remove. 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 since installing JIRA, you will only have JIRA's 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. Choose Issues > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.
   ✔️ Keyboard shortcut: g + g + start typing field configuration schemes
3. Click the Configure link for the field configuration scheme contains an association between a field configuration and issue type you want to change. 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 since installing JIRA, you will only have JIRA's 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.

6. Click the Update button.

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. Choose Issues > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.
   ✔️ Keyboard shortcut: g + g + start typing field configuration schemes
3. Click the Edit link next to the field configuration scheme whose name and description you wish to modify.
4. On the Edit Field Configuration Scheme page, edit the Name and Description of the field configuration scheme.
5. 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. Choose 🔄 > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.

✅ Keyboard shortcut: g + g + start typing field configuration schemes

3. Click the Delete link next to the field configuration scheme you wish to delete. You will be prompted to confirm your deletion.

ℹ️ You can only delete a field configuration scheme that is not associated with a project. The Delete link will not be available for field configuration schemes which are associated with one or more projects.

**Copying a field configuration scheme**

To copy a field configuration scheme:

1. Log in as a user with the JIRA Administrators global permission.

2. Choose 🔄 > Issues. Select Fields > Field Configurations to open the View Field Configurations page, which lists all your field configurations.

3. Select Administration > Issues > Fields > Field Configuration Schemes (tab) to open the View Field Configuration Schemes page, which lists all your field configuration schemes (if any exist).

✅ Keyboard shortcut: g + g + start typing field configuration schemes

4. Click the Copy link next to the field configuration scheme you wish to copy.

5. On the subsequent page, specify the Name and Description of the field configuration scheme to be copied.

6. Click the Copy button.

ℹ️ The (initial) associations between field configurations and issue types in both the original and copied field configuration schemes will be identical.

**Associating a field configuration scheme with a project**

To make your JIRA projects use your field configuration(s), you need to associate these field configuration(s) with issue types in a field configuration scheme (above) and then associate this field configuration scheme with a project. (This association means that the field configuration scheme will be applied to the project.) Once this is done:

- The issues in this project will use the field configuration(s) 'mapped' to their issue type (defined by the field configuration scheme associated with the project)

*but also:*

- The issue types available to this project are defined by the issue type scheme associated with the project.

Therefore, even though a project's field configuration scheme may associate various different field configurations with a large set of issue types, only a subset of these issue types (as defined by the project's issue type scheme) and hence, field configurations themselves, may be available in that project. In other words, the issue types available to a project are restricted by the project's issue type scheme.

ℹ️ 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. Access the Project Summary administration page for your project (see Configuring a project).

2. In the Fields section of this page, click the name of the current field configuration scheme.

3. Click the Actions dropdown menu and choose Use a different scheme.

4. In the resulting page, select the scheme you want to associate with this project.

ℹ️ Selecting None will result in all issue types available to your project using JIRA's Default Field Configuration.

5. Click the Associate button. You will be returned to the Project Summary administration page, with the project now associated with the selected field configuration scheme.

**Configuring Renderers**

**Overview**

JIRA renderers affect how a JIRA field's content is either displayed to the user (for text fields) or how a user...
enters field data (for multi-select fields), thereby 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 dropdown list of options; and
  - The **Select List Renderer**, which simply provides a dropdown list of options.
    
    For custom fields of type **Multi Select**, only the Select List Renderer is available. Furthermore, when modifying a field configuration, you will not be able to configure a **Multi Select** custom field's renderer.

Renderers are configured on a per field basis. To configure a renderer for a particular field, see *Specifying Field Behavior*. Note that you can configure the same field differently for different projects and issue types — see *Associating Field Behavior 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*.

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.

---

**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:
<table>
<thead>
<tr>
<th>Field</th>
<th>Available Renderers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Wiki Style Renderer (default), Default Text Renderer.</td>
</tr>
<tr>
<td>Comment</td>
<td>Wiki Style Renderer (default), Default Text Renderer.</td>
</tr>
<tr>
<td>Environment</td>
<td>Wiki Style Renderer (default), Default Text Renderer.</td>
</tr>
<tr>
<td>Component</td>
<td>Autocomplete Renderer (default), Select List Renderer.</td>
</tr>
<tr>
<td>Affects Version</td>
<td>Autocomplete Renderer (default), Select List Renderer.</td>
</tr>
<tr>
<td>Fix Version</td>
<td>Autocomplete Renderer (default), Select List Renderer.</td>
</tr>
<tr>
<td>Custom field of type &quot;Free Text Field (unlimited text)&quot;</td>
<td>Wiki Style Renderer, Default Text Renderer.</td>
</tr>
<tr>
<td>Custom field of type &quot;Text Field&quot;</td>
<td>Wiki Style Renderer, Default Text Renderer.</td>
</tr>
<tr>
<td>Custom field of type &quot;Multi Select&quot;</td>
<td>Select List Renderer.</td>
</tr>
<tr>
<td>Custom field of type &quot;Version Picker&quot;</td>
<td>Autocomplete Renderer (default), Select List Renderer.</td>
</tr>
</tbody>
</table>

**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.

```
Description
This relates to ANGRY-304
```

> 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 organize 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.

**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 and Select List Renderers**

The Autocomplete and Select List Renderers let you start typing text, which is then autocompleted, or to select from a dropdown list of options.

![Autocomplete and Select List Renderers](image)

**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 present 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.

**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:

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 options. Unfortunately, in this context it is not possible to tell which renderer should be used for editing. However, 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 Behavior.

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 as follows.

1. Choose > Add-ons. The 'Find add-ons' screen shows add-ons available via the Atlassian Marketplace. Choose Manage Add-ons to view the plugins currently installed on your JIRA site.
2. Select Manage Add-ons and then search for 'renderer', filtering for System Add-ons, as shown here:
This screen displays all the configured renderers within JIRA.

- Click the **Disable** button to deactivate the renderer for the entire instance of JIRA.

Any fields still set up to use a disabled renderer will fall back to the default text renderer. When you attempt to edit the field, a warning message alerts you to the fact that you are configured to use a renderer that is not available.

When a renderer is disabled it will not be available for selection when changing a field's renderer. To enable the renderer, click the **Enable** button. Enabling or 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 as follows.

1. Choose **Add-ons**. The ‘Find add-ons’ screen shows add-ons available via the Atlassian [Marketplace](#). Choose **Manage Add-ons** to view the plugins currently installed on your JIRA site.
2. Select **Manage Add-ons** and then search for 'renderer', filtering for System Add-ons.
3. Expand the **Wiki Renderer Macros Plugin** to display the following screen.
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**

Screens group all available fields (or a subset of all available fields) defined in JIRA and organize them for presentation to a user. Through screens, you can control what fields are displayed to the user during issue operations (e.g. Create Issue and Edit Issue dialog boxes) or workflow transitions (e.g. Resolve Issue dialog box), as well as define the order in which these fields are shown to them. A screen also allows you to split subsets of fields across multiple tabs.

When it comes to field visibility, screens functionally overlap slightly with field configurations. For example, on the Create Issue dialog box, users will only see issue fields that:

1. are present on the screen associated with the issue's Create Issue issue operation,
2. are also not hidden in the field configuration applicable to the issue (as defined by the project's field configuration scheme),
3. the user has permission to edit (e.g. the Due Date field can only be edited by users with the Schedule Issues project permission),

Hence, a field may be present on a screen used by a project, but if that field is hidden in the field configuration used by the project, that field will not be visible to the user when that screen in the project is displayed.

If a particular field needs to be hidden at all times, it is easier to hide the field in the relevant field configuration than remove it from all screens. For more information please see the Overview.

Be aware that any newly created screen in JIRA is not usable by a JIRA project until it has been associated with either:

- An issue operation and issue type (via a screen scheme and then issue type screen scheme)
- OR
- A workflow transition.

See Activating a screen (below) for details.
JIRA ships with the Default Screen, Resolve Issue Screen and Workflow Screen, which are used as described below:

- **Default Screen** — used for the default issue operations for creating, editing or viewing an issue.
- **Resolve Issue Screen** — used for the transition view for the default Close Issue and Resolve Issue transitions, originating from the Open, In Progress and Reopened steps in JIRA's default workflow.
- **Workflow Screen** — used for the transition view for the default Reopen Issue transitions, originating from the Resolved and Closed steps and Close Issue transition, originating from the Resolved step in JIRA's default workflow.

The **Workflow Screen** defines a smaller set of fields than the **Resolve Issue Screen**.

JIRA's default workflow, showing transitions (arrows) and steps (blue boxes):

Adding a screen

**To add a new screen to JIRA:**

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > Issues. Select Screens to open the View Screens page, which lists all screens that have been defined in JIRA.

   ✔ **Keyboard shortcut:** `g + g` + type screens
3. Click the **Add New Screen** button to open the **Add New Screen** dialog box.
4. Complete the **Add New Screen** dialog box:
   - **Name** — enter a short phrase that best describes your new screen.
   - **Description** — enter a sentence or two to describe the situations screen will be used.
5. Click the **Add** button to add your new screen to JIRA.
   
   You will be taken directly to the **Configure Screen** page, where you can add fields to your new screen. See the **Configuring a screen’s fields** section below for details.

### 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. Choose **Issues** > **_screen**. Select **_screens** to open the View Screens page, which lists all screens that have been defined in JIRA.
   
   **Keyboard shortcut:** `g + g + _screens`
3. 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.

```java
Edit Screen

Use the form below to change properties of the Resolve Issue Screen.

<table>
<thead>
<tr>
<th>Name</th>
<th>Resolve Issue Screen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Allows to set resolution, change fix versions and assign an issue.</td>
</tr>
</tbody>
</table>

Update Cancel
```

### Copying a screen

1. Log in as a user with the **JIRA Administrators** **global permission**.
2. Choose **Issues** > **_screens**. Select **_screens** to open the View Screens page, which lists all screens that have been defined in JIRA.
   
   **Keyboard shortcut:** `g + g + _screens`
3. 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:
Deleting a screen

1. Log in as a user with the **JIRA Administrators global permission**.
2. Choose `> Issues`. Select **Screens** to open the View Screens page, which lists all screens that have been defined in JIRA.
   - **Keyboard shortcut:** `g + g + type screens`
3. Click the **Delete** link next to the screen you wish to delete. You will be prompted to confirm your deletion.

   Screens that are associated with one or more **screen schemes**, or one or more **workflow transitions**, cannot be deleted.

Configuring a screen’s tabs and fields

You can configure the fields that display on a particular screen by adding/removing fields, as well as reordering them. Tabs can also be used to help group related fields. Tabs are useful for organizing complex screens, as you can place less used fields onto separate tabs. You can also add, remove and reorder tabs, as well as rename them.

**To configure a screen’s tabs and fields:**

1. Log in as a user with the **JIRA Administrators global permission**.
2. Choose `Issues`. Select **Screens** to open the View Screens page, which lists all screens that have been defined in JIRA.
   - **Keyboard shortcut:** `g + g + type screens`
3. Click the **Configure** link (under the **Operations** column) next to the screen you want to add a field to, to open the **Configure Screen** page for that screen.
### Rename a tab

1. Hover over the tab name and click the **pencil icon**.
2. Enter the new name and click **OK**.

### Delete a tab

Hover over the tab name and click the **X**.

### Add a field

1. Click the tab that you want to add the field to.
2. Type the name of the field in the dropdown displayed at the bottom of the current fields. Field suggestions will appear as you type.
3. Click **Add Field** to add it to the current tab.

### Move a field

Hover over the dotted part of the field (next to the field name) and drag the field to the desired position.

Move a field to a different tab by dragging it to the name of the tab and dropping it.

### Delete a field

Hover over the field and click the **Delete** button that appears.

---

### Tips on configuring screens

- **Date fields on View Issue screen** — Fields of type 'Date' will always be displayed in the 'Dates' area of the default 'View Issue' screen, regardless of how you reorder them. This applies even if the dates are custom fields.

- **System fields on View Issue screen** — System fields on the default 'View Issue' screen (e.g. Summary, Security Level, Issue Type, etc.) are fixed. This means that they will always appear on the 'View Issue' screen, even if you configure the Screen to move them onto a separate tab. Any custom fields (except 'Date' fields) that have been added to the 'View Issue' screen can be moved onto a separate tab. Note, this restriction only applies to the screen associated with the 'View Issue' operation.

- **Timetracking** — You can add the ability to log work and/or specify/modify time estimates to a screen by adding the special **Log Work** and/or **Time Tracking** fields respectively. 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.
  
  - If these fields cannot be found in the **Add Field** selection box and they have not already been added to the screen, check whether JIRA's **Time Tracking feature** has been enabled. These fields will not be available to add to any screen if Time Tracking is disabled.
  
  - 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.

- **Renaming standard JIRA fields** — You cannot rename the standard JIRA fields (e.g. Priority, Summary, etc) via the JIRA administration console. If you want to rename the standard JIRA fields, you will need to modify files in your JIRA installation. Please see this [knowledge base article](https://confluence.atlassian.com/jirahelp/for-users/os-gui-11-0-0-important-base-articles) for instructions. Note, renaming standard JIRA fields is not supported.

### 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.

---

*Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
In a screen scheme, you can specify the same screen (or choose different screens) for these issue operations. Once you have created your screen scheme, you will need to activate it by associating the screen scheme with issue types via an 'issue type screen scheme'. (In turn, issue type screen schemes are associated with JIRA projects.)

Please be aware that although it is possible to associate any screen defined in your JIRA installation with either a screen scheme or a workflow transition view, screen schemes and workflow transition views are distinct and unrelated.

On this page:
- What is a 'screen scheme'?
- Adding a screen scheme
- Editing a screen scheme's details
- Deleting a screen scheme
- Copying a screen scheme
- Configuring a screen scheme
  - Associating a screen with an issue operation
  - Editing an association
  - Deleting an association
- Activating a screen scheme

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. Log in as a user with the JIRA Administrators global permission.
2. Choose > Issues. Select Screens > Screen Schemes to open the View Screen Schemes page.
   Keyboard shortcut: g + g + type screen schemes
3. Click the Add New Screen Scheme button on the View Screen Schemes page.
4. Fill out the details for the new screen scheme on the form that is displayed.

Note: The default screen is 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. Log in as a user with the JIRA Administrators global permission.
2. Choose > Issues. Select Screens > Screen Schemes to open the View Screen Schemes page.
   Keyboard shortcut: g + g + type screen schemes
3. The View Screen Schemes page is displayed. Click Edit next to the desired screen scheme.
4. 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.

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. Log in as a user with the **JIRA Administrators** global permission.

2. Choose 🚀 > **Issues**. Select **Screens > Screen Schemes** to open the View Screen Schemes page.

3. The **View Screen Schemes** page is displayed. Click the **Delete** link next to the desired Screen Scheme. You will be prompted to confirm your deletion.

4. Copying a screen scheme

   1. Log in as a user with the **JIRA Administrators** global permission.

   2. Choose 🚀 > **Issues**. Select **Screens > Screen Schemes** to open the View Screen Schemes page.

   3. The **View Screen Schemes** page is displayed. Click **Copy** next to the Screen Scheme you wish to copy.

   4. 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.

5. Configuring a screen scheme

   Associating a screen with an issue operation

   1. Log in as a user with the **JIRA Administrators** global permission.

   2. Choose 🚀 > **Issues**. Select **Screens > Screen Schemes** to open the View Screen Schemes page.

   3. Locate the screen scheme in which you are interested, and click the **Configure** link next to it. The **Configure Screen Scheme** page is displayed:
4. Click **Associate an Issue Operation with a Screen** to open this window:

5. Do the following:
   a. Select the Issue Operation with which you wish to associate a Screen.
   b. Select the desired Screen.

**Important Notes**

1. 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.

2. 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. Click **Edit** next to the **Default** entry to change the Screen that is associated with it.

3. The **View Issue** operation only allows you to control the layout of custom fields in the middle of the **View Issue** page. It 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. Choose 🏛 > **Issues**. Select **Screens** > **Screen Schemes** to open the View Screen Schemes page.

3. **Keyboard shortcut**: `g + g` + type `screen schemes`

4. The **View Screen Schemes** page is displayed.

5. Locate the Screen Scheme in which you are interested, and click the **Configure** link next to it. The **Configure Screen Scheme** page is displayed.

6. On the **Configure Screen Scheme** page, click **Edit** next to the issue operation you wish to edit. The **Edit Screen Scheme Item** page is displayed:

6. Select the desired screen and click **Update**.
Deleting an association

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🌐 > Issues. Select Screens > Screen Schemes to open the View Screen Schemes page.
3. The View Screen Schemes page is displayed.
4. Locate the Screen Scheme in which you are interested, and click the Configure link next to it. The Configure Screen Scheme page is displayed.
5. On the Configure Screen Scheme page, click the Delete link next to the issue operation you wish to remove.

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.
2. Associate the Issue Type Screen Scheme with a project.

For details of both procedures, see Associating screens with Issue Types.

What is an 'issue type screen scheme'?

An 'issue type screen scheme' associates a screen scheme (which defines mappings between screens and issue operations) with issue types. Hence, an issue type screen scheme allows you to specify different screens for different issue types when used for the same issue operation (e.g. 'Create Issue') in a given JIRA project.

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.

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'.

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

Associating an issue type with a screen scheme

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🌐 > Issues. Select Screens > Issue Type Screen Schemes to open the View Issue Type Screen Schemes page.
3. 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: 

   ![Configure Issue Type Screen Scheme](image)

4. Click **Associate an issue Type with a Screen Scheme**, which displays this screen: 

   ![Associate an issue Type with a Screen Scheme](image)

5. Select an **Issue Type** you wish to associate a Screen Scheme with.
6. Select the desired **Screen 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. Choose **> Issues**. Select **Screens > Issue Type Screen Schemes** to open the View Issue Type Screen Schemes page.
   - 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.
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. Choose 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'
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.

Adding an issue type screen scheme

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 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'
3. Click the Add Issue Type Screen Scheme button to open this page:

   ![Add Issue Type Screen Scheme](image)

4. Enter the name for the new scheme. You can optionally add a description.
5. 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 entry in the scheme.
6. 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. Choose 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'
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.

Deleting an issue type screen scheme

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > Issues. Select Screens > Issue Type Screen Schemes to open the View Issue Type Screen Schemes page.

   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.

   [Image of delete screen]

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. Choose > Issues. Select Screens > Issue Type Screen Schemes to open the View Issue Type Screen Schemes page.

   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.

[Image of copy screen]

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. Choose > Projects.
3. Select the project you wish to configure by clicking on its name.
4. Select Screens, you should see something like this:

![Angry Nerd Screen Scheme](image)

5. Click the Actions dropdown menu and choose **Use a different scheme**.

6. Select the Screen Scheme you wish to associate with this project.

![Type Screen Scheme Association](image)

7. Click the **Associate** button.

> To control which issue types apply to a project, please see ‘Associating Issue Types with Projects’.

### Configuring Workflow

A JIRA workflow is the set of **statuses** and **transitions** that an issue goes through during its lifecycle.

*Workflows typically represent business processes.*

JIRA ships with a **built-in workflow** called **jira**, which is the default system workflow and cannot be edited. However, you can make copies of this workflow to get started creating your own workflows quickly. You can also create your own workflows from scratch, or import them from **Atlassian Marketplace**, if you prefer. Each workflow you create can be associated with particular projects and, optionally, specific **issue type(s)** by using a **workflow scheme**.
What is a status?

A status represents the state of an issue at a particular point in a specific workflow. An issue can be in only one status at a given point in time.

When defining a status, you can optionally specify properties.

What is a transition?

A transition is a link between two statuses that enables an issue to move from one status to another. In order for an issue to move between two statuses, a transition must exist.

A transition is a one-way link, so if an issue needs to move back and forth between two statuses, two transitions need to be created. The available workflow transitions for an issue are listed on the View issue screen, shown here.
Editing a workflow

Editing a workflow means that you are modifying the statuses and transitions that make up a workflow. There are slight differences between editing an inactive and an active workflow, described below. We place restrictions 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 use this workflow.

Active versus inactive workflows

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 edit the workflow’s steps and transitions. For details on this, see Working in text mode.

An active workflow is a workflow that is currently being used by one or more projects.

When you edit an active workflow, JIRA first creates a draft of it, as shown here.

You can then modify the draft as you see fit. Once you have finished modifying your draft workflow, you can publish your draft and, optionally, save your original workflow as an inactive backup.

Limitations when editing an active workflow

Please note that the following limitations apply when editing an active workflow (i.e. a draft workflow):

- It is not possible to edit the workflow name (only the description) if a workflow is active.
- Workflow statuses cannot be deleted.
• An associated Status cannot be edited.
• If a status has no outgoing transitions (Global transitions are not considered), it cannot have any new outgoing transitions added.
• The step ID cannot be changed. See the following article for details on this: Cannot Add Transitions or Delete Steps in Draft Workflows.

If you wish to make any of the modifications listed above, then you will need to copy the workflow (see Creating a workflow), modify the copy and then activate it.

Workflow designer

- Statuses in the workflow designer
- Transitions in the workflow designer
- Workflow designer tips
- Advanced transition configuration
- Differences from the classic mode
  - JIRA workflow designer changes
  - JIRA workflow transition changes

Workflow designer, also referred to as Diagram edit mode, allows you to visualize the entire layout of your workflow as well as create and edit a workflow's steps and transitions. For versions of this feature prior to JIRA 6.1, see Classic mode workflow designer.

The JIRA workflow designer looks like this:

![Workflow designer diagram]

**Statuses in the workflow designer**

There are a number of different actions you can perform with statuses in the workflow designer. By selecting a status, extra information and actions also become available in the Properties panel on the right-hand side of the screen, where you can:

- **Add existing status** – gives you a list of existing statuses to choose from.
- **Create new status** – displays a dialog where you can create a new status in your JIRA instance.
- **Edit a status** – lets you change the name and description of the status.
- **Remove a status** – removes the status from the workflow, but not the JIRA instance.
- **Move the status on the screen** – click and drag the status to reposition it in the diagram. Geometric snap lines appear to help you align the status with other statuses in the workflow.
- **Edit properties** – properties are advanced configurations on a workflow. Please see Workflow properties for details.
- **Add a global transition** – global transitions are transitions that allow every other status in the workflow to transition to the selected status.
Diagram showing a selected status, and the operations that can be performed on statuses

>Note: Statuses are global objects in JIRA. Changing the name of a status on one workflow also changes the name of the status on all workflows that use that status.

Transitions in the workflow designer

There are a number of different actions you can perform with transitions in the workflow designer. When you select a transition, additional information and actions become available in the Properties panel on the right-hand side of the screen, where you can:

- **Add a transition** – dragging a port of any status to a port of another status (see the illustration on the right) creates a new transition between the two statuses.
- **Reposition a transition** – selecting a transition highlights the two endpoints of the transition with black dots. Clicking and dragging either of those dots gives you the ability to reposition a transition around its given status.
- **Delete a transition** – click the Delete transition button or just use the delete key on your keyboard.
- **Edit a transition** – lets you change the name and description. You can also change the screen that the transition uses, see Working in text mode for details.
- **Configure advanced options** – such as properties, post functions, conditions and validators. Please see the Advanced transition configuration section below.
create a new transition

Diagram showing a selected transition and the operations that can be performed on transitions

Workflow designer tips

- Hover over a transition or a status to see the relevant transition labels.
- When dragging a status on the page, the red lines that display are geometric snap lines that can be used to snap to other statuses.
- Zoom the diagram with your mouse wheel. Pan the diagram by clicking and holding the mouse while on white space, then moving your mouse across the diagram.
- On the View Issue page, all statuses except the current one are displayed with the white background and blue text. The current status displays a blue background with white text to give it more prominence.
- The older Flash version of the workflow designer is still accessible from classic mode. For more information, see Classic mode workflow designer.

Advanced transition configuration

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, such as:
  - Assign the issue to a particular user.
  - Send an email notification.
  - Update a field in the issue.

For more information on working with conditions, validators, and post functions, see Advanced workflow configuration.
Workflow transition screen, displaying the steps that use this transition (at the top) as well as the conditions available (at the bottom)

Transitions into and out of a specific step (the "In Progress" step)

Differences from the classic mode

**JIRA workflow designer changes**

Features that exist in the classic mode of the Flash-based workflow designer, but not in the current workflow designer:

- Create common transitions
- Clone transitions
- Global transitions with no result status
- You cannot reassign the target of a transition – in other words, you cannot grab the arrow and make it point to something else – you need to remove the transition and recreate it instead
- You cannot directly set the `issue.editable` property. To do this, simply add the `issue.editable` property to the `status properties`

**JIRA workflow transition changes**

- Post-functions, conditions, and validators cannot be viewed together in one long list.
- Workflow transition IDs are no longer displayed in the user interface. (You can still see it when viewing a workflow text mode.)

Configuring a workflow

- Creating a workflow from an existing workflow
- Creating a workflow from scratch
- Importing a workflow
Creating a workflow from an existing workflow

1. Log in as a user with the JIRA Administrators global permission.
2. Choose "Issues". Select Workflows to open the Workflows page, which displays all of the workflows in your system.

![Workflows page](image)

3. Copy an existing workflow using the Copy link in the Operations column (shown above).
   a. Enter a name and description for your workflow.
   b. Click the Copy button. The workflow opens in edit mode.
4. Once you have created your new workflow, you may customize it by adding and/or editing steps and transitions.

When you have finished customizing your workflow, see Activating workflow for details on how to use it with a JIRA project.

Creating a workflow from scratch

For advanced administrators

1. Follow Steps 1 and 2 as described above (in "Creating a workflow from an existing workflow").
2. At Step 3, click the Add Workflow button instead of copying an existing workflow.
3. Enter a name and description for your workflow. Click the Add button.
   The workflow opens in edit mode and contains a step called Open. If you are viewing your workflow in Diagram mode, you see an incoming transition called Create.
4. Continue with your workflow customizations, by adding and editing steps and transitions.

Importing a workflow

Please see the documentation on Importing from Atlassian Marketplace.

Setting the Resolution field

Within JIRA 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).

Therefore, 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. See Working in text mode for details on this.

Renaming workflow transition buttons
If you copied the system workflow and you wish to rename the workflow transition buttons on the View issue page, you must delete the following properties from all transitions in the copied workflow:

- `jira.i18n.title`
- `jira.i18n.description`

Otherwise, the default names (i.e. values of these properties) will persist. Read more about transition properties.

Editing a project's workflow for the first time

Whenever a new JIRA project is created, your project automatically uses the default workflow scheme, which associates all available issue types in the project with the JIRA system workflow. Since neither the JIRA system workflow nor the default workflow scheme are editable, JIRA creates an editable copy of the system workflow and workflow scheme for your project.

To begin editing your project's workflow for the first time:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > Projects.
3. On the Project Administration page, select Workflows.
4. In the displayed workflow, click the Edit icon at the top-right of the box (shown here):

A message is displayed letting you know that you are editing your workflow for the first time. Click Continue to proceed.

5. JIRA automatically does the following:
   - Creates a copy of the system workflow named Your Project Name Workflow.
   - Creates a new workflow scheme for Your Project Name Workflow named Your Project Name Workflow Scheme.
   - Associates any existing issues in your project with the new Your Project Name Workflow.

You can now edit your draft workflow. When you are finished, you are presented with a dialog where you can publish your draft and, optionally, save your original workflow as an inactive backup.

Usage notes:

- If you have only a small number of existing issues in your JIRA project, this process is relatively quick.
- If you have many (e.g. thousands of) existing issues in your JIRA project, this process may take some time.
- Once this process begins, it cannot be paused or cancelled. Please avoid editing or transitioning any issues within your project while this process is taking place.

Working in text mode

Text mode is an advanced way of working with workflows, and it shows the difference between steps and statuses. In text mode, you work directly with steps. For details, see Working in text mode.

Advanced workflow transitions
For more information on workflow transitions, including built-in JIRA conditions, combining conditions into groups, applying validators and post functions, see Advanced workflow configuration.

**Activating workflow**

Workflows need to be activated to use them in JIRA. Activating a workflow is the process of mapping the workflow to a workflow scheme, and then associating the workflow scheme with a project.

A workflow scheme defines a set of associations – or mappings – between a workflow and an issue type. Workflow schemes are associated with a project and make it possible to use a different workflow for every combination of project and issue type.

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.
- **Active** workflow schemes are those associated with projects. **Inactive** workflow schemes are not associated with any projects.

To configure a workflow scheme, see [Configuring workflow schemes](#).

**On this page:**
- Activating a workflow
- Associating a workflow scheme with a project
- Disassociating a workflow scheme from a project

**Related topics:**
- Configuring workflow schemes
- Configuring Workflow
- Sharing your workflow

### Activating a workflow

To activate a workflow:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Create a workflow scheme or find an existing workflow scheme. See [Configuring workflow schemes](#) for instructions.
3. Configure the workflow scheme to use your workflow. See [Configuring workflow schemes](#) for instructions.
4. Associate your workflow scheme with a project, as described in the **Associating a workflow scheme with a project** section below.

### 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.
   - In some cases, JIRA will drop you out of administration mode. To get back, click the **Administer Project** button on the top right.
2. Choose **Projects**. The **Project Summary** page is displayed.
   - **Keyboard shortcut:** `g + g +` start typing `projects`
Projects Summary screen

3. Click Workflows on the left of the Project Summary page (you can also click the More link in the Workflows section in the middle of the screen). The Workflows page is displayed, indicating the current workflow scheme used by the project.

4. Click the Switch Scheme link to display the Associate Workflow Scheme to Project page.

5. Select the relevant workflow scheme from the Scheme list and click the Associate button to begin the migration process.

Each issue has to be in a valid status. The valid statuses for an issue are defined by its workflow. This means that when changing a workflow, you may need to tell JIRA the status for specific issues after the change.

Publish Workflows status mapping screen

A screen displays that indicates the progress of migrating all the project's issues to the updated scheme's workflows.

6. Click Acknowledge to finish the process. A message displays letting you know that your workflows have been published.

Please Note:

- You can associate a single workflow scheme with more than one project although, only one workflow scheme can be associated with a given project.
- The issue type scheme associated with a project defines the issue types that are available to that project. If an issue type is not defined in the project's issue type scheme, its workflow is not used.

Disassociating a workflow scheme from a project

A JIRA project must always be associated with a workflow scheme, since all issues must move through a
workflow, even if that workflow only consists of a single Create Issue transition.

All JIRA projects with workflows that haven't been modified (in any way) or that have not yet been associated with a different workflow scheme, use JIRA's system workflow. So, in this sense, disassociating a workflow scheme means reassociating your project's workflow with JIRA's default workflow scheme.

To dissociate a workflow scheme from a project:

1. Follow the instructions in Associating a workflow scheme with a project above.
2. When selecting the workflow scheme from the Scheme list, select the Default workflow scheme.
3. Click the Associate button.
4. Follow the wizard, which guides you through migrating all of the project's issues to the JIRA's system workflow.

Classic mode workflow designer

If you are using the latest version of JIRA, there is a newer version of the workflow designer. Go here for details. For the Classic mode workflow designer, see the information on this page.

- Usage notes
- Using the main toolbar
- Using the Statuses panel
- Using the Global Transitions panel
- Editing or deleting annotations

The illustration below shows the main features of the Classic mode workflow designer, which are discussed on this page.

Usage notes

- The Statuses and Global Transitions panels are expandable. If these panels are not visible (i.e. collapsed), expand them 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.

Using the main toolbar
The main toolbar lets you visually arrange your workflow's steps/statuses and transitions on the workflow design area. For more on working with transitions, go [here](#).

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Select Tool" /></td>
<td><strong>Select Tool</strong>&lt;br&gt;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.</td>
</tr>
<tr>
<td><img src="image" alt="Create Transition — Straight Line" /></td>
<td><strong>Create Transition — Straight Line</strong>&lt;br&gt;Creates a transition with a single straight line.</td>
</tr>
<tr>
<td><img src="image" alt="Create Transition — Polygonal Line" /></td>
<td><strong>Create Transition — Polygonal Line</strong>&lt;br&gt;Creates a transition with multiple straight lines.</td>
</tr>
<tr>
<td><img src="image" alt="Create Transition — Bezier Line" /></td>
<td><strong>Create Transition — Bezier Line</strong>&lt;br&gt;Creates a transition with a curved line.</td>
</tr>
<tr>
<td><img src="image" alt="Create Annotation" /></td>
<td><strong>Create Annotation</strong>&lt;br&gt;Adds a 'sticky note' where you can add a short description or annotation to your workflow's layout. Annotations only appear in the workflow design area and are not visible when viewing workflows from the <a href="#">View issue</a> page (via the View Workflow link).</td>
</tr>
<tr>
<td><img src="image" alt="Zoom In" /></td>
<td><strong>Zoom In</strong></td>
</tr>
<tr>
<td><img src="image" alt="Zoom Out" /></td>
<td><strong>Zoom Out</strong></td>
</tr>
<tr>
<td><img src="image" alt="Save Snapshot Image" /></td>
<td><strong>Save Snapshot Image</strong>&lt;br&gt;Saves an image of the current position of all items on the workflow design area in PNG format.</td>
</tr>
<tr>
<td><img src="image" alt="Toggle Transition Labels" /></td>
<td><strong>Toggle Transition Labels</strong>&lt;br&gt;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.</td>
</tr>
</tbody>
</table>

### Using the Statuses panel

The **Statuses** panel lists all available JIRA statuses that have not already been associated with a step on the workflow design area. It also provides tools for adding new statuses or editing existing ones.

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Refresh List" /></td>
<td><strong>Refresh List</strong>&lt;br&gt;Refreshes the statuses list.</td>
</tr>
<tr>
<td><img src="image" alt="Add New Status" /></td>
<td><strong>Add New Status</strong>&lt;br&gt;Adds a new global status to JIRA. See <a href="#">Defining Status field values</a> for details.</td>
</tr>
<tr>
<td><img src="image" alt="Status Editor" /></td>
<td><strong>Status Editor</strong>&lt;br&gt;Opens the Status Editor dialog box where you can edit an existing global status in JIRA. See <a href="#">Defining Status field values</a> for details.</td>
</tr>
</tbody>
</table>
You can use the **Statuses** panel to add a step to your workflow.

**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 only requires you to edit or update this transition in one place.

- **Note:**
  - Global Transitions are similar to – but not the same as – **common transitions**.
  - To avoid clutter, global transitions do not appear on the workflow design area. Instead, they display in the **Global Transitions** panel as well as every step of your workflow in **Text** edit mode.
  - Global Transitions can only be edited/deleted in **Diagram** mode. They cannot be configured in **Text** mode.

The **Global Transitions** panel also provides tools for adding new global transitions, as well as editing or deleting existing ones.

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Add New Transition" /></td>
<td>Adds a new global transition to the workflow.</td>
</tr>
<tr>
<td><img src="image" alt="Edit Selected Transition" /></td>
<td>Edits the currently selected global transition in the <strong>Global Transitions</strong> panel.</td>
</tr>
<tr>
<td><img src="image" alt="Delete Selected Transition" /></td>
<td>Deletes the currently selected global transition in the <strong>Global Transitions</strong> panel, from the workflow.</td>
</tr>
</tbody>
</table>

**Editing or deleting annotations**

Once you have added an annotation using the **Create Annotation** tool (above), you can do the following:

- To resize an annotation, drag the edges or corners of the **sticky note**.
- To edit the text inside an annotation, click inside the annotation and begin modifying the text.
- To remove an annotation, click the **X** at the top-right of the sticky note.

**Configuring workflow schemes**

A workflow scheme defines a set of associations – or mappings – between a workflow and an issue type. Workflow schemes are associated with a project and make it possible to use a different workflow for every combination of project and issue type.

By default, projects use JIRA’s **system workflow**. The default workflow scheme:

- Associates JIRA’s system workflow **jira** with all issue types (available to the JIRA project).
- Appears as **Default Workflow Scheme** in JIRA (or just **Default** in the context of workflow scheme selection dropdown menu).

This page describes how to configure workflows and issue type workflow associations in the scheme.

To associate a workflow scheme with a project (part of activating a workflow), see **Activating workflow**.
Adding a workflow scheme

1. Log in as a user with the JIRA Administrators global permission.
   ✓ Keyboard shortcut: g + g + start typing workflow schemes
3. Click the Add Workflow Scheme button.
4. Enter the Name and Description of the new workflow scheme.
5. Click the Add button. The new workflow scheme is created.
6. Follow the instructions in Configuring workflows for a workflow scheme below.

Configuring workflows for a workflow scheme

If your scheme is associated with a project, follow the instructions in Configuring a workflow scheme associated with a project. Otherwise, follow the instructions in Configuring a workflow scheme outside of a project.

Configuring a workflow scheme associated with a project

JIRA’s default workflow scheme cannot be modified. If you attempt to modify it, a copy of the scheme is created with the name of the project you are administering. You cannot configure a workflow scheme shared by multiple projects using this method; follow the instructions in Configuring a workflow scheme outside of a project instead.

To configure the workflow scheme associated with a project:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🌐 > Projects.
   ✓ Keyboard shortcut: g + g + start typing projects
3. Click Workflows on the left of the Project Summary page. The Workflows page is displayed, indicating the current workflow scheme used by the project.
4. Configure the issue types for the workflow scheme as desired.
   ✓ This is not the same as editing the workflow (clicking the Edit button in the workflow diagram at the center of your screen). If you do that you will be asked to Publish your draft workflow scheme.
5. At the Publish Workflows screen, click Associate to begin the migration process. Each issue has to be in a valid status. The valid statuses for an issue are defined by its workflow. This means that when changing a workflow, you may need to tell JIRA the status for specific issues after the change.
6. A screen displays that indicates the progress of migrating all the project’s issues to the updated scheme’s workflows.
7. Click Acknowledge to finish the process. A message displays letting you know that ‘your workflows have been published.’

### Operation Instructions

<table>
<thead>
<tr>
<th>Operation</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a workflow to the scheme</td>
<td>1. Click Add Workflow and select Choose From Marketplace or Add Existing.</td>
</tr>
<tr>
<td></td>
<td>* Choose from Marketplace lets you import a workflow from the Atlassian Marketplace. For more information, see the section called ‘Importing a workflow from Atlassian Marketplace’ in Sharing your workflow.</td>
</tr>
<tr>
<td></td>
<td>2. Select the desired workflow and issue types.</td>
</tr>
</tbody>
</table>
### Edit a workflow
Hover over the desired workflow and click the **Edit** button. See Configuring Workflow for further instructions.

**Note:** The **Edit** button only displays if you have edit permission. You can only edit the **system workflow** if you are viewing it in JIRA's default workflow scheme.

### Remove a workflow from the scheme
Click the cog icon ⚙️ for the desired workflow and select **Remove this workflow**.

### Change the issue types associated with a workflow
1. Click the **Assign** link under **Issue Types** for the desired workflow.
2. Select the desired issue types in the dialog that appears.
3. Click **Finish**.

### View the text-based representation of a workflow
Hover over the desired workflow and click the **View as Text** link.

### Change the workflow scheme associated with the project
Click the **Switch Scheme** button next to the scheme name. See Activating workflow for further instructions.

### Configuring a workflow scheme outside of a project
You can use this procedure to edit any workflow scheme in the system, including those shared by multiple projects. The workflow scheme can be either active or inactive.

**Before you begin:**
- If your workflow scheme is associated with a project, you may want to follow the instructions above instead. When a workflow scheme is used by more than one project, you must use this configuration method.
- When a workflow scheme is active, it creates a **draft workflow scheme** when you edit it.

**To configure the workflow scheme that is not associated with a project:**
1. Log in as a user with JIRA Administrators global permission.
   
   Choose ⚙️ > Issues. Select **Workflow Schemes** to open the Workflow Schemes page.

   **Keyboard shortcut:** g + g + start typing workflow schemes
2. Click the **Edit** link under the **Operations** column for the desired workflow.
3. Edit your workflow scheme as described in the table below.
4. If your workflow is active, you need to publish it to make your changes active.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Instructions</th>
</tr>
</thead>
</table>
| Add a workflow to the scheme | 1. Click **Add Workflow** and select Choose From Marketplace or Add Existing.  
   - **Choose from Marketplace** lets you import a workflow from the Atlassian Marketplace. For more information, see the section called "Importing a workflow from Atlassian Marketplace" in Sharing your workflow.  
   2. Select the desired workflow and issue types. |
| Remove a workflow from the scheme | Click the **Remove** link in the Operations column. |
| Change the issue types associated with a workflow | 1. Click the **Assign** link under **Issue Types** for the desired workflow.  
   2. Select the desired issue types in the dialog that appears.  
   3. Click **Finish**. |
<table>
<thead>
<tr>
<th>Operation</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit the name and description of a workflow scheme</td>
<td>Click the Edit link. Use inline edit mode – click in the associated field – to update the name and description.</td>
</tr>
<tr>
<td>Copy a workflow scheme</td>
<td>Click the Copy link. A copy of the workflow scheme is created with the prefix &quot;Copy of (name of current workflow)&quot; and placed in the inactive workflow schemes.</td>
</tr>
<tr>
<td>Delete a workflow scheme</td>
<td>Click the Delete link and confirm the deletion.</td>
</tr>
<tr>
<td></td>
<td>i You cannot delete an active workflow scheme. You must first disassociate it from all projects.</td>
</tr>
</tbody>
</table>

### Working in text mode

Text mode is an advanced way of working with workflows, and it shows the difference between steps and statuses. In text mode, you work directly with steps.

For all of the following procedures, you must be logged in as a user with the JIRA Administrators global permission and start from the Workflows page.

#### To access the workflows page:

Choose 🚀 > Issues. Select Workflows to open the Workflows page, which displays all of the workflows in your system.

#### Text edit mode:

Make sure the Text button is selected, so that your workflow appears in Text edit mode. A list of existing steps that comprise the workflow and each step's Linked Status and Outgoing Transitions (under Transitions (id)), is shown.

You should be starting from a screen that looks like this:
Basic procedures

**Editing a step**

To edit an existing step in a workflow:

Click the following link of any step:

- **Add Transition** — to add an *Outgoing Transition* to that step.
- **Delete Transitions** — to delete one or more *Outgoing Transitions* of that step.
- **Edit** — to edit the step's *Step Name* or *Linked Status*.
- **View Properties** — to view and edit the step's *Properties*.
- **Delete Step** — only available if the step has no *Incoming Transitions*.

**Adding a step**

The *Add New Step* form appears below the list of steps when you are editing an inactive workflow.

To add a new step to a workflow:

1. In the *Step Name* field, type a short name for the step.
2. In the *Linked Status* field, select the status that corresponds to this step.  
   Each status can only correspond to one step in each workflow.
3. Click the *Add* button. Your new step appears in your workflow's list of steps in *Text* edit mode.

If you do not see Add New Step, this means that all available statuses defined in your JIRA installation have been used in your workflow and you need to *define a new status*.

---

**Deleting a step**
A step can only be deleted if it has no incoming transitions.

To delete a step from a workflow:

Click the **Delete Step** link that corresponds to the relevant step.

*This link is not displayed if the step has no incoming transitions or if it only has incoming **Global Transitions**.*

**Adding a transition**

To add a transition to a workflow:

1. 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.

![Add Workflow Transition](image)

2. In the **Transition Name** field, type a short name for the transition.
   
   *This name will be shown to users on the relevant transition button on the View issue page.*

3. *(Optional)* In the **Description** field, type a short description of the purpose of the transition.

4. In the **Destination Step** field, choose the step to which issues will move when this transition is executed.

5. 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 wish to **create a new screen** for the transition.

**Editing or deleting a transition**

To edit or delete an existing transition of a workflow:

1. In the **Transitions (id)** column, click the link of the **Outgoing Transition** of the step you wish to edit. The **Transition** page is displayed.
2. From this point, you can:
   - Click the buttons at the top of the page to Edit or Delete the transition. 
     Note: You will only be able to delete a transition if this step has at least one outgoing transition indicated in the Workflow Browser section. In the image above, this is not the case.
   - Click View Properties to edit the transition’s properties. See Advanced workflow configuration for details.
   - Add a new condition, validator, or post function. See Advanced workflow configuration for details.

Advanced procedures

Preventing issues from being edited

You can use a workflow step’s properties to prevent issues from being edited in a particular workflow step. For example, in a copied system workflow, Closed issues cannot be edited, even by users who have the Edit Issue project permission.

Note:
- Issues that cannot be edited cannot be updated using Bulk edit.
- 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).

To stop issues from being editable in a particular workflow step or to set any property of a step:

1. Click the View Properties link that corresponds to the relevant step.
2. In the Property Key field, type: jira.issue.editable (or any other Property Key you wish to add).
3. In the Property Value field, type: false (or any other Property Value you wish to add).
4. Click the Add button.

Note:
- It is not possible to edit a step's properties on this page. To change any property's key or value, you must first delete the property you wish to change and then add the new, updated property.
- It is possible to implement restrictions on steps using step properties. For more information, see Workflow properties.

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 moves the issue to a Closed status. To do this:

1. Create or edit your transition.
2. Select the **Resolve Issue Screen** in the **Transition View** field.

![Add Workflow Transition](image)

3. Click **Add** when you are finished editing the workflow transition. You will be back on the **Text** view screen of the project's workflow.

**Sharing your workflow**

The new Workflow Sharing feature allows you to share your team's workflow with other teams in your organization on different JIRA instances, or external parties in other organizations via the Atlassian Marketplace. This feature allows you to easily share and use workflows that other people have published, or to move a workflow from staging to production in your own organization.

**On this page:**
- Exporting your workflow
- Uploading to Atlassian Marketplace
- Importing from Atlassian Marketplace
- Importing from a local instance
- Custom fields in workflow imports

**Exporting your workflow**

If you wish to share your JIRA Workflow with another instance of JIRA or upload it to the Atlassian Marketplace, you first need to download it. Follow this procedure.

1. Choose ![issues](image) > Issues.
2. Find the workflow you wish to share by clicking on the **Workflows** section in the left-hand panel.
3. Click **View** or **Edit** under the Operations column.
4. Select **Export** > **As Workflow**. You will see this screen:
Export Workflow screen
5. Click Next to continue.
6. In the Add Notes field add any special configuration notes; for example, information about plugins that should be installed. JIRA auto-populates these notes for you when it discards parts of your workflow (for example, plugins, post functions, conditions, validators).
7. Click Export and select a download location. Ensure the location is publicly accessible.

Uploading to Atlassian Marketplace
To share your workflow with other JIRA users, upload it to the Atlassian Marketplace.

1. Create an account on Atlassian Marketplace.
3. Click Create new add-on.
4. Choose My add-on is not directly installable.
5. Ensure Add-on Type is listed as 'Not a Plugin.'
6. You will need to host the workflow on your own servers and add information about where the workflow export can be accessed in the Binary URL textbox. This should be the location you specified in step 7 of the prior instruction set.
7. When you fill out the submission form, be sure to note the following:
   a. The Summary field contains the information that will be displayed to users searching the Marketplace.
   b. The Category for your workflow must be Workflow Bundles.
   c. The Add-on Key must be unique.

You don't have to complete the form in one session. You can save your form and come back to it later. Once you accept The Atlassian Marketplace Publisher Agreement, the system submits your add-on for review by Atlassian's Developer Relations team.

Importing from Atlassian Marketplace
This procedure covers importing a workflow from Atlassian Marketplace.

2. Click on the Workflows section in the left-hand panel.
3. Select Import > Import Workflow in the top right of the screen.
4. The From Atlassian Marketplace option should be selected by default, this is the screen you will see:
This screen displays the available workflows, ordered by popularity (determined by the number of downloads).

5. Find the workflow you want and click the Select button.
6. Follow steps 5 through 8 of the Importing from a local instance procedure.

Importing from a local instance

This procedure covers importing a workflow from a local instance. For importing from Marketplace, see the procedure above, Importing from Atlassian Marketplace.

You must be logged in as System Administrator to perform this function.

2. Click on the Workflows section in the left-hand panel.
3. Select Import > Import Workflow.

Import Workflow option

4. Select a workflow from your computer to upload and then click Next.
5. JIRA automatically generates a workflow name, but you can change this if you like. Click Next.
6. Next, you are presented with a screen that details your workflow statuses, as shown below. You can map the steps of the workflow to your existing workflow statuses or create new statuses at this point. When
you are finished, click **Next** to continue.

![Map Workflow Statuses screen](image)

7. You will be presented with a screen that presents a summary of the workflow changes, as shown below. Click **Import** at the bottom of this screen to accept these changes and import the workflow.
8. Your workflow is imported and you are presented with a screen with additional configuration details. Click **Done** to exit this process.

Custom fields in workflow imports

If the workflow that you are importing contains custom fields that are disabled, the workflow importer will not create these fields unless they are enabled before importing. You will receive a warning about this. To fix this, you need to enable the missing custom fields before proceeding with the import.

1. Click on the highlighted **Custom Field Types & Searchers** plugin in the displayed warning. This opens the plugin in a new window and scrolls to the right place to make the necessary changes:
2. Click to expand the list of enabled modules.
3. Find the modules that are disabled and enable them.

After enabling the corresponding modules of the Custom Field Types & Searchers plugin, return to the summary page and proceed. You may need to refresh the page first.

For information on installing add-ons, see Viewing Installed Add-ons.

Advanced workflow configuration

This section covers advanced topics on workflow transitions, including built-in JIRA conditions, combining conditions into groups, and applying validators and post functions. These are topics that are best explored once you've mastered the basics. To get started with workflow configuration, see Configuring Workflow.

Recap: What is a transition?

A transition is a link between two statuses that enables an issue to move from one status to another. In order for an issue to move between two statuses, a transition must exist.

A transition is a one-way link, so if an issue needs to move back and forth between two statuses, two transitions need to be created. The available workflow transitions for an issue are listed on the View issue screen, shown here.
What are conditions, validators, and post functions?

Before we get into the details, let's cover a few basics.
Workflow transition screen, displaying the statuses that use this transition (at the top) as well as the conditions available (at the bottom)

<table>
<thead>
<tr>
<th>JIRA term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Conditions</strong></td>
<td>Conditions control who perform a transition and under what circumstances. They can take into consideration a user's project permissions, the status of any subtasks, as well as the state of source code associated with issues. If a condition fails, the user will not see the transition link on the View issue page.</td>
</tr>
<tr>
<td><strong>Validators</strong></td>
<td>Validators check that any input made to the transition is valid before the transition is performed. If a transition's validator fails, the transition's post functions are not executed. When this happens the issue does not progress to the destination status of the transition. JIRA ships with a few default validators that can be added immediately to any transition.</td>
</tr>
</tbody>
</table>
| **Post functions** | Post functions carry out any additional processing required after a transition is executed, such as:  
  - updating an issue's fields  
  - generating change history for an issue  
  - adding a comment to an issue  
  - generating an event to trigger email notifications  
  There are two categories of post functions in JIRA — essential and optional — that are described later in this page. |

**How do conditions differ from validators?**

- **Conditions** are used to determine whether an issue's transition can be executed by a particular user. However, conditions cannot validate input parameters provided by the user on 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.

JIRA’s built-in conditions
JIRA includes the following individual conditions, which can be added immediately to any transition:

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
<th>Additional Parameters Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only Assignee Condition</td>
<td>Only allow the issue's current assignee to execute the transition.</td>
<td>None</td>
</tr>
<tr>
<td>Only Reporter Condition</td>
<td>Only allow the issue's reporter to execute the transition.</td>
<td>None</td>
</tr>
<tr>
<td>Permission Condition</td>
<td>Only allow users with a given permission to execute the transition.</td>
<td>A project-wide permission</td>
</tr>
<tr>
<td>Sub-Task Blocking Condition</td>
<td>Only allow a parent issue's transition to execute if all its sub-tasks have at least one of a specified set of statuses.</td>
<td>One or more statuses</td>
</tr>
<tr>
<td>User Is In Group</td>
<td>Only allow users in a given group to execute the transition.</td>
<td>A group</td>
</tr>
<tr>
<td>User Is In Group Custom Field</td>
<td>Only allow users in a given group-type custom field to execute a transition.</td>
<td>A custom field of type &quot;Group&quot;</td>
</tr>
<tr>
<td>User Is In Project Role</td>
<td>Only allow users in a given project role to execute a transition.</td>
<td>A project role</td>
</tr>
<tr>
<td>Code Committed Condition</td>
<td>Only allow the transition to execute if code either has or has not been committed against the issue.</td>
<td>Specify whether or not code must be committed</td>
</tr>
<tr>
<td>No Open Reviews Condition</td>
<td>Only allow the transition to execute if no related Crucible reviews are open against the issue.</td>
<td>None</td>
</tr>
<tr>
<td>Unreviewed Code Condition</td>
<td>Only allow the transition to execute if no unreviewed changesets related to the issue exist.</td>
<td>None</td>
</tr>
</tbody>
</table>

You can also create your own conditions via the plugin system. See the Workflow Plugin Modules for details.

Adding a condition

This procedure documents adding a condition using the Diagram edit mode. For text mode, see this procedure instead.

To add a condition to a transition using diagram edit mode:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose ‣ Issues. Select Workflows to open the Workflows page, which displays all of the workflows in your system.
3. Click the Edit link next to the workflow with the transition you wish to change.
4. Select the transition in the diagram, as shown here:
5. Click on **Conditions** to open this screen:

6. Click **Add condition** to display a list of conditions that can be added to the transition. Select a condition from the list and click **Add**.

7. If required, select parameters to add to the condition (as shown here). When finished, click **Add**.

Your new condition is displayed on the **Workflow transition screen**:
To add a condition to a transition using text edit mode:

1. Follow steps 1 through 3 as outlined above.
2. In the Transitions (id) column, click the name of the relevant transition. The Workflow transition screen is displayed.
3. Click the Conditions tab (if not already selected) to see a list of the transition's existing conditions.
4. Follow steps 6 and 7 as outlined above.

Combining conditions into groups

You can construct complex conditions by combining two or more individual conditions using the procedure above, or by creating a grouped condition, as described below. You can also combine multiple grouped conditions by nesting grouped conditions. A group can be configured to be successful either when all the conditions in the group are successful (use All of the following conditions ) or when at least one of the conditions in the group are successful (use Any of the following conditions ).

Converting an individual condition into a grouped condition

Use this procedure to create grouped conditions.

To convert an individual condition into a grouped condition:

1. Follow the Adding a condition procedure to step 3.
2. Select the Add grouped condition icon for the individual condition you wish to change, as shown here:

3. Select a new condition and click Add.
4. If required, select parameters to add to the condition (as shown here). When finished, click Add.

Your grouped condition is displayed as follows:
Switching a grouped condition’s logic

The logic of all individual conditions within a grouped condition can be switched between All of the following conditions and Any of the following conditions. To do this, switch the link (as shown below) for any condition you wish to change.

If you need to switch the logic of some, but not all, individual conditions within a grouped condition, create a nested grouped condition (as described above) instead.

Adding a validator

This procedure documents adding a validator using the Diagram edit mode. For text mode, see this procedure in stead.

To add a validator to a transition:

1. Follow the Adding a condition procedure to step 4.
2. At step 5, click Validators from the displayed Properties panel.
3. Click Add validator to display a list of validators that can be added to the transition. Select a validator from the list and click Add.
4. If required, select parameters to add to the validator. When finished, click Add.

Your validator is displayed as follows:

To add a validator to a transition using text edit mode:

1. Follow steps 1 through 3 from the Adding a condition procedure.
2. In the Transitions (id) column, click the name of the relevant transition. The Workflow transition screen is displayed.
3. Click the Validators tab to see a list of the transition’s existing validators.
4. Follow steps 3 and 4 as outlined directly above.

Post functions

- Essential post functions
Optional post functions
- The initial transition
- Adding a post function
- Using a post function to set a field
- Using a post function to send HipChat notifications
- Using a post function to send email notifications

**Essential post functions**

JIRA includes the following essential post functions, which are automatically added to every newly-created transition and performed in this order:

1. Set issue status to the linked status of the destination workflow status.
2. Add a comment to an issue if one is entered during a transition.
3. Update change history for an issue and store the issue in the database.
4. Reindex an issue to keep indices in sync with the database.
5. 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 between them.

**Optional post functions**

JIRA includes optional post functions (described below), which can be added to transitions. There may be additional post functions available, depending on your particular JIRA installation and installed plugins. Additionally, you can create your own post functions using the plugin system; see the Workflow Plugin Modules for details.

<table>
<thead>
<tr>
<th>Optional post function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign to Current User</td>
<td>Assigns the issue to the user who is executing the transition.</td>
</tr>
<tr>
<td><strong>Note:</strong> This post function is ignored unless the user has the Assignable User permission. Create a condition to give the logged-in user this permission before executing the transition.</td>
<td></td>
</tr>
<tr>
<td>Assign to Lead Developer</td>
<td>Assigns the issue to the component lead, if one exists, or project lead.</td>
</tr>
<tr>
<td>Assign to Reporter</td>
<td>Assigns the issue to the user who created the issue.</td>
</tr>
<tr>
<td>Create Perforce Job Function</td>
<td>Creates a Perforce Job (if required) after completing the workflow transition.</td>
</tr>
<tr>
<td>Notify HipChat</td>
<td>Sends a notification to one or more HipChat rooms. See Using a post function to send HipChat notifications for more information. Before you can use this post function, you must first have configured your JIRA installation with HipChat. See Configuring JIRA with HipChat for more information.</td>
</tr>
<tr>
<td>Trigger a Webhook</td>
<td>Triggers the specified webhook after completing the workflow transition.</td>
</tr>
<tr>
<td><strong>Note:</strong> When you add this post function, you will be asked to specify a webhook. This webhook must already be defined in JIRA (see Managing Webhooks).</td>
<td></td>
</tr>
</tbody>
</table>
**Update Issue Field**

Updates one of the issue's fields to a given value. Fields that can be updated include:
- Assignee
- Description
- Environment
- Priority
- Resolution
- Summary
- Original Estimate
- Remaining Estimate

**Note:** This post function cannot update custom fields and must be positioned after the other optional post functions.

---

**The initial transition**

You can add post functions to a workflow's initial transition in cases where you need to perform specific processing tasks – such as setting a particular field's value – when creating an issue. The *initial transition* is called Create (if you created a blank workflow) or Create Issue (if you copied the system workflow).

JIRA includes the following essential post functions that are specific to a workflow's initial transition. These post functions are automatically added to the initial transition of each newly-created workflow (and performed in this order):

**Essential post function (initial transition only)**
- Creates the issue
- Fires an event that can be processed by the listeners

The optional post functions listed above can be added to a workflow's initial transition, in addition to the following post functions.

<table>
<thead>
<tr>
<th>Optional post function (initial transition only)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Comment</td>
<td>Adds a comment to an issue if one is entered during a transition.</td>
</tr>
<tr>
<td>Update Issue Status</td>
<td>Sets the issue's status to the linked status of the destination workflow status.</td>
</tr>
<tr>
<td>Store Issue</td>
<td>Stores updates to an issue (no change history is created).</td>
</tr>
</tbody>
</table>

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), must be placed before the Creates the issue originally post function. If you wish to, you can configure the initial status for your workflow to go to a different initial transition. See Configuring the initial status for details.
Special Cases

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 the 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 (for example, Version or Component)

Adding a post function

This procedure documents adding a post function using the Diagram edit mode. For text mode, see this procedure instead.

To add a post function to a transition:

1. Follow the Adding a condition procedure to step 4.
2. At step 5, click Post Functions from the displayed Properties panel.
3. Click Add post function to display a list of post functions that can be added to the transition. Select a post function from the list and click Add.
4. If required, select parameters to add to the validator. When finished, click Add.
5. The Post Functions tab is displayed again, showing your new post function in the list. 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 Text edit mode to add the post function to the transition:

1. Follow steps 1 through 3 from the Adding a condition procedure.
2. In the Transitions (id) column, click the name of the relevant transition. The Workflow transition screen is displayed.
3. Click the Post Functions tab to see a list of the transition’s existing post functions.
4. Follow steps 3 and 5 as outlined directly above.

Using a post function to set a field
You can use an **Update Issue Field** post function to set the value of an issue's field after a particular transition is executed.

For a particular step in a workflow, you might need to create a transition that moves the issue to a *closed* status. As part of this transition, you might want to automatically set the **Resolution** field, as described below.

**Using a post function to set the Resolution field (diagram mode):**

1. Log in as a user with the **JIRA Administrators** global permission.
2. Choose > Issues. Select **Workflows** to open the Workflows page, which displays all of the workflows in your system.
3. Click the Edit link next to the workflow you wish to edit.
4. Drag a port of any status to a port of another status (see the illustration below) to create a new transition between the two statuses.

5. In the **Add Transition** dialog box, select either None or a screen that does not contain the **Resolution** field.

6. **Add a new post function** of type **Update Issue Field** and do the following:
   a. Select **Resolution** from the **Issue Field** select list.
   b. Select a suitable resolution from the **Field Value** select list.

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, select None from the **Field Value** select list.

The list of post functions for this transition includes 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 HipChat notifications

You can use a Notify HipChat post function to send a notification to one or more HipChat rooms whenever an issue passes through a transition with this post function. You can also apply a JQL query to filter issues in your Notify HipChat post function, so that if an issue transitions with this post function, *but* its JQL query does not retrieve that issue, the HipChat notification will not be sent.

Before you can use the Notify HipChat post function, you must first have configured your JIRA installation with HipChat. See Configuring JIRA with HipChat for more information.

To do this:

1. Create or edit your transition.
2. Add a new post function of type Notify HipChat.
3. Do the following on the Add Parameters to Function page:
   a. Specify an optional JQL query in the field provided if you want to send notifications to issues which not only pass through this transition *but are also* retrieved by this query.
   b. Leave this field empty to send notifications to all issues that pass through this transition.
   c. Select the HipChat Rooms you want to link with your workflow transition.

Using a post function to send email notifications

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've 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

Use the Generic Event to send email notifications.

To do this:

1. Create or edit your transition.
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 can be used to further customize transitions. For example, transition properties help to extend a copied system workflow to allow language translations.

To view and edit the properties of a transition in diagram mode:
1. Select a transition in the diagram.
2. Click **Properties** option in the Properties panel to display the **View Workflow Transition Properties** screen shown here:

![View Workflow Transition Properties](image)

3. Do any of the following:
   - Add a new property to the transition.
   - Delete a property, by clicking the icon to the right of the property.

**Important**

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.

**Using Text edit mode to view or edit the properties of a transition**

1. In the **Transitions (id)** column of the displayed workflow, click the name of the relevant transition. The **Workflow transition** page is displayed.
2. Click the **View Properties** link at the top of the page.

It is possible to implement restrictions on transitions using transition properties. For more information, see **Workflow properties**.

**Customizing workflow transitions**

When viewing an issue, most of the operations and workflow transitions are available from a row of buttons at the top of the issue (see the illustration at the top of this page).

By default, the first two transitions appear as separate buttons in the set of transition buttons. Additional transitions appear in the **Workflow** button dropdown menu. The order in which these buttons appear is based on...
the order defined in the system workflow.

**To change the number of transition buttons from the default of two:**

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**

To change the order of transition buttons, including additional transitions in the Workflow menu, add the property key `opsbar-sequence` to each workflow transition that you wish to reorder. 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.
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.

**Note:** Be aware that adding the `opsbar-sequence` property to a workflow transition does not change the order of these transitions in the workflow in Text edit mode. The addition of this property only affects the order of transitions on the View issue page.

**Working with common and global transitions**

A **common transition** is one 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.

**Global transitions** are ones that can be used by any status in a workflow, as shown here:
Known Limitations

- You can add a global transition from the workflow designer, but not a common transition. To add a common transition, you need to switch to Classic mode workflow designer.
- Global transitions can only be deleted from the workflow designer, not in text mode.
- You cannot perform the Issue editable function, which is only available in the classic mode designer (shown at right).
- You cannot create a global transition without a result step.

Common transitions

To add a common transition, you need to use the Classic mode workflow designer. Switch into this mode by clicking the Enter classic mode link in your workflow diagram.

1. Create a transition using the classic mode toolbar options.
2. When you are presented with the Add Transition dialog, select common transition and the transition to reuse:
You can turn an ordinary transition into a common transition in as follows:

1. Create a transition from one status to another (a destination status) as described in Adding a transition.
2. Create another transition from a different status to the same destination status used in the first step of this procedure.
   - In the Add Transition dialog, choose Use Common Transition.
3. In the Transition To Reuse dropdown menu, select the transition you created in step 1 of this procedure.
4. Click OK.

**Global transitions**

You can add a global transition as part of the status creation step or when adding an existing status.

**Creating a new status with a global transition:**

1. Click Create new status.
2. In the displayed dialog, enter the status name, description, and check the Add global transition to status option.
3. Click Create. Your global transition and new status are available immediately:
You can also add a global transition this way:

1. Select a status, for example, **Closed**.
2. Select the **Add a global transition** option from the Properties panel, as shown here:

![Add Global Transition Dialog](image)

3. The **Add Global Transition** dialog is displayed. Enter a name and description, select a transition screen, and click **Create**.

Adding a custom event

JIRA uses an event-listener mechanism to alert the system that something has happened, and to perform 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).

**On this page:**
- Overview
- System Events
- Custom Events
- Configuring Notifications for a Custom Event
- Adding a custom event
• Configuring the notification scheme to send mail
• Configuring a post function to fire the custom event

Overview

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).

System Events

JIRA’s built-in system events are:

<table>
<thead>
<tr>
<th>Event Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue Created:</td>
<td>An issue has been entered into the system.</td>
</tr>
<tr>
<td>Issue Updated:</td>
<td>An issue has had its details changed.</td>
</tr>
<tr>
<td>Issue Assigned:</td>
<td>An issue has been assigned to a new user.</td>
</tr>
<tr>
<td>Issue Resolved:</td>
<td>An issue has been resolved (usually after being worked on and fixed).</td>
</tr>
<tr>
<td>Issue Closed:</td>
<td>An issue has been closed. (Note that an issue may be closed without being resolved; see Statuses ).</td>
</tr>
<tr>
<td>Issue Commented:</td>
<td>An issue has had a comment added to it.</td>
</tr>
<tr>
<td>Issue Comment Edited:</td>
<td>An issue’s comment has been modified.</td>
</tr>
<tr>
<td>Issue Reopened:</td>
<td>An issue has been re-opened.</td>
</tr>
<tr>
<td>Issue Deleted:</td>
<td>An issue has been deleted.</td>
</tr>
<tr>
<td>Issue Moved:</td>
<td>An issue has been moved into this project.</td>
</tr>
<tr>
<td>Work Logged On Issue:</td>
<td>An issue has had hours logged against it (i.e. a worklog has been added).</td>
</tr>
<tr>
<td>Work Started On Issue:</td>
<td>The Assignee has started working on an issue.</td>
</tr>
<tr>
<td>Work Stopped On Issue:</td>
<td>The Assignee has stopped working on an issue.</td>
</tr>
<tr>
<td>Issue Worklog Updated:</td>
<td>An entry in an issue's worklog has been modified.</td>
</tr>
<tr>
<td>Issue Worklog Deleted:</td>
<td>An entry in an issue's worklog has been deleted.</td>
</tr>
<tr>
<td>Generic Event:</td>
<td>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 ).</td>
</tr>
</tbody>
</table>

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.

**Adding a custom event**

1. Log in as a user with the JIRA Administrators global permission.
2. Choose `>` System. Select 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.
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.

**Configuring the notification scheme to send mail**

1. Log in as a user with the JIRA Administrators global permission.
2. Choose `>` Issues. Select Notifications Schemes to open the Notification Schemes page.
   - Keyboard shortcut: `g + g + start typing notification schemes`
3. Select the notification scheme to edit, by clicking the notification scheme’s name or its Notifications link (under Operations).
4. Add the recipients for the custom event as required. See Creating a Notification Scheme for more information.

**Configuring a post function to fire the custom event**

1. Log in as a user with the JIRA Administrators global permission.
2. Choose `>` Issues. Select Workflows to open the Workflows page, which displays all of the workflows in your system.
   - Keyboard shortcut: `g + g + start typing workflows`
3. Navigate to workflow transition post-function screen to be edited. See Configuring Workflow and Applying Post Functions to Transitions for more information.
4. Update the post function to fire the custom event.
5. Activate or associate the workflow (and scheme) with the appropriate project. See Activating workflow for more information.

- Overview
- System Events
- Custom Events

**Configuring Notifications for a Custom Event**

- Adding a custom event
- Configuring the notification scheme to send mail
- Configuring a post function to fire the custom event

**Configuring the initial status**
Use this procedure to configure the initial status for your workflow. You can start off with an active workflow, which you can then switch to draft mode, or any other workflow in your system.

1. Click on **Open** under the Step Name column to view or edit a step's properties:

   ![Workflow Diagram](image1)

   **Note:** If you happen to be in an active workflow, which you cannot edit, you will be asked to switch to a draft workflow to continue.

2. Click on the **Create Issue** incoming transition:

   ![Transition Diagram](image2)

   **Note:** If you happen to be in an active workflow, which you cannot edit, you will be asked to switch to a draft workflow to continue.

3. Click **Edit** to set the new destination step:

   ![Edit Transition](image3)

4. Select a new **Destination Step**, and then click **Update** to save it:
Now, when a new issue is created, it goes straight to the In Progress step, as shown here:

Configuring JIRA with HipChat

If your JIRA site has been configured with HipChat, you can:

- Generate a HipChat notification when an issue goes through a particular workflow transition. To do this, add a post function to the workflow transition you want to notify users about.
- Apply JQL criteria to a post function to fine tune when a notification is sent as a result of a workflow transition being executed.

Before you can configure HipChat notifications in JIRA, you need to configure JIRA with a HipChat API Auth Token of type Admin, which is generated through (and associated with) a specific HipChat account. This allows JIRA to send notifications to HipChat rooms associated with this HipChat account.

Generating a HipChat API Auth Token of type Admin requires a HipChat account with admin-level access.

To configure your HipChat API Auth Token in JIRA:

1. Visit the HipChat API Auth Token page to generate an admin-level token.
2. Log in as a user with the JIRA Administrators global permission.

Keyboard shortcut: g + g + start typing hipchat configuration
4. Copy your HipChat API Auth Token and paste it into the **Admin Token** field.

5. Click the **Save** button.

**Please Note:**
- If you receive a message indicating that your HipChat configuration was saved successfully, you can proceed to **add HipChat notifications to your workflow transition's post functions**.
- The HipChat rooms that JIRA has access to are those rooms configured through your admin-level HipChat account.

### Using validators with custom fields

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 JIRA Agile 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.

### 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, 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. Choose **Issues > Workflows** to open the Workflows page, which displays all of the workflows in your system.

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 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 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. 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.

### Workflow properties

You can use workflow properties to implement restrictions on certain steps or transitions of a workflow (below).

#### 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 at your own risk!**

For details on how to implement workflow properties (i.e. step and transition properties) in your workflow, please refer to **Configuring Workflow**.

### 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 API Documentation - JiraWorkflow constant values](#)

<table>
<thead>
<tr>
<th>Name</th>
<th>Values</th>
<th>Related Issues</th>
<th>References</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>jira.field.resolution.exclude</code></td>
<td>Resolution id</td>
<td></td>
<td>Resolutions per workflow step</td>
<td>Add comma-separated resolution ids to the transition properties where you want to not show certain resolutions</td>
</tr>
<tr>
<td><code>jira.field.resolution.include</code></td>
<td>Resolution id</td>
<td>JRA-16443</td>
<td>Resolutions per workflow step</td>
<td>Add comma-separated resolution ids to the transition properties</td>
</tr>
<tr>
<td><code>jira.i18n.submit</code></td>
<td>i18n property key</td>
<td>JRA-6798</td>
<td>Transition (usage: action submit button name)</td>
<td></td>
</tr>
<tr>
<td><code>jira.i18n.title</code></td>
<td>i18n property key</td>
<td>JRA-6798</td>
<td>Transition (usage: action name, etc.)</td>
<td></td>
</tr>
</tbody>
</table>
### Configuring Email

#### Configuring Email Notifications
- 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).

**Email notifications**

- **Enabling email notifications**
  - Configure JIRA's SMTP mail server to send notifications

- **Configure a notification scheme and associate it with the appropriate projects.**

- It is possible to customize 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.

### Configuring Workflow

<table>
<thead>
<tr>
<th>jira.issue.editable</th>
<th>true, false</th>
<th>Configuring Workflow</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira.permission.*</td>
<td>user1, user2 / group1, group2 / role/...?</td>
<td>JRA-6381 JRA-34621</td>
<td>- WorkflowBased PermissionManager class description (API documentation) - Permissions based on Workflow Status - For link permissions - How to deactivate comments for closed issues shows how to use jira.permission comment.user=denied - jira.permission.edit.group=jira-administrators means that only JIRA administrators can edit an issue (blog)</td>
</tr>
<tr>
<td>opsbar-sequence</td>
<td>Integer value greater than or equal to 0</td>
<td>Configuring Workflow (Customizing Transitions)</td>
<td>Transitions on the 'View Issue' page</td>
</tr>
</tbody>
</table>
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. Choose Projects. The 'Project Summary' page (see Defining a Project) for your selected project is shown.

3. 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.

4. 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.

Note: 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.

JIRA’s default setting is to not notify users of their own changes. This can be changed on a per user basis via their Profile Preferences.

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 by choosing the cog icon at top right of the screen, then User Management > User Preferences.

The HTML email format can accommodate internationalized 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

Troubleshooting email notifications

Using the JIRA admin helper

The JIRA admin helper can help you diagnose why a user isn't receiving email notifications when they should be, or why a user is receiving email notifications when they shouldn't be. This tool is only available to JIRA administrators.

To diagnose why a user is or is not receiving notifications for an issue:

1. View the issue in JIRA.
2. Click More Actions > Notification Helper.
3. Enter the username of the user.
4. Click Submit.

Tip: You can also access the Notifications Helper via the cog menu for each issue in the issue navigator, or by selecting the cog icon at top right of the screen, then Add-ons. Select Admin Helper > Notification Helper to open the following page.

Keyboard shortcut: g + g + start typing 'Notification Helper
Configuring JIRA’s SMTP Mail Server to Send Notifications

On this page:
- Define or edit the SMTP mail server
- Specify a host name or JNDI location for your SMTP mail server
  - Specify the SMTP host details
  - Specify a ‘JNDI Location’
- Configuring a JNDI location
- Troubleshooting

To enable JIRA to send notifications about various events, you need to first configure an SMTP mail server in JIRA.

**Define or edit the SMTP mail server**

1. Log in as a user with the JIRA System Administrators global permission.
2. Choose System > System. Select 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Specify an arbitrary name to identify this SMTP mail server configuration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>(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.</td>
</tr>
</tbody>
</table>
### 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), 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

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

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:

<table>
<thead>
<tr>
<th>Service Provider (not available when updating an existing SMTP mail server)</th>
<th>Choose between using your own SMTP mail server (i.e. <strong>Custom</strong>), or either Gmail (i.e. <strong>Google Apps Mail / Gmail</strong>) or Yahoo! (i.e. <strong>Yahoo! Mail Plus</strong>) as the service provider for your SMTP mail server. <strong>Tip:</strong> If you choose either Gmail or Yahoo! options and then switch back to <strong>Custom</strong>, some of the key fields in this section will automatically be populated with the relevant SMTP mail server settings for these service providers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>Choose between whether your SMTP mail server is a standard (i.e. <strong>SMTP</strong>) or a secure (i.e. <strong>SECURITY_SMTP</strong>) one.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Specify the hostname or IP address of your SMTP mail server. Eg. <code>smtp.yourcompany.com</code></td>
</tr>
<tr>
<td>SMTP Port</td>
<td><em>(Optional)</em> The SMTP port number, usually 25 for SMTP or 465 for SMTPS, either of which are assumed if this field is left blank.</td>
</tr>
<tr>
<td>Timeout</td>
<td><em>(Optional)</em> 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.</td>
</tr>
<tr>
<td>TLS</td>
<td><em>(Optional)</em> Select this check box if your SMTP host uses the Transport Layer Security (TLS) protocol.</td>
</tr>
<tr>
<td>Username</td>
<td><em>(Optional)</em> 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.)</td>
</tr>
<tr>
<td>Password</td>
<td><em>(Optional)</em> Again, if your SMTP host requires authentication, specify the password associated with the username you specified above. <strong>Tip:</strong> When editing an existing SMTP mail server, select the <strong>Change Password</strong> check box to access and change this field.</td>
</tr>
</tbody>
</table>

**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:

```xml
<Context path="" docBase="${catalina.home}/atlassian-jira" reloadable="false">
    ...
    <Resource name="mail/JiraMailServer" class="javax.mail.Session">
        <Parameter name="mail.smtp.host" value="mail.yourcompany.com"/>
        <Parameter name="mail.smtp.port" value="25"/>
        <Parameter name="mail.transport.protocol" value="smtp"/>
        <Parameter name="mail.smtp.auth" value="true"/>
        <Parameter name="mail.smtp.user" value="jirauser"/>
        <Parameter name="password" value="mypassword"/>
    </Resource>
    ...
</Context>
```

Or if you do not require authentication (e.g. if you are sending via localhost, or only internally within the company):

```xml
<Context path="" docBase="${catalina.home}/atlassian-jira" reloadable="false">
    ...
    <Resource name="mail/JiraMailServer" class="javax.mail.Session">
        <Parameter name="mail.smtp.host" value="localhost"/>
        <Parameter name="mail.smtp.port" value="25"/>
        <Parameter name="mail.transport.protocol" value="smtp"/>
    </Resource>
    ...
</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
java.lang.NoClassDefFoundError: javax/mail/Authenticator
```

or:

```java
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" type="javax.mail.Session">
  <mail.smtp.host>smtp.gmail.com</mail.smtp.host>
  <mail.smtp.port>465</mail.smtp.port>
  <mail.smtp.auth>true</mail.smtp.auth>
  <mail.smtp.user>myusername@gmail.com</mail.smtp.user>
  <password>mypassword</password>
  <mail.smtp.starttls.enable>true</mail.smtp.starttls.enable>
  <mail.smtp.socketFactory.class>javax.net.ssl.SSLSocketFactory</mail.smtp.socketFactory.class>
</Resource>

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 (e.g. 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 Mail Server 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

JIRA can generate email notifications for various events that happen throughout the lifecycle of an issue, including custom events. 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. This scheme is associated with all new projects by default. This means that if you have an outgoing (SMTP) mail server set up, that email notifications will be sent as soon as there is any activity (e.g. issues created) in the new project. However, you can disassociate this notification scheme from the project via the Project Summary page, as described below. You can also modify this scheme or if you prefer, create other notifications schemes for particular projects.

On this page:
- 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. Choose Issues > Notification Schemes to open the Notification Schemes page, which lists all the current notification schemes 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. Choose **Issues** > Issues. Select **Notification Schemes** to open the Notification Schemes page, which lists all the current notification schemes in your JIRA installation.

   ✓ **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**

<table>
<thead>
<tr>
<th>Event</th>
<th>Notifications</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue Created (System)</td>
<td>Current Assignee, Reporter, All Watchers</td>
<td>Add</td>
</tr>
<tr>
<td>Issue Updated (System)</td>
<td>Current Assignee, Reporter, All Watchers</td>
<td>Add</td>
</tr>
<tr>
<td>Issue Assigned (System)</td>
<td>Current Assignee, Reporter, All Watchers</td>
<td>Add</td>
</tr>
<tr>
<td>Issue Resolved (System)</td>
<td>Current Assignee, Reporter, All Watchers</td>
<td>Add</td>
</tr>
<tr>
<td>Issue Closed (System)</td>
<td>Current Assignee, Reporter, All Watchers</td>
<td>Add</td>
</tr>
<tr>
<td>Issue Comment Edited (System)</td>
<td>Current Assignee, Reporter, All Watchers</td>
<td>Add</td>
</tr>
</tbody>
</table>

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 as a user with the **JIRA Administrators** global permission.
2. Choose 

**Keyboard shortcut:** `g + g + start typing projects`

3. 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.

4. Click the **Actions** dropdown menu and choose **Use a different scheme** (or **Select a scheme**).

**Screenshot 4: The Project Notifications page**

5. 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).

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue Created:</td>
<td>An issue has been entered into the system.</td>
</tr>
<tr>
<td>Issue Updated:</td>
<td>An issue has had its details changed. This includes the deletion of an issue comment.</td>
</tr>
<tr>
<td>Issue Assigned:</td>
<td>An issue has been assigned to a new user.</td>
</tr>
<tr>
<td>Issue Resolved:</td>
<td>An issue has been resolved (usually after being worked on and fixed).</td>
</tr>
<tr>
<td>Issue Closed:</td>
<td>An issue has been closed. (Note that an issue may be closed without being resolved; see <strong>Workflow</strong>).</td>
</tr>
<tr>
<td>Issue Commented:</td>
<td>An issue has had a comment added to it.</td>
</tr>
<tr>
<td>Issue Comment Edited:</td>
<td>An issue's comment has been modified.</td>
</tr>
<tr>
<td>Issue Reopened:</td>
<td>An issue has been re-opened.</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th><strong>Recipient</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Assignee</td>
<td>The user to whom the issue is currently assigned.</td>
</tr>
<tr>
<td>Reporter</td>
<td>The user who originally created the issue.</td>
</tr>
<tr>
<td>Current User</td>
<td>The user who performed the action that has triggered this event.</td>
</tr>
<tr>
<td>Project Lead</td>
<td>The user who is managing the project to which the issue belongs.</td>
</tr>
<tr>
<td>Component Lead</td>
<td>The user who is managing the component to which the issue belongs.</td>
</tr>
<tr>
<td>Single User</td>
<td>A particular user in your JIRA system.</td>
</tr>
<tr>
<td>Group</td>
<td>A particular group in your JIRA system.</td>
</tr>
<tr>
<td>Project Role</td>
<td>The members of a particular project role for this project.</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>Single Email Address</td>
<td>Any email address that you wish to alert.</td>
</tr>
<tr>
<td></td>
<td>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). The text template is used for notifications to a single email address.</td>
</tr>
<tr>
<td>All Watchers</td>
<td>All users who are watching the issue.</td>
</tr>
</tbody>
</table>
### 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**).
- JIRA’s default setting is to not notify users of their own changes. This can be changed on a per user basis via their **Profile Preferences**.

**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 behavior to be re-instated, for those customers who prefer this behavior.

See JRA-6344 for more details.

### Customizing Email Content

JIRA generates emails in reaction to events using a templating engine. The templating engine is Apache’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.

**Please Note:**

- To change the columns in your filter subscriptions, you don't need to customize the mail templates. See **Customizing your Issue Navigator**.
- There’s a feature request to improve this at JRA-7266, which you can vote on to improve its chances of being implemented.
- Bear 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 Velocity templates (as well as JSPs) into the new installation of JIRA. If the Velocity templates and/or JSPs have changed in the newer version, you will have to manually port your customizations into them (as opposed to copying these files directly over from your old JIRA installation to your upgraded one).

Customizations to Velocity templates or other JIRA files are not included in the scope of Atlassian **Support**.

### Email template locations

To customize 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 favorite text editor. Referring to the JIRA template documentation (particularly Velocity Context for Email Templates) and Velocity Users Guide, make the customizations 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 customization

The Issue object is passed into the vm templates. Notice some of its 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.

Deploying Velocity templates without restarting JIRA

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. Remove the comment sign (#) from #velocimacro.library.autoreload=true

Making this change in production will eventually lead to JIRA not serving pages along with the ran out of parsers error in the log file.

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.

On this page:

- Configuring issue or comment creation from email
- Mail handlers
- Issue/comment creation
- Handy tips with mail handlers
- Best practices (pre-processing JIRA email messages)
- Troubleshooting

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.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.

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.

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. Choose System > System. Select Mail > Incoming Mail to open the Incoming Mail page.
   - Keyboard shortcut: g + g + start typing incoming mail
3. Click the Add incoming mail handler button (or the Edit link next to an existing mail handler) in the Mail Handlers section 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 `impo` subdirectory of the **JIRA Home Directory**, specify the subdirectory structure (within `impo rt/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. If you are using Local Files as the server, copy a saved email that contains a “Subject: ” line to the configured directory. JIRA will remove this file after it is parsed, or log a message about why an issue could not be created. You may have to specify the project, issuertype and reporterusername properties as a minimum configuration.

    A sample email file might look like this:
    ```
    To: jira@example.com
    From: some-jira-user@example.com
    Subject: (TEST-123) issue summary title here
    Body of the email goes here
    ```

    Click the **Add / Save** button to save your mail handler.

**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**

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:

<table>
<thead>
<tr>
<th>Project</th>
<th>Specify the project key of the default project to which new issues are created by this handler — for example, JRA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue Type</td>
<td>Choose the default issue type for new issues.</td>
</tr>
<tr>
<td>Strip Quotes</td>
<td>Select this check box to remove quoted text 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.</td>
</tr>
</tbody>
</table>

**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.
| **Catch Email Address** | If specified, only email messages whose **To:**, **Cc:** or **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:** or **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`) for multiple mail services (e.g. each one to create issues in separate JIRA projects).

**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 (e.g. 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.

**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.

**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 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) for multiple mail services (e.g. each one to create issues in separate JIRA projects).

   *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.  

> **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`  

> **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) for multiple mail services (e.g. each one to create issues in separate JIRA projects).

**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.

**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`

*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:
<table>
<thead>
<tr>
<th><strong>Project</strong></th>
<th>Specify the project key of the default project to which new issues are created by this handler — for example, JRA.</th>
</tr>
</thead>
</table>
| **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). |

**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. |

**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.

ℹ️ **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`.

ℹ️ **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.

ℹ️ **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:**; **Cc:** and **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 behavior 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).  

**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.  

**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`.

**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**

**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 autoresponses:
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). If you cannot resolve a problem yourself, please refer to the Getting Help page.

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 replies by email of JIRA’s notifications list JIRA’s SMTP server rather than the configured handler POP account (i.e. in Outlook’s ‘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 behavior. The email comment handler was designed to do plain text conversion.

**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 server as described below.

**Add or edit a POP or IMAP mail server**

1. Log in as a user with the JIRA Administrators global permission.
2. Choose ‣ > System. Select Mail > Incoming Mail to open the Incoming Mail 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:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>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.</td>
</tr>
<tr>
<td>Description</td>
<td>(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 &lt;Project&gt;’. This description appears below the Name of the POP / IMAP mail server on the POP / IMAP Mail Servers configuration page.</td>
</tr>
<tr>
<td>Service Provider</td>
<td>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.</td>
</tr>
<tr>
<td>Protocol</td>
<td>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.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Specify the hostname or IP address of your POP / IMAP mail server. Eg. pop.yourcompany.com or imap.yourcompany.com</td>
</tr>
<tr>
<td>POP / IMAP port</td>
<td>(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.</td>
</tr>
<tr>
<td>Timeout</td>
<td>(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.</td>
</tr>
<tr>
<td>Username</td>
<td>The username used to authenticate your POP / IMAP account.</td>
</tr>
<tr>
<td>Password</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.
**Add POP / IMAP Mail Server**

Use this page to add a new POP / IMAP server for JIRA to retrieve mail from.

- **Name**: The name of this server within JIRA.
- **Description**: Optional description.
- **Service Provider**: Custom.
- **Protocol**: POP.
- **Host Name**: The host name of your POP / IMAP server.
- **POP / IMAP Port**: Optional. The port to use to retrieve mail from your POP / IMAP account. Leave blank for default. (default: POP - 110, IMAP - 993, IMAP-SSL - 993)
- **Timeout**: 10000 milliseconds. 0 or negative values indicate infinite timeout. Leave blank for default (10000 mSec).
- **Username**: The username used to authenticate your POP / IMAP account.
- **Password**: The password for your POP / IMAP account.

[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 article applies to JIRA versions 4.3 and below. For all other versions, please refer to Configuring JIRA’s SMTP Mail Server to Send Notifications.

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.

**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-1.4.5.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">
</Resource>

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
   ----BEGIN CERTIFICATE-----
   MIIDzCAaSgAwIBAgIkm3sQqoX0TmEZyvYhvZBh8R1MnIRZa27vUlLI ...
   END CERTIFICATE-----
   ```

d. Exit the openssl prompt and return to your Java installation's bin directory. Import the cert into
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 another issue tracking application to JIRA, you may wish to take your data with you. Depending on what issue tracker you are migrating from, we recommend using the relevant instructions (linked below) to import data from your other issue tracker into JIRA.

Our website highlights some top reasons why people **migrate from other issue trackers to JIRA**.

### On this page:

- Built-in importers
- CSV importer
- Third-party import tools
- Requests for non-supported importers
- Other non-supported methods
- Other references

**Built-in importers**

The **JIRA Importers plugin**, which is bundled with JIRA, allows you to import data from the importers listed
below.

**Note:** Not all of these importers are available for available JIRA OnDemand.

- Bugzilla
- FogBugz for Your Server
- FogBugz On Demand (SaaS)
- Mantis
- Pivotal Tracker (SaaS)
- Trac
- Redmine
- JSON (beta)

**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.

**Third-party import tools**

Third-party tools created by [Atlassian Experts](https://www.atlassian.com/experts) are also available for the following:

- **HEAT**
  - Go2Group’s Migration scripts
- **HP Quality Center**
  - Go2Group’s JaM
  - Orasi Software’s JIRA Bridge for HP Quality Center
- **IBM ClearQuest**
  - See: JIRA Cookbook - Migration off IBM Rational ClearQuest
  - Appfire’s Enterprise Migration Utility for JIRA
  - Clearvision’s Affinity
  - Go2Group’s ClearCase Add on for JIRA or Go2Group’s ConnectALL
- **IBM DOORS**
  - Go2Group’s ConnectALL
- **Microsoft Team Foundation Server**
  - Appfire’s Enterprise Migration Utility for JIRA
  - Go2Group’s ConnectALL
- **Rally**
  - Appfire’s Enterprise Migration Utility for JIRA
  - Go2Group’s Migration scripts
- **Redmine (now supported)**
  - Go2Group’s Migration scripts
- **Remedy**
  - Go2Group’s Migration scripts
- **SeaPine**
  - Go2Group’s ConnectALL
- **SILK Test**
  - Go2Group’s Silk Add on for JIRA
- **Serena’s TeamTrack PVCS and Business Mashups**
  - Go2Group’s integration mashup tool
- **SalesForce issue tracking**
  - Go2Group’s CRM Plugin
- **StarTeam**
  - Go2Group’s Migration scripts and Go2Group’s BBI (Base Branch Importer)
- **SugarCRM issue tracking**
  - Go2Group’s CRM Plugin
- **VersionOne**
  - Appfire’s Enterprise Migration Utility for JIRA
  - Go2Group’s Migration scripts
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:

- Rally
- Gemini
- GitHub
- Bitbucket
- Code Spaces

Other non-supported methods

There are also a few other non-supported options to get your data into JIRA:

- Write a Jelly script that will import your data. JIRA ships with some Jelly tags that make operations like creating issues in JIRA easy.
- 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.
- 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 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.
- 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.
- 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 (e.g. ServiceRocket, formerly Customware 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

The JIRA Importers plugin, which is bundled with JIRA, allows you to import data from Bugzilla by connecting to a live Bugzilla database.

Our main website highlights some top reasons why people migrate from Bugzilla to JIRA.

Version 4.1 or later of the JIRA Importers plugin is compatible with Bugzilla 2.20 to 4.0.2. Users of older Bugzilla versions will need to first upgrade the Bugzilla database tables to a supported version, using Bugzilla’s checkse tup.pl script. The JIRA Importers plugin requires that your Bugzilla database is MySQL, PostgreSQL or Microsoft SQL Server.

JIRA is able to import data from Bugzilla 2.20 only if it’s using a supported database – in this case, MySQL 5.1 or higher.

JIRA does not bundle the MySQL driver anymore. If the Bugzilla data is located in a MySQL database, follow the instructions at Connecting JIRA to MySQL to install the MySQL database driver before attempting to import from Bugzilla.

The Bugzilla import process consists of simply running the Bugzilla Import Wizard (below).

- You can choose to map individual fields and field values during the import process, some of which are mandatory.
• At the end of the Bugzilla Import Wizard, you will be given the option of creating a Bugzilla configuration file, which contains the settings you configured whilst running through the Bugzilla Import Wizard. This is useful if you need to test your Bugzilla import on a test JIRA server first before performing the import on a production system.

⚠️ 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.

On this page:
• Running the Bugzilla Import Wizard
• Tips for importing Bugzilla data into JIRA fields

Running the Bugzilla Import Wizard

If your JIRA installation has existing data, then before you begin, back up your existing JIRA data.

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 to JIRA as a user with the JIRA Administrators global permission.
4. Choose 🔄 > System. Select Import & Export > External System Import to open the Import external projects page.

✅ Keyboard shortcut: `g + g +` start typing external system import

5. Select the Import button associated with the Bugzilla option to open the Bugzilla Import Wizard: Setup page.
6. On the **Bugzilla Import Wizard: Setup** page, complete the following fields/options:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bugzilla URL</strong></td>
<td>Specify the URL of your Bugzilla site. This is the URL you would normally use to access Bugzilla through a web browser.</td>
</tr>
<tr>
<td><strong>Specify credentials</strong></td>
<td>Select this check box if you want to import Bugzilla issues into JIRA, which require user credentials on your Bugzilla site to access them. Selecting this check box reveals/hides the Bugzilla <strong>Login</strong> and Bugzilla <strong>Password</strong> fields, into which you should specify these user credentials. If your Bugzilla site requires credentials and you do not specify them here, Bugzilla &quot;Big File&quot; attachments will not be imported.</td>
</tr>
<tr>
<td><strong>Database Type</strong></td>
<td>Select the type of database that your Bugzilla installation uses:   • PostgreSQL   • Microsoft SQL Server   • MySQL</td>
</tr>
<tr>
<td><strong>Hostname</strong></td>
<td>Specify the hostname or IP address of the server running your Bugzilla site's database server.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Specify the TCP/IP port that the Bugzilla site's database server is listening on. This field is automatically populated with the default port value based on the <strong>Database Type</strong> you choose above.</td>
</tr>
<tr>
<td><strong>Database</strong></td>
<td>Specify the name of your Bugzilla database (into which Bugzilla saves its data). This database name can usually be found in the 'localconfig' file in Bugzilla's root directory, for example, /etc/bugzilla/</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>Specify the database user that Bugzilla uses to connect to its database. This database user can usually be found in the 'localconfig' file in Bugzilla's root directory, for example, /etc/bugzilla/</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>Specify the password of the database user (above) that Bugzilla uses to connect to its database. This password can usually be found in the 'localconfig' file in Bugzilla's root directory, for example, /etc/bugzilla/</td>
</tr>
</tbody>
</table>
Use an existing configuration file

Leave this 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 fields in Bugzilla and those in JIRA.

**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 Bugzilla Import Wizard, JIRA will create a configuration file which you can use for subsequent Bugzilla imports (for re-use at this step of the Bugzilla Import Wizard).

JDBC connection parameters

(in expanded Advanced tab)

The Bugzilla Import Wizard will construct a JDBC-based database URL from the Bugzilla database server details you specify above. JIRA uses this URL to connect to and import issues from Bugzilla. If you need to specify any additional connection parameters to your Bugzilla database, specify them here.

If you chose MySQL (above), the Bugzilla Import Wizard will add several additional connection parameters by default.

Click the Next button to proceed to the Setup project mappings step of the Bugzilla Import Wizard.

7. Click the Next button to proceed to the Setup project mappings step of the Bugzilla Import Wizard.

8. On the Setup project mappings page, select which Bugzilla projects you wish to import into JIRA.

   If all Bugzilla projects are selected by default, so clear the check boxes under Import of the Bugzilla projects you do not wish to import into JIRA.

   For Bugzilla projects you wish to import into JIRA, click in 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 new project Name
     - A new project Key

   **Note:** This will be used as the prefix for all issue IDs in your JIRA project.

   - The Project Lead.

9. Click the Next button to proceed to the Setup custom fields step of the Bugzilla Import Wizard.

   This step will almost always appear because at least one Bugzilla field is not likely match an existing JIRA field.

10. On the Setup custom fields page, for each External field in Bugzilla which the Bugzilla Import Wizard cannot match to an existing JIRA field, you can choose to either:

    - have the Bugzilla Import Wizard automatically create new custom fields in JIRA based on the names of Bugzilla’s fields. This is the default option - whereby the names of the JIRA custom fields to be automatically created appear in the JIRA field dropdown lists.
    - create your own custom fields in JIRA to map data from Bugzilla’s fields. To do this, choose Other from the JIRA field dropdown list and specify the name of your custom field in the new field appearing immediately below Other.

   **Note:** For more information about matching Bugzilla fields to JIRA fields, see Tips for importing Bugzilla data into JIRA fields below.

11. Click the Next button to proceed to the Setup field mappings step of the Bugzilla Import Wizard.

12. On the Setup field mappings page, if there External fields in Bugzilla whose values you wish to modify before they are imported into JIRA, select the Map field value check boxes next to the appropriate fields.

   **Note:** Please note that it is mandatory to map Bugzilla’s bug_status (i.e. Status) field to specific JIRA Status field values as the JIRA Status field is an integral part of JIRA workflows.

   - Other External fields in Bugzilla which are likely to appear on the Setup field mappings page are:
### External field in Bugzilla vs. Not choosing the 'Map field value' check box

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>login_name</td>
<td>The Bugzilla Import Wizard will automatically map Bugzilla usernames to JIRA usernames (lowercase).</td>
</tr>
<tr>
<td>priority</td>
<td>The Bugzilla Import Wizard will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority (e.g. &quot;Normal&quot; in Bugzilla to newly-created &quot;Normal&quot; in JIRA).</td>
</tr>
<tr>
<td>resolution</td>
<td>The importer will create corresponding Resolutions in JIRA instead of using the existing ones.</td>
</tr>
</tbody>
</table>

- Select the appropriate JIRA **Workflow Scheme** in that will be used by the Bugzilla issues you will import into your JIRA project.
  - If you are importing your Bugzilla issues into an existing JIRA project, ensure that you choose the JIRA workflow scheme used by that existing JIRA project.

13. Click the **Next** button to proceed to the **Setup value mappings** step of the Bugzilla Import Wizard.

14. On the **Setup value mappings** page, specify JIRA field values for each Bugzilla field value (as detected by the Bugzilla Import Wizard).

  - Any fields whose **Map field value** check boxes were selected in the previous step of the Bugzilla Import Wizard will be presented on this page, including the mandatory **bug_status** Bugzilla field.

15. Click the **Next** button to proceed to the **Setup links** step of the Bugzilla Import Wizard.

16. On the **Setup links** page, specify the JIRA link type for each Bugzilla link type (as detected by the Bugzilla Import Wizard). To learn more about JIRA link types, please see **Configuring Issue Linking**.

17. Click the **Begin Import** button when you are ready to begin importing your Bugzilla 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 Bugzilla Import Wizard process.
- If you need to import data from another Bugzilla product/project or site with the same (or similar) settings to what you used through this procedure, click the **save the configuration** link to download a Bugzilla configuration file, which you can use at the **first step** of the Bugzilla Import Wizard.

Congratulations, you have successfully imported your Bugzilla projects into JIRA! If you have any questions or encounter any problems, please contact [Atlassian support](https://support.atlassian.com/).

### Tips for importing Bugzilla data into JIRA fields

During the import process, the following data is copied from the Bugzilla database into JIRA:

<table>
<thead>
<tr>
<th>In Bugzilla</th>
<th>In JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Project</td>
<td>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](<a href="https://confluence.atlassian.com/jira-issues">https://confluence.atlassian.com/jira-issues</a> gestión/).)</td>
</tr>
<tr>
<td>External Project</td>
<td>Project Category</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>Version</td>
<td>Affects Version</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Fix Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Versions are imported from Bugzilla (if you choose) and are set to the Un-Released and Un-Archived state.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bug</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>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'.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>External issue ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Each imported issue will be given a new JIRA ID, and the old Bugzilla 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 Bugzilla ID. If you don't need this custom field, delete it or 'hide' it (as described in Specifying Field Behavior).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Summary</th>
<th>Summary</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Comments</th>
<th>Comments</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Attachments</th>
<th>Attachments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attachments are extracted from the Bugzilla database and saved to disk. To specify the location on disk, see Configuring File Attachments.</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Source</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority (or a custom field)</td>
</tr>
<tr>
<td>Severity</td>
<td>Priority (or a custom field)</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>Resolution</td>
</tr>
</tbody>
</table>
| Duplicates | Link | You can configure mapping of specific Bugzilla link types to JIRA link types.  
|------------|------|---------------------------------------------------------------------  
| Depends on |      | • In JIRA, you can configure different types of links (please see Configuring Issue Linking).  
| Blocks     |      |                                                                      
|            |      | **Work History**                                                     |  
| 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.  

*Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
• 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.

Importing Data from FogBugz for Your Server

The JIRA Importers plugin, which is bundled with JIRA, allows you to import data from FogBugz for Your Server by connecting to a live FogBugz for Your Server database.

Our main website highlights some top reasons why people migrate from FogBugz to JIRA. Version 4.2 or later of the JIRA Importers plugin is compatible with Fogbugz for Your Server versions 7.3.6 to 8.7.60. The JIRA Importers plugin requires that your FogBugz for Your Server database is MySQL, Microsoft SQL Server or Microsoft SQL Server Express.

The FogBugz for Your Server import process consists of simply running the FogBugz Import Wizard (below):

- You can choose to map individual fields and field values during the import process, some of which are mandatory.
- At the end of the FogBugz Import Wizard, you will be given the option of creating a FogBugz configuration file, which contains the settings you configured whilst running through the FogBugz Import Wizard. This is useful if you need to test your FogBugz import on a test JIRA server first before performing the import on a production system.

These instructions refer to a FogBugz for Your Server, which is an installable implementation of FogBugz that operates behind your firewall. To import from a FogBugz On Demand (SaaS) issue tracker site, please follow the instructions for here.

On this page:
- Running the FogBugz for Your Server Import Wizard
- Tips for importing FogBugz for Your Server data into JIRA fields

Running the FogBugz for Your Server Import Wizard

1. Before you begin, please backup your JIRA data.
2. Log in to JIRA as a user with the JIRA Administrators global permission.
3. Choose 🔄 > System. Select Import & Export > External System Import to open the Import external projects page.

Keyboard shortcut: g + g + start typing external system import
4. Select the Import button associated with the FogBugz for Your Server option to open the FogBugz Import Wizard: Setup page.
5. On the **FogBugz Setup** page, complete the following fields/options:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Database Type</strong></td>
<td>Select the type of database that your FogBugz for Your Server installation uses:</td>
</tr>
<tr>
<td></td>
<td>- PostgreSQL</td>
</tr>
<tr>
<td></td>
<td>- Microsoft SQL Server</td>
</tr>
<tr>
<td></td>
<td>- MySQL</td>
</tr>
<tr>
<td><strong>Hostname</strong></td>
<td>Specify the hostname or IP address of the server running your FogBugz site's database server.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Specify the TCP/IP port that the FogBugz site's database server is listening on.</td>
</tr>
</tbody>
</table>

- This field is automatically populated with the default port value based on the **Database Type** you choose above.
| **Database** | Specify the name of your FogBugz database (into which FogBugz for Your Server saves its data).  
| If you need to specify an instance ID for your database, do so using the syntax `fogbugz;instance=sqlexpress` (where `fogbugz` is the name of your FogBugz database and `sqlexpress` is your FogBugz database's instance ID. The database name can usually be found in the Windows registry. See [http://bugs.movabletype.org/help/topics/setup/WindowsWhatSetupDoes.html](http://bugs.movabletype.org/help/topics/setup/WindowsWhatSetupDoes.html) and then search for 'Initialize Registry Settings' (for details on how to access the relevant registry keys and values). |
| **Username** | Specify the database user that FogBugz uses to connect to its database. |
| **Password** | Specify the password of the database user (above) that FogBugz uses to connect to its database. |
| **Use an existing configuration file** | Leave this 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 fields in FogBugz for Your Server and those in JIRA.  
| **JDBC connection parameters** (in expanded Advanced tab) | The FogBugz Import Wizard will construct a JDBC-based database URL from the FogBugz database server details you specify above. JIRA uses this URL to connect to and import issues from FogBugz for Your Server. If you need to specify any additional connection parameters to your FogBugz database, specify them here.  
| If you chose MySQL (above), the FogBugz Import Wizard will add several additional connection parameters by default. |

6. Click the **Next** button to proceed to the **Set up project mappings** step of the FogBugz Import Wizard.  
7. On the **Set up project mappings** page, select which FogBugz projects you wish to import into JIRA.  
8. All projects are selected by default, so clear the check boxes under **Import** of the FogBugz projects you do not wish to import into JIRA.  
For FogBugz projects you wish to import into JIRA, click in **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**  
  c. The **Project Lead**.  
If you chose MySQL (above), the FogBugz Import Wizard will add several additional connection parameters by default.  
9. Click the **Next** button to proceed to the **Set up custom fields** step of the FogBugz Import Wizard.  
10. On the **Set up custom fields** page, for each **External field** in FogBugz which the FogBugz Import
Wizard cannot match to an existing JIRA field, you can choose to either:
  • have the FogBugz Import Wizard automatically create new custom fields in JIRA based on the names of FogBugz's fields. This is the default option - whereby the names of the JIRA custom fields to be automatically created appear in the JIRA field dropdown lists.
  • create your own custom fields in JIRA to map data from FogBugz's fields. To do this, choose Other from the JIRA field dropdown list and specify the name of your custom field in the new field appearing immediately below Other.

11. Click the Next button to proceed to the Set up field mappings step of the FogBugz Import Wizard.
12. On the Set up field mappings page, if there External fields in FogBugz whose values you wish to modify before they are imported into JIRA, select the Map field value check boxes next to the appropriate fields.

Please note that it is mandatory to map FogBugz's sStatus (i.e. Status) field to specific JIRA Status field values as the JIRA Status field is an integral part of JIRA workflows.

• Other External fields in FogBugz which are likely to appear on the Set up field mappings page are:

<table>
<thead>
<tr>
<th>External field in FogBugz</th>
<th>Not choosing the 'Map field value' check box</th>
</tr>
</thead>
<tbody>
<tr>
<td>sCategory</td>
<td>The FogBugz Import Wizard will automatically create missing issue types in JIRA and will ensure that the issues are migrated with the correct issue type.</td>
</tr>
<tr>
<td>sCustomerEmail</td>
<td>The FogBugz Import Wizard will not map values for this field.</td>
</tr>
<tr>
<td>sComputer</td>
<td>The FogBugz Import Wizard will not map values for this field.</td>
</tr>
<tr>
<td>sFullName</td>
<td>The FogBugz Import Wizard will automatically map FogBugz usernames to JIRA usernames (lowercase).</td>
</tr>
<tr>
<td>sPriority</td>
<td>The FogBugz Import Wizard will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority (e.g. &quot;Normal&quot; in FogBugz to newly-created &quot;Normal&quot; in JIRA).</td>
</tr>
<tr>
<td>sStatus (Resolution)</td>
<td>The importer will create corresponding Resolutions in JIRA instead of using the existing ones.</td>
</tr>
</tbody>
</table>

• Select the appropriate JIRA Workflow Scheme in that will be used by the FogBugz issues you will import into your JIRA project.

If you are importing your FogBugz issues into an existing JIRA project, ensure that you choose the JIRA workflow scheme used by that existing JIRA project.

13. Click the Next button to proceed to the Set up value mappings step of the FogBugz Import Wizard.
14. On the Set up value mappings page, specify JIRA field values for each FogBugz field value (as detected by the FogBugz Import Wizard).

Any fields whose Map field value check boxes were selected in the previous step of the FogBugz Import Wizard will be presented on this page, including the mandatory sStatus FogBugz field.

15. Click the Next button to proceed to the Set up links step of the FogBugz Import Wizard.
16. On the Set up links page, specify the JIRA link type for each FogBugz link type (as detected by the FogBugz Import Wizard). To learn more about JIRA link types, please see Configuring Issue Linking.
17. Click the Begin Import button when you are ready to begin importing your FogBugz 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 FogBugz Import Wizard process.
  • If you need to import data from another FogBugz product/project or site with the same (or similar)
settings to what you used through this procedure, click the save the configuration link to download a FogBugz configuration file, which you can use at the first step of the FogBugz Import Wizard.

Congratulations, you have successfully imported your FogBugz projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Tips for importing FogBugz for Your Server data into JIRA fields

During the import process, the following data is copied from the FogBugz Server database into JIRA:

<table>
<thead>
<tr>
<th>In FogBugz</th>
<th>In JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Project</td>
<td>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.)</td>
</tr>
<tr>
<td>Area</td>
<td>Component</td>
<td>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.</td>
</tr>
<tr>
<td>Milestone</td>
<td>Fix Version</td>
<td>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.</td>
</tr>
<tr>
<td>Case</td>
<td>Issue</td>
<td>Every FogBugz case becomes a JIRA issue.</td>
</tr>
<tr>
<td>Case ID</td>
<td>Bug Import ID</td>
<td>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 Behavior).</td>
</tr>
<tr>
<td>Summary</td>
<td>Summary</td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>Comments</td>
<td>FogBugz allows for links to other issues to be automatically generated by using the format &quot;bug issueld&quot; or &quot;case issue id&quot;. After import, any string matching this pattern will be rewritten to their new JIRA key. For example, a comment &quot;Please see case 100&quot; may be rewritten to &quot;Please see IMP-100&quot;.</td>
</tr>
<tr>
<td>Category</td>
<td>Issue Type</td>
<td>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.</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority</td>
<td>You can configure mapping of specific Case Categories to specific Issue Types.</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
<td>You can configure mapping of specific FogBugz values to specific JIRA values, provided you create your workflows in JIRA before running the importer.</td>
</tr>
<tr>
<td>Resolution</td>
<td>Resolution</td>
<td>You can configure mapping of specific FogBugz values to specific JIRA values.</td>
</tr>
<tr>
<td>Duplicates</td>
<td>Links</td>
<td>You can configure mapping of specific FogBugz link types to JIRA link types.</td>
</tr>
<tr>
<td>BugRelations</td>
<td></td>
<td>• In JIRA, you can configure different types of links (please see Configuring Issue Linking).</td>
</tr>
<tr>
<td>Computer</td>
<td>Computer</td>
<td>The FogBugz Computer field is imported into a JIRA Custom Field called 'Computer'.</td>
</tr>
<tr>
<td>Customer Email</td>
<td>Customer Email</td>
<td>The FogBugz Customer Email field is imported into a JIRA Custom Field called 'Customer Email'.</td>
</tr>
<tr>
<td>User</td>
<td>User</td>
<td>You can choose to have the importer automatically create JIRA users for any FogBugz users who do not already exist in JIRA.</td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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 &quot;fogbugz-import-unused-users&quot; and will be deactivated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Users with no real name stored in FogBugz will get the portion of their email address (login name) before the &quot;@&quot; character as their Full Name in JIRA.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
</tbody>
</table>

| 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. |

**Importing Data from FogBugz On Demand**
The **JIRA Importers plugin**, which is bundled with JIRA, allows you to import data from **FogBugz On Demand**, a 'Software as a Service' (SaaS) issue tracker product.

ℹ️ Our main website highlights some top reasons why people migrate from FogBugz to JIRA. Version 3.1 or later of the JIRA Importers Plugin is required.

These instructions refer to **FogBugz On Demand**, which is a SaaS implementation of FogBugz. To import from the installable **FogBugz for Your Server** implementation of FogBugz, please follow the instructions for here.

---

### On this page:
- Running the FogBugz On Demand Import Wizard
- Tips for importing FogBugz On Demand data into JIRA fields

---

#### Running the FogBugz On Demand Import Wizard

**Before you begin:** If your JIRA installation has existing data — **Back up** your existing JIRA data.

✅ **Tip:** FogBugz On Demand supports hierarchical issues. During the FogBugz On Demand Import Wizard, you are given the option to recreate this issue hierarchy through **JIRA issue links**. Hence, before commencing the FogBugz On Demand Import Wizard, you may wish to **configure an custom issue link** to replicate this hierarchy — for example:

- **Name** — 'Hierarchy'
- **Outward Link Description** — 'parent of'
- **Inward Link Description** — 'child of'

#### To import issues FogBugz On Demand:

1. Log in to JIRA as a user with the **JIRA Administrators** global permission.

2. Choose 🕹️ > **System**. Select **Import & Export > External System Import** to open the Import external projects page.
3. Select the Import button associated with the FogBugz On Demand option to open the Connect with FogBugz page.

4. On the Connect with FogBugz page, complete the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FogBugz On Demand URL</td>
<td>Specify the URL of your FogBugz On Demand site. This is the URL you would normally use to access FogBugz On Demand through a web browser. This is usually of the format <a href="http://myfogbugzondemand.fogbugz.com">http://myfogbugzondemand.fogbugz.com</a></td>
</tr>
<tr>
<td>FogBugz Username</td>
<td>Specify the user account that JIRA will use to access issues on your FogBugz On Demand site.</td>
</tr>
<tr>
<td>FogBugz Password</td>
<td>Specify the password of the user (above).</td>
</tr>
</tbody>
</table>

5. Click the Next button to proceed to the Setup project mappings step of the FogBugz On Demand Import Wizard.

6. On the Setup project mappings page, select which FogBugz On Demand projects you wish to import into JIRA.

7. All FogBugz On Demand projects are selected by default, so clear the check boxes under Import of the FogBugz On Demand projects you do not wish to import into JIRA. For FogBugz On Demand projects you wish to import into JIRA, click in 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 new project Name.
b. A new project **Key**
   - This will be used as the prefix for all issue IDs in your JIRA project.

8. Click the **Next** button to proceed to the **Setup field mappings** step of the FogBugz On Demand Import Wizard.

9. On the **Setup field mappings** page, if there are **External fields** in FogBugz On Demand whose values you wish to modify before they are imported into JIRA, select the **Map field value** check boxes next to the appropriate fields.
   - Please note that it is mandatory to map FogBugz On Demand's **sStatus** (i.e., **Status**) field to specific JIRA **Status** field values as the JIRA **Status** field is an integral part of JIRA workflows.
     * The FogBugz On Demand field **sStatus (Resolution)** (i.e., **Resolution**), which will be mapped to the JIRA **Resolution** field, may also appear on this page.
     * Select the appropriate JIRA **Workflow Scheme** in that will be used by the FogBugz On Demand issues you will import into your JIRA project.

   - **Warning:** If you are importing your FogBugz On Demand issues into an existing JIRA project, ensure that you choose the JIRA workflow scheme used by that existing JIRA project. Otherwise, your import may not complete successfully.

10. Click the **Next** button to proceed to the **Setup value mappings** step of the FogBugz On Demand Import Wizard.

11. On the **Setup value mappings** page, specify JIRA field values for each FogBugz On Demand field value (as detected by the FogBugz On Demand Import Wizard).
   - Any fields whose **Map field value** check boxes were selected in the previous step of the FogBugz On Demand Import Wizard will be presented on this page, including the mandatory **sStatus** FogBugz On Demand field.

12. Click the **Next** button to proceed to the **Setup links** step of the FogBugz On Demand Import Wizard.

13. On the **Setup links** page, specify how you want to map FogBugz On Demand's **Parent / Subcase** relationships through a JIRA **issue links**. To learn more about JIRA link types, please see Configuring Issue Linking.
   - You may wish to choose the 'Hierarchy' custom issue link you created before running the FogBugz On Demand Import Wizard.

14. Click the **Begin Import** button when you are ready to begin importing your FogBugz On Demand 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 FogBugz On Demand Import Wizard process.

Congratulations, you have successfully imported your FogBugz On Demand projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

**Tips for importing FogBugz On Demand data into JIRA fields**

The import process converts FogBugz On Demand data as follows:

<table>
<thead>
<tr>
<th>FogBugz On Demand</th>
<th>In JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Project</td>
<td>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.)</td>
</tr>
<tr>
<td>Area</td>
<td>Component</td>
<td>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.</td>
</tr>
<tr>
<td>Milestone</td>
<td>Fix Version</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Version are imported from FogBugz (if you choose). After importing, you can manually set appropriate versions to the Released state in JIRA if you wish.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every FogBugz case becomes a JIRA issue.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case ID</th>
<th>External issue ID and External issue URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>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 Behavior).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Summary</th>
<th>Summary</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Comments</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>FogBugz allows for links to other issues to be automatically generated by using the format &quot;bug issueId&quot; or &quot;case issue id&quot;. After import, any string matching this pattern will be rewritten to their new JIRA key. For example, a comment &quot;Please see case 100&quot; may be rewritten to &quot;Please see IMP-100&quot;.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attachments</th>
<th>Attachments</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Issue Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can configure mapping of specific Case Categories to specific Issue Types.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Priority</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can configure mapping of specific FogBugz values to specific JIRA values.</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>Resolution</td>
</tr>
<tr>
<td>--------</td>
<td>------------</td>
</tr>
<tr>
<td>You can configure mapping of specific FogBugz values to specific JIRA values, provided you create your workflows in JIRA before running the importer.</td>
<td></td>
</tr>
<tr>
<td><em>The JIRA Status field is integral to JIRA workflow. To learn more, please see What is Workflow.</em></td>
<td></td>
</tr>
<tr>
<td><em>To create a JIRA workflow, please see Configuring Workflow.</em></td>
<td></td>
</tr>
<tr>
<td><em>To create a JIRA workflow scheme (which you can then associate with appropriate projects and Issue Types), please see Activating workflow.</em></td>
<td></td>
</tr>
<tr>
<td>You can configure mapping of specific FogBugz values to specific JIRA values.</td>
<td></td>
</tr>
<tr>
<td>They are not imported due to limitations of FogBugz Remote API</td>
<td></td>
</tr>
<tr>
<td>The FogBugz Computer field is imported into a JIRA Custom Field called 'Computer'.</td>
<td></td>
</tr>
<tr>
<td>The FogBugz Customer Email field is imported into a JIRA Custom Field called 'Customer Email'.</td>
<td></td>
</tr>
</tbody>
</table>
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.

Importing Data from Mantis
The **JIRA Importers plugin**, which is bundled with JIRA, allows you to import data from **Mantis** by connecting to a live Mantis database.

**Our main website highlights some top reasons why people migrate from Mantis to JIRA.**

Version 4.2 or later of the JIRA Importers plugin is compatible with Mantis versions 1.1.8 to 1.2.8. The JIRA Importers plugin requires that your Mantis database is MySQL, PostgreSQL or Microsoft SQL Server. We have also received reports that the JIRA Importers plugin works with Oracle and DB2 databases. However, we have not tested this plugin against these databases.)

The Mantis import process consists of simply running the Mantis Import Wizard (below).

- You can choose to map individual fields and field values during the import process, some of which are mandatory.
- At the end of the Mantis Import Wizard, you will be given the option of creating a Mantis configuration file, which contains the settings you configured whilst running through the Mantis Import Wizard. This is useful if you need to test your Mantis import on a test JIRA server first before performing the import on a production system.

### On this page:

- Running the Mantis Import Wizard
- Tips for importing Mantis data into JIRA fields

### Running the Mantis Import Wizard

1. Before you begin, please **backup** your JIRA data.
2. Log in to JIRA as a user with the **JIRA Administrators** global permission.
3. Choose 🔗 System. Select Import & Export > External System Import to open the Import external projects page.

   **Keyboard shortcut:** `g + g + external system import`

4. Select the **Import** button associated with the **Mantis** option to open the Mantis Import Wizard: Setup page.
5. On the **Mantis Import Wizard: Setup** page, complete the following fields/options:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mantis URL</strong></td>
<td>Specify the URL of your Mantis site. This is the URL you would normally use to access Mantis through a web browser.</td>
</tr>
<tr>
<td><strong>Specify credentials</strong></td>
<td>Select this check box if you want to import Mantis issues into JIRA, which require user credentials on your Mantis site to access them. Selecting this check box reveals/hides the <strong>Mantis Login</strong> and <strong>Mantis Password</strong> fields, into which you should specify these user credentials.</td>
</tr>
<tr>
<td><strong>Database Type</strong></td>
<td>Select the type of database that your Mantis installation uses:</td>
</tr>
<tr>
<td></td>
<td>- PostgreSQL</td>
</tr>
<tr>
<td></td>
<td>- Microsoft SQL Server</td>
</tr>
<tr>
<td></td>
<td>- MySQL</td>
</tr>
<tr>
<td><strong>Hostname</strong></td>
<td>Specify the hostname or IP address of the server running your Mantis site's database server.</td>
</tr>
<tr>
<td>Port</td>
<td>Specify the TCP/IP port that the Mantis site's database server is listening on. This field is automatically populated with the default port value based on the Database Type you choose above.</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Database</td>
<td>Specify the name of your Mantis database (into which Mantis saves its data). The database name, username and user password can usually be found in the Mantis file config_inc.php. (Typically, the default username is &quot;root&quot; and the default password is empty). See also <a href="http://www.mantisbt.org/manual/manual.configuration.database.php">http://www.mantisbt.org/manual/manual.configuration.database.php</a></td>
</tr>
<tr>
<td>Username</td>
<td>Specify the database user that Mantis uses to connect to its database.</td>
</tr>
<tr>
<td>Password</td>
<td>Specify the password of the database user (above) that Mantis uses to connect to its database.</td>
</tr>
<tr>
<td>Use an existing configuration file</td>
<td>Leave this 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 fields in Mantis and those in JIRA. 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 Mantis Import Wizard, JIRA will create a configuration file which you can use for subsequent Mantis imports (for re-use at this step of the Mantis Import Wizard).</td>
</tr>
<tr>
<td>JDBC connection parameters (in expanded Advanced tab)</td>
<td>The Mantis Import Wizard will construct a JDBC-based database URL from the Mantis database server details you specify above. JIRA uses this URL to connect to and import issues from Mantis. If you need to specify any additional connection parameters to your Mantis database, specify them here. If you chose MySQL (above), the Mantis Import Wizard will add several additional connection parameters by default.</td>
</tr>
</tbody>
</table>

6. Click the Next button to proceed to the Set up project mappings step of the Mantis Import Wizard.  
7. On the Set up project mappings page, select which Mantis projects you wish to import into JIRA. All Mantis projects are selected by default, so clear the check boxes under Import of the Mantis projects you do not wish to import into JIRA.  
   For Mantis projects you wish to import into JIRA, click in 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  
     This will be used as the prefix for all issue IDs in your JIRA project.  
     c. The Project Lead.
8. Click the Next button to proceed to the Set up custom fields step of the Mantis Import Wizard.
   This step will almost always appear because at least one Mantis field is not likely match an existing JIRA field.

9. On the Set up custom fields page, for each External field in Mantis which the Mantis Import Wizard cannot match to an existing JIRA field, you can choose to either:
   - have the Mantis Import Wizard automatically create new custom fields in JIRA based on the names of Mantis's fields. This is the default option - whereby the names of the JIRA custom fields to be automatically created appear in the JIRA field dropdown lists.
   - create your own custom fields in JIRA to map data from Mantis's fields. To do this, choose Other from the JIRA field dropdown list and specify the name of your custom field in the new field appearing immediately below Other.

10. Click the Next button to proceed to the Set up field mappings step of the Mantis Import Wizard.
11. On the Set up field mappings page, if there External fields in Mantis whose values you wish to modify before they are imported into JIRA, select the Map field value check boxes next to the appropriate fields.
   Please note that it is mandatory to map Mantis's status (i.e. Status) field to specific JIRA Status field values as the JIRA Status field is an integral part of JIRA workflows.
   - Other External fields in Mantis which are likely to appear on the Set up field mappings page are:

<table>
<thead>
<tr>
<th>External field in Mantis</th>
<th>Not choosing the 'Map field value' check box</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>The Mantis Import Wizard will automatically map Mantis usernames to JIRA usernames (lowercase).</td>
</tr>
<tr>
<td>priority</td>
<td>The Mantis Import Wizard will automatically create missing values in JIRA and will ensure that the issues are migrated with the correct priority (e.g. &quot;Normal&quot; in Mantis to newly-created &quot;Normal&quot; in JIRA).</td>
</tr>
<tr>
<td>severity</td>
<td>The Mantis Import Wizard will not map values for this field.</td>
</tr>
<tr>
<td>resolution</td>
<td>The importer will create corresponding Resolutions in JIRA instead of using the existing ones.</td>
</tr>
</tbody>
</table>

   - Select the appropriate JIRA Workflow Scheme in that will be used by the Mantis issues you will import into your JIRA project.
   If you are importing your Mantis issues into an existing JIRA project, ensure that you choose the JIRA workflow scheme used by that existing JIRA project.

12. Click the Next button to proceed to the Set up value mappings step of the Mantis Import Wizard.
13. On the Set up value mappings page, specify JIRA field values for each Mantis field value (as detected by the Mantis Import Wizard).
   Any fields whose Map field value check boxes were selected in the previous step of the Mantis Import Wizard will be presented on this page, including the mandatory status Mantis field.

14. Click the Next button to proceed to the Set up links step of the Mantis Import Wizard.
15. On the Set up links page, specify the JIRA link type for each Mantis link type (as detected by the Mantis Import Wizard). To learn more about JIRA link types, please see Configuring Issue Linking.
16. Click the Begin Import button when you are ready to begin importing your Mantis 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 Mantis Import Wizard process.
   - If you need to import data from another Mantis product/project or site with the same (or similar) settings to what you used through this procedure, click the save the configuration link to download a Mantis configuration file, which you can use at the first step of the Mantis Import Wizard.

Congratulations, you have successfully imported your Mantis projects into JIRA! If you have any questions or
encounter any problems, please contact Atlassian support.

**Tips for importing Mantis data into JIRA fields**

During the import process, the following data is copied from the Mantis database into JIRA:

<table>
<thead>
<tr>
<th>In Mantis</th>
<th>In JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Project</td>
<td>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.)</td>
</tr>
<tr>
<td>Sub Project</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Category</td>
<td>Component</td>
<td>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.</td>
</tr>
<tr>
<td>Version</td>
<td>Fix Version</td>
<td>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.</td>
</tr>
<tr>
<td>Bug</td>
<td>Issue</td>
<td>Every Mantis bug becomes a JIRA issue of type 'Bug'.</td>
</tr>
<tr>
<td>ID</td>
<td>Bug Import ID</td>
<td>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 Behavior).</td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td>Within text, Mantis links (e.g. #1234) are converted to JIRA links (e.g. TST-123).</td>
</tr>
<tr>
<td>Comments</td>
<td>Comments</td>
<td>Within text, Mantis links (e.g. #1234) are converted to JIRA links (e.g. TST-123).</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Attachments</td>
<td>Attachments</td>
<td>Attachments are extracted from the Mantis database and saved to disk. To specify the location on disk, see <a href="#">Configuring File Attachments</a>.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority (or a custom field)</td>
<td>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.</td>
</tr>
<tr>
<td>Severity</td>
<td>Priority (or a custom field)</td>
<td>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.</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
<td>You can configure mapping of specific Mantis values to specific JIRA values, provided you create your workflows in JIRA before running the importer.</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The JIRA 'Status' field is integral to JIRA workflow. To learn more, please see <a href="#">What is Workflow</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• To create a JIRA workflow, please see <a href="#">Configuring Workflow</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• To create a JIRA workflow scheme (which you can then associate with appropriate projects and Issue Types), please see <a href="#">Activating workflow</a>.</td>
</tr>
<tr>
<td>Resolution</td>
<td>Resolution</td>
<td>You can configure mapping of specific Mantis values to specific JIRA values.</td>
</tr>
<tr>
<td>Relationships</td>
<td>Links</td>
<td>You can configure mapping of specific Mantis relationship types to JIRA link types.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In JIRA, you can configure different types of links (please see <a href="#">Configuring Issue Linking</a>).</td>
</tr>
<tr>
<td>CC List</td>
<td>Watchers</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>User</td>
<td>You can choose to have the importer automatically create JIRA users for any Mantis users who do not already exist in JIRA.</td>
</tr>
</tbody>
</table>
- 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.
### Importing Data from Pivotal Tracker

The JIRA Importers plugin, which is bundled with JIRA, allows you to import data from **Pivotal Tracker**, a 'Software as a Service' (SaaS) issue tracker product.

ℹ️ Our main website highlights some top reasons why people migrate from Pivotal Tracker to JIRA.

Version 2.5 or later of the JIRA Importers Plugin is required.

#### On this page:
- Preparing Pivotal Tracker for data import into JIRA
- Running the Pivotal Tracker Import Wizard
- Tips for importing Pivotal Tracker data into JIRA fields

#### Preparing Pivotal Tracker for data import into JIRA

In Pivotal Tracker, please ensure you have switched on **Allow API Access** in your Pivotal Project's Settings.

![Click here and then Projects tab and Settings "api"](image)

#### Running the Pivotal Tracker Import Wizard

**Before you begin:** If your JIRA installation has existing data — **Back up** your existing JIRA data.

1. Log in to JIRA as a user with the **JIRA Administrators** global permission.
2. Choose **⚙️ > System**. Select **Import & Export > External System Import** to open the Import external projects page.
   - **Keyboard shortcut:** `g + g + start typing external system import`
3. Select the **Import** button associated with the Pivotal Tracker option to open the **Connect with Pivotal Tracker** form.

---

**Other fields**

| Custom fields | If your Mantis system contains any custom fields, you can choose to map them to specific JIRA fields. If your custom fields don’t yet exist in JIRA, the importer can automatically create them for you. |  |

---

*Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](http://creativecommons.org/licenses/by/2.5/au/).*
Tracker page.

4. On the Connect with Pivotal Tracker page, specify the following:

<table>
<thead>
<tr>
<th><strong>Pivotal Username or Email</strong></th>
<th>Specify the user account that JIRA will use to access issues on your Pivotal Tracker site.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pivotal Password</strong></td>
<td>Specify the password of the user (above).</td>
</tr>
<tr>
<td><strong>Map user names</strong> (in expanded Advanced tab)</td>
<td>Select this check box if you want to modify the name details of Pivotal Tracker users (which would be associated with Pivotal Tracker issues) when these users are created in JIRA.</td>
</tr>
<tr>
<td><strong>Use an existing configuration file</strong> (in expanded Advanced tab)</td>
<td>Leave this 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 fields in Pivotal Tracker and those in JIRA.</td>
</tr>
</tbody>
</table>

**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 Pivotal Tracker Import Wizard, JIRA will create a configuration file which you can use for subsequent Pivotal Tracker imports (for re-use at this step of the Pivotal Tracker Import Wizard).

5. Click the Next button to proceed to the Setup project mappings step of the Pivotal Tracker Import Wizard.

6. On the Setup project mappings page, select which Pivotal Tracker projects you wish to import into JIRA.

   - All Pivotal Tracker projects are selected by default, so clear the check boxes under Import of the Pivotal Tracker projects you do not wish to import into JIRA.
   - For Pivotal Tracker projects you wish to import into JIRA, click in Select a project and then do either of the following:
     - 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.
       c. The Project Lead.
     - 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.

   **Tip:** Only JIRA projects that use the PT Workflow Scheme (which is created with your first Pivotal Tracker import into JIRA) can be chosen from the Select a project list. The PT Workflow Scheme consists of the:
   - PT Workflow — mapped to all standard issue types.
   - PT Subtask Workflow — mapped to JIRA's sub-task issue type.

   **Tip:** If you have not yet performed a Pivotal Tracker import into JIRA but you would like to import your Pivotal Tracker issues into an existing JIRA project, consider doing the following:
   a. Use the Pivotal Tracker Import Wizard to import your issues into a new JIRA project. Upon doing so, JIRA will create the PT Workflow Scheme and PT Issue Type Scheme. The PT Issue Type Scheme consists of additional issue types that do not exist in a default JIRA installation, such as Chore and Release.
   b. (Optional) Delete this project if you do not intend to use it any further.
   c. Apply the PT Workflow Scheme and PT Issue Type Scheme to the existing JIRA project you want to import your Pivotal Tracker issues into. (See Configuring a project for details.)
   d. Re-use the Pivotal Tracker Import Wizard to import your issues into this existing JIRA project.

7. Click the Next button to proceed to the Setup user mappings step of the Pivotal Tracker Import Wizard.

   **Tip:** If you did not select Map user names option above, skip to step 8. (The Next button will not be
8. On the **Setup user mappings** step of the Pivotal Tracker Import Wizard, in the **Target value in JIRA** field:
   - Specify the **username** of a JIRA user to match Pivotal Tracker users to existing JIRA users.
   - Leave blank to add the Pivotal Tracker user's name details 'as is'. The user's Full Name in JIRA is derived from the Pivotal Tracker's username value and the JIRA username is derived from this Full Name (made lower-case).
   - Specify the Full Name in JIRA to change a Pivotal Tracker's user's name details. The JIRA username is derived from this Full Name (made lower-case).

9. Click the **Begin Import** button when you are ready to begin importing your Pivotal Tracker 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 Pivotal Tracker Import Wizard process.
   - If you need to import data from another Pivotal Tracker project or site with the same (or similar) settings to what you used through this procedure, click the [save the configuration](#) link to download a Pivotal Tracker configuration file, which you can use at the first step of the Pivotal Tracker Import Wizard.

Congratulations, you have successfully imported your Pivotal Tracker project(s) into JIRA! If you have any questions or encounter any problems, please contact [Atlassian support](#).

**Tips for importing Pivotal Tracker data into JIRA fields**

The import process converts Pivotal Tracker data as follows:

<table>
<thead>
<tr>
<th>Pivotal Tracker</th>
<th>JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Project</td>
<td>Each Pivotal Tracker project is imported into a new JIRA project. You can optinally import into an existing project if you have used the importer before.</td>
</tr>
<tr>
<td>Story</td>
<td>Issue</td>
<td>Pivotal Tracker story types are recreated in JIRA.</td>
</tr>
<tr>
<td>Summary</td>
<td>Summary</td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>Comments</td>
<td></td>
</tr>
<tr>
<td>Attachments</td>
<td>Attachments</td>
<td>Attachments are extracted from the Pivotal Tracker database and saved to disk. The dates and user attaching the attachments will be retained.</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
<td>JIRA will recreate the Pivotal Tracker workflow and statuses during import.</td>
</tr>
<tr>
<td>Labels</td>
<td>Labels</td>
<td>Pivotal Tracker labels with spaces are imported with underscores (JIRA does not support spaces in labels).</td>
</tr>
<tr>
<td>Story ID</td>
<td>Story ID and Story URL</td>
<td>JIRA will create these as custom fields.</td>
</tr>
<tr>
<td>Iterations</td>
<td>Fix Version/s</td>
<td>Past iterations in Pivotal are imported as released versions in JIRA.</td>
</tr>
<tr>
<td>Story Estimates</td>
<td>Story Points</td>
<td></td>
</tr>
<tr>
<td>Order of stories</td>
<td>Rank</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>You will need to configure this custom field in JIRA after the import. If you are using JIRA Agile, you may wish to activate issue ranking. This can be done either before or after importing your Pivotal Tracker data.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Tracker</td>
<td>Time Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>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 &quot;Placeholder for imported time tracking data&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>User</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The importer will automatically create JIRA users for any Pivotal Tracker users who do not exist in JIRA.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- 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.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- 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.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- 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.</td>
<td></td>
</tr>
<tr>
<td>User Roles</td>
<td>Project Roles</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Viewer = User ; Member = Developers ; Owner = Administrators</td>
<td></td>
</tr>
</tbody>
</table>

**Importing Data from Trac**

The [JIRA Importers plugin](https://confluence.atlassian.com/display/JIRA/JIRA+Importers+Plugin), which is bundled with JIRA, allows you to import data from Trac from a compressed Trac environment.

Our main website highlights some top reasons why people [migrate from Trac to JIRA](https://confluence.atlassian.com/display/JIRA/JIRA+Importers+Plugin).

Version 2.6.1 or later of the JIRA Importers Plugin is compatible with Trac version 0.12.2.

**On this page:**
- Preparing Trac data for import into JIRA
- Running the Trac Import Wizard
- Tips for importing Trac data into JIRA fields
Preparing Trac data for import into JIRA

Compress your Trac environment:

1. Access your Trac environment.
2. If you use SQLite (the Trac default), PostgreSQL or MySQL for your Trac database, ensure your database URL (defined in Trac's conf/trac.ini file) is also reachable from JIRA server (using 'localhost' or a UNIX socket will not work).
3. Zip the contents of Trac Environment without any leading directories.

Running the Trac Import Wizard

Before you begin: If your JIRA installation has existing data — Back up 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 Pivotal Tracker option to open the Trac Import Wizard: Setup page.
   ✔️ Keyboard shortcut: g + g + start typing external system import
3. On the Trac Import Wizard: Setup page, select your compressed Trac environment file, which you prepared above.
4. Leave the Use an existing configuration file 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 fields in Trac and those in JIRA.
   - 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 Trac Import Wizard, JIRA will create a configuration file which you can use for subsequent Trac imports (for re-use at this step of the Trac Import Wizard).
5. Click the Next button to proceed to the Setup project mappings step of the Trac Import Wizard.
6. On the Setup project mappings page, select which Trac projects you wish to import into JIRA.
   - 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:
     i. A new project Name.
     ii. A new project Key.
        ✔️ This will be used as the prefix for all issue IDs in your JIRA project.
     iii. The Project Lead.
7. Click the Next button to proceed to the Setup custom fields step of the Trac Import Wizard.
   ✔️ This step will almost always appear because at least one Trac field is not likely match an existing JIRA field.
8. 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.
9. 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).
10. 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.
    - '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.
    - '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.

11. 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.

Tips for importing Trac data into JIRA fields

The import process converts Trac data as follows:

<table>
<thead>
<tr>
<th>In Trac</th>
<th>In JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Environment</td>
<td>Project</td>
<td>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.</td>
</tr>
<tr>
<td>Ticket Type</td>
<td>Issue Type</td>
<td>You can configure mapping of Trac Ticket Types to specific JIRA Issue Types.</td>
</tr>
<tr>
<td>Ticket #</td>
<td>External Issue ID</td>
<td>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.</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
<td>You can configure mapping of specific Trac values to specific JIRA values.</td>
</tr>
<tr>
<td>Summary</td>
<td>Summary</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>Versions</td>
<td>Versions</td>
<td>Versions are imported from Trac (if you choose), and are set to the Un-Released and Un-Archived state.</td>
</tr>
<tr>
<td>Component</td>
<td>Components</td>
<td>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.</td>
</tr>
<tr>
<td>Comments</td>
<td>Comments</td>
<td></td>
</tr>
<tr>
<td>Priority</td>
<td>Priority (or a custom field)</td>
<td>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.</td>
</tr>
<tr>
<td>Field</td>
<td>JIRA Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Severity</td>
<td>Priority (or a custom field)</td>
<td>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.</td>
</tr>
<tr>
<td>Milestone</td>
<td>Milestone</td>
<td>JIRA will create this as a custom field.</td>
</tr>
<tr>
<td>Attachments</td>
<td>Attachments</td>
<td>Attachments are extracted from the Trac Environment and saved to disk. To specify the location on disk, see <a href="#">Configuring File Attachments</a>.</td>
</tr>
<tr>
<td>Resolution</td>
<td>Resolution</td>
<td>You can configure mapping of specific Trac values to specific JIRA values.</td>
</tr>
<tr>
<td>CC</td>
<td>Watcher</td>
<td></td>
</tr>
<tr>
<td>Keywords</td>
<td>Labels</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>User</td>
<td>The importer will automatically create JIRA users for any Trac users who do not exist in JIRA.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Passwords from Trac are not imported. Users from Trac will need to get their passwords emailed to them.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If you are using <a href="#">External User Management</a>, 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td>Other fields</td>
<td>Custom fields</td>
<td>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.</td>
</tr>
</tbody>
</table>
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 feature allows you to import issues from an external (issue tracking) system which:

- JIRA does not provide a dedicated import tool for and
- Can export its data in a structured/tabulated format (preferably CSV).

Our main website highlights some top reasons why people migrate from such an external issue tracking system to JIRA.

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.
   - At the end of the CSV file import wizard, you will be given the option of creating a CSV configuration file, which contains the settings you configured whilst running through the CSV file import wizard. This is useful if you need to test your CSV file import on a test JIRA server first before performing the import on a production system.

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.

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:
Encapsulating JIRA data structure in your CSV file

Capturing data that spans multiple lines

Use double-quote marks ("") in your CSV file to capture data that spans multiple lines. For example, upon import, JIRA will treat the following as a valid CSV file with a single record:

<table>
<thead>
<tr>
<th>Summary, Description, Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Login fails&quot;, &quot;This is on a new line&quot;, Open</td>
</tr>
</tbody>
</table>

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"
- Discarding 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:

<table>
<thead>
<tr>
<th>IssueType, Summary, FixVersion, FixVersion, FixVersion, Component, Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>bug, &quot;First issue&quot;, v1, v1, Component1, Component1, Component1, Component1</td>
</tr>
<tr>
<td>bug, &quot;Second issue&quot;, v2, v2, Component1, Component2, Component2, Component2</td>
</tr>
<tr>
<td>bug, &quot;Third issue&quot;, v1, v2, v3, Component1, Component1, Component1</td>
</tr>
</tbody>
</table>

In the above example, the Component field of the second issue and the Fix Version field of the third issue will generate multiple values in appropriate JIRA fields upon import.

⚠ 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"
Admin, "CSV attachment import with timestamp,author and filename", "Please check
the attached image below.", "01/01/2012
13:10;Admin;image.png;file://image-name.png", "01/01/2012 10:10;Admin; This comment
works"

URLs for attachments support the HTTP and HTTPS protocols and can be any location that your JIRA server
must be able to access. You can also use the FILE protocol to access files in the import/attachments subdir-
rectory of your JIRA Home Directory.

Creating sub-tasks
You can create sub-tasks of issues through a CSV file import, by encapsulating this structure in your 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:

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug, &quot;First issue&quot;, v1, , Component1, , 1, , jbloggs</td>
</tr>
<tr>
<td>Bug, &quot;Second issue&quot;, v2, , Component1, Component2, 2, , fferdinando</td>
</tr>
<tr>
<td>Bug, &quot;Third issue&quot;, v1, v2, v3, Component1, , 3, , fferdinando</td>
</tr>
<tr>
<td>Sub-task, &quot;Fourth issue&quot;, v1, v2, , Component2, , 2, jbloggs</td>
</tr>
</tbody>
</table>

In the example above, the fourth issue will be sub-task of the second issue upon import, assuming you match
the 'Issue ID' and 'Parent ID' fields in your CSV file to the Issue Id and Parent Id JIRA fields, respectively during
the CSV file import wizard.

Importing issues into multiple JIRA projects
You can import issues from your CSV file into different JIRA projects through a CSV file import. To do this:

- Your CSV file requires two additional columns whose headings should be named similarly to Project
  Name and Project Key.
- Ensure that every issue represented in your CSV file contains the appropriate name and key in these
  columns for the JIRA projects to which they will be imported.

  The project name and key data is the minimum JIRA project data required for importing issues from a
  CSV file into specific JIRA projects.

<table>
<thead>
<tr>
<th>IssueType, Summary, Project Name, Project Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>bug, &quot;First issue&quot;, Sample, SAMP</td>
</tr>
<tr>
<td>bug, &quot;Second issue&quot;, Sample, SAMP</td>
</tr>
<tr>
<td>task, &quot;Third issue&quot;, Example, EXAM</td>
</tr>
</tbody>
</table>

In the example above, the first and second issues will be imported into the 'Sample' project (with project key
'SAMP') and the third issue will be imported into the 'Example' project (with project key 'EXAM'), assuming you
match the 'Project Name' and 'Project Key' fields in your CSV file to the Project name and Project key JIRA
fields, respectively during the CSV file import wizard.
How to handle unresolved issues

For fields mapping to Resolution, Priority and Issue Type, you will get a select list with the available values in JIRA. In addition, you can quickly create values that do not exist in JIRA by clicking the green plus symbols.

For fields mapping to Status, you will get the select list with JIRA’s available values, but no plus symbol for creating new status values.

For these four fields, there are two special options in the select list in addition to JIRA’s available values:

- ‘Import as blank’ — this causes the JIRA value to be blank for that field. Note that, if you are importing Unresolved issues, you should create a field mapping for the Resolution field and set the value ‘Unresolved’ to ‘Import as blank’.
- ‘No mapping’ — this attempts to import the value in the CSV file as-is. Note that using ‘No mapping’ for a field value will result in a failed import if the value is not valid for that JIRA field. For fields mapping to Status and Issue Type, default values are used when the ‘Import as blank’ option is selected.

Importing worklog entries

Your CSV file can contain worklog entries. For example:

```
Summary,Worklog
Only time spent (one hour),3600
With a date and an author,2012-02-10 12:30:10;wseliga;120
With an additional comment,Testing took me 3 days;2012-02-10
12:30:10;wseliga;259200
```

To track time spent, you need to use seconds.

Updating existing issues

From version 4.3 of JIRA Importers plugin you can update existing issues. Your CSV file needs to contain a column that during the import wizard is mapped to Issue Key. If an issue exists for a given key it will be updated. For example:

```
isssue key,summary,votes,labels,labels
TT-1,Original summary,1,label1,label2
TT-1,,7,label1,label2
TT-1,Changed summary,,,TT-2,Original summary,2,label1,label2
TT-2,<<!clear!>>,<<!clear!>>,<<!clear!>>,<<!clear!>>,<<!clear!>>,
```

First row will create an issue, second row will set votes to 7, and add two labels. Following row will change the summary. Issue TT-2 will be created with two labels, the second row will remove those labels with a special marker <<!clear!>>.

⚠ Importing a CSV to update existing issues will reset columns to their default values if they are not specified in the CSV.

Running the CSV file import wizard

**Before you begin:** If your JIRA installation has existing data — Back up 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](image)
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:](image)
   - The file will be imported using the File encoding you specify here (which is UTF-8 by default).
4. Leave the Use an existing configuration file 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 either one JIRA project (new or existing), or multiple JIRA projects (by ensuring that your CSV file includes the minimum JIRA project data required — i.e. the JIRA project name and key). Complete the following fields/options:

<table>
<thead>
<tr>
<th>Import to JIRA Project</th>
<th>Choose either of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Select a project and then do either of the following:</td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td>• Select Create New from the dropdown menu and in the resulting Add A New Project dialog box, type the following:</td>
</tr>
<tr>
<td></td>
<td>a. A new project Name</td>
</tr>
<tr>
<td></td>
<td>b. A new project Key</td>
</tr>
<tr>
<td></td>
<td>This will be used as the prefix for all issue IDs in your JIRA project.</td>
</tr>
<tr>
<td></td>
<td>c. The Project Lead.</td>
</tr>
<tr>
<td></td>
<td>• Defined in CSV. Ensure that every issue in your CSV file includes data for the JIRA Project Name and Project Key.</td>
</tr>
<tr>
<td></td>
<td>This option is useful if you want to import issues from your CSV file into multiple JIRA projects. See Importing issues into multiple JIRA projects for details.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>E-mail Suffix for New Users</th>
<th>Enter the email address domain for any new users specified in the CSV file which will be added to JIRA during the import.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date format in import file</td>
<td>Specify the date format used in your CSV file. Use the syntax that complies with the Java SimpleDateFormat.</td>
</tr>
</tbody>
</table>

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). For more information about matching CSV fields to JIRA fields, see Tips for importing CSV data into JIRA fields below.

   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.
- If you are importing sub-tasks, remember to match the Issue ID and Parent ID fields in JIRA to those in your CSV file.
- If you are importing issues into multiple projects, ensure that you selected Defined in CSV during the Setup project mappings step above and remember to match the Project Name and Project Key fields in JIRA to those in your CSV file.

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, specify the JIRA field values for each CSV file field value (which has been detected by the CSV file import wizard).

Please Note:

- 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.
- 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 select the Map field value check box, 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:

<table>
<thead>
<tr>
<th>JIRA Field</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>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.</td>
</tr>
<tr>
<td>Summary</td>
<td>This is the only required field.</td>
</tr>
<tr>
<td>Issue Key</td>
<td>You can set the issue key for an imported issue. If an issue with a given key already exists in JIRA, it will be updated instead.</td>
</tr>
<tr>
<td>Component(s)</td>
<td>You can import issues with multiple components by entering each component in a separate column.</td>
</tr>
<tr>
<td>Affects Version(s)</td>
<td>You can import issues with multiple 'Affects Versions' by entering each version in a separate column.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Fix Version(s)</td>
<td>You can import issues with multiple 'Fix Versions' by entering each version in a separate column.</td>
</tr>
<tr>
<td>Comment Body</td>
<td>You can import issues with multiple comments by entering each comment in a separate column.</td>
</tr>
<tr>
<td>Date Created</td>
<td>Please use the date format specified on the second step of the CSV import wizard.</td>
</tr>
<tr>
<td>Date Modified</td>
<td>Please use the date format specified on the second step of the CSV import wizard.</td>
</tr>
<tr>
<td>Due Date</td>
<td>Please use the date format specified on the second step of the CSV import wizard.</td>
</tr>
<tr>
<td>Issue Type</td>
<td>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 <a href="#">Defining 'Issue Type' Field Values</a>. You can also create new JIRA values on-the-fly during the import process.</td>
</tr>
<tr>
<td>Labels</td>
<td>You can import issues with multiple labels by entering each label in a separate column.</td>
</tr>
<tr>
<td>Priority</td>
<td>If not specified in your CSV file, imported issues will be given the default (i.e. first) Priority as specified in your JIRA system <a href="#">Defining 'Priority' Field Values</a>. You can also create new JIRA values on-the-fly during the import process.</td>
</tr>
<tr>
<td>Resolution</td>
<td>If not specified in your CSV file, imported issues will be given the default (i.e. first) Resolution as specified in your JIRA system <a href="#">Defining 'Resolution' Field Values</a>. You can also create new JIRA values on-the-fly during the import process. Also see <a href="#">How to handle unresolved issues</a> for helpful tips.</td>
</tr>
<tr>
<td>Status</td>
<td>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.</td>
</tr>
<tr>
<td>Original Estimate</td>
<td>The value of this field needs to be specified as number of seconds.</td>
</tr>
<tr>
<td>Remaining Estimate</td>
<td>The value of this field needs to be specified as number of seconds.</td>
</tr>
<tr>
<td>Time Spent</td>
<td>The value of this field needs to be specified as number of seconds.</td>
</tr>
</tbody>
</table>
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

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:
There are a few possible reasons:

- The format of dates is not correctly set in the import configuration file. The date format for custom fields must match the "Date format in input file" which has a default format of `yyyyMMddHHmmss`.
- JIRA system date fields such as Created, Updated and Due Date use "yyyy-MM-dd HH:mm:ss" but may need an offset adding.
- Date Picker and Date Time Picker formats are not consistent, e.g.

```java
jira.date-picker.java.format=dd/MMM/yy
jira.date.time-picker.java.format=dd/MMM/yy hh:mm a
```

should be corrected to,

```java
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"

Importing Data from JSON (beta release)

⚠️ The JSON importer in still in beta and is not yet fully supported. For assistance, please post the problem description on Atlassian Answers with a sample JSON file that doesn't work, ensuring to add the "import" tag to it.

The JIRA Importers plugin, which is bundled with JIRA, allows you to import data from a JavaScript Object Notation (JSON) file.

JSON files are easy to read and are capable of encapsulating more structure and information than CSV files. They can also be used to populate data from other systems.

The JIRA Importers plugin comes with an export option that allows you to export your existing issues to JSON formatted files, which can be used to copy issues between different JIRA servers, or to prepare templates that can be used to populate new projects.

ℹ️ This is an experimental feature so feel free to contact us and share your story or ideas how this feature can help you.

You can generate JSON file with JIRA Importers plugin version 4.3 or later.

⚠️ Please note that the import issue format used by JIRA Importers plugin is a simplified version when compared to the version returned from the JIRA REST API.
If you want to prepare the JSON file yourself follow this pattern:

```
{
    "users": [
        {
            "name": "alice",
            "fullname": "Alice Foo"
        },
        {
            "name": "bob",
            "fullname": "Bob Bar"
        }
    ],
    "links": [
        {
            "name": "sub-task-link",
            "sourceId": "2",
            "destinationId": "1"
        },
        {
            "name": "Duplicate",
            "sourceId": "3",
            "destinationId": "2"
        }
    ],
    "projects": [
        {
            "name": "A Sample Project",
            "key": "ASM",
            "description": "JSON file description",
            "versions": [
                {
                    "name": "1.0",
                    "released": true,
                    "releaseDate": "2012-08-31T15:59:02.161+0100"
                },
                {
                    "name": "2.0"
                }
            ],
            "components": [
                "Component",
                "AnotherComponent"
            ],
            "issues": [
                {
                    "priority": "Major",
                    "description": "Some nice description here\nMaybe _italics_ or *bold*?",
                    "status": "Closed",
                    "reporter": "alice",
                    "labels": [ "impossible", "to", "test" ],
                    "watchers": [ "bob" ],
                    "issuetype": "Bug"
                }
            ]
        }
    ]
}
```
"resolution" : "Resolved",
"created" : "2012-08-31T17:59:02.161+0100",
"updated" : "P-1D",
"affectedVersions" : [ "1.0" ],
"summary" : "My chore for today",
"assignee" : "bob",
"fixedVersions" : [ "1.0", "2.0" ],
"components" : [ "Component", "AnotherComponent" ],
"externalId" : "1",
"history" : [ {
  "author" : "alice",
  "created": "2012-08-31T15:59:02.161+0100",
  "items": [ {
    "fieldType" : "jira",
    "field" : "status",
    "from" : "1",
    "fromString" : "Open",
    "to" : "5",
    "toString" : "Resolved"
  }
  ]
},
"customFieldValues": [ {
  "fieldName": "Story Points",
  "fieldType": "com.atlassian.jira.plugin.system.customfieldtypes:float",
  "value": "15"
}, {
  "fieldName": "Business Value",
  "fieldType": "com.atlassian.jira.plugin.system.customfieldtypes:float",
  "value": "34"
} ],
"attachments" : [ {
  "name" : "battarang.jpg",
  "attacher" : "admin",
  "created" : "2012-08-31T17:59:02.161+0100",
  "description" : "This is optimus prime"
} ]
],
{ "status" : "Open",
"reporter" : "bob",
"issueType" : "Sub-task",
"created" : "P-3D",
"updated" : "P-1D",
"summary" : "Sub-task",
"externalId" : "2"
},
{ "status" : "Closed",
"reporter" : "alice",
"issueType" : "Sub-task",}
"created": "P-3D",
"updated": "P-1D",
"resolution": "Duplicate",
"summary": "Duplicate Sub-task",
"externalId": "3"}
]
Running the JSON Import Wizard

If your JIRA installation has existing data, then before you begin, back up 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 JSON option to open the JSON Import Wizard: Setup page.
   - Keyboard shortcut: g + g + start typing external system import
3. 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 JSON projects into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Importing Data from Redmine

The JIRA Redmine Importer plugin allows you to import data from the Redmine Issue Tracker product into your local JIRA site.

Version 5.0.2 or later of the JIRA Importers Plugin is compatible with Redmine versions 1.3.0+ and 2.0+.

Before you begin

- Ensure that you are using Redmine versions 1.3.0+ and 2.0+.
- Ensure that you are using version 5.0.2 or later of the JIRA Importers Plugin. This plugin is bundled with JIRA. For instructions on how to update a plugin, see Updating Add-ons.
- Install the JIRA Redmine Importer plugin, if you haven't installed it already. For instructions on how to install a plugin, see Installing Add-ons.
- Enable the REST web service in Redmine in Administration > Settings > Authentication > Enable REST web service, if you haven't already enabled it.

Dates can be represented in SimpleDateFormat "yyyy-MM-dd'T'HH:mm:ss.SSSZ" (example output: "2012-08-31T15:59:02.161+0100") or you can use relative dates like "P-1D" (which means one day ago).

If your JIRA installation has existing data — Back up your existing JIRA data.

Tip: Redmine supports hierarchical issues. During the Redmine Import Wizard, you are given the option to recreate this issue hierarchy through JIRA issue links. Therefore, before importing Redmine data, you may want to configure a custom issue link to replicate this hierarchy. For example:

- Name — 'Hierarchy'
- Outward Link Description — ‘parent of’
- Inward Link Description — ‘child of’

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Import your Redmine data

The JIRA Redmine Importer plugin provides a wizard that walks you through the process of importing data and integrating it with JIRA. To access the import wizard:

1. Log into JIRA as a user with the **JIRA Administrators** global permission.
2. Choose > **System**. Select **Import & Export** > **External System Import** to open the Import external projects page.
   - Keyboard shortcut: g + g + start typing redmine
3. Select the **Import** button associated with the Redmine option.
4. Complete the fields as prompted in the wizard.

If you are importing your Redmine issues into an existing JIRA project, you must choose the JIRA workflow scheme used by that existing JIRA project when you are prompted to select the workflow scheme. Otherwise, your import may not complete successfully.

Please note that it is mandatory to map Redmine status field to a specific JIRA status field and Redmine tracker field to a JIRA issue type field since these JIRA fields are an integral part of JIRA workflows.

Tips for importing Redmine On Demand data into JIRA fields

The import process converts Redmine data as follows:

<table>
<thead>
<tr>
<th>In Redmine</th>
<th>In JIRA</th>
<th>Import Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Project</td>
<td>Redmine 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.)</td>
</tr>
<tr>
<td>Target Version</td>
<td>Affects Version</td>
<td>Redmine target version is mapped to JIRA “affects version”.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority</td>
<td>You can configure mapping of specific Redmine values to specific JIRA values.</td>
</tr>
<tr>
<td>Summary</td>
<td>Subject</td>
<td>Redmine subject is imported as the JIRA issue summary.</td>
</tr>
<tr>
<td>Worklog</td>
<td>Worklog</td>
<td>See Configuring Time Tracking.</td>
</tr>
<tr>
<td>Author</td>
<td>Reporter</td>
<td>Redmine issue author is mapped as JIRA Issue Reporter.</td>
</tr>
<tr>
<td>Attachments</td>
<td>Attachments</td>
<td>Attachments are extracted from Redmine and saved. Information on the date the file was attached and the user who attached it is retained, as well. To specify the location where the attachments are stored, see Configuring File Attachments.</td>
</tr>
<tr>
<td>Tracker</td>
<td>Issue Type</td>
<td>You can configure the mapping of specific trackers to specific JIRA issue types.</td>
</tr>
<tr>
<td>Category</td>
<td>User</td>
<td>User</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>You can choose to have the importer automatically create JIRA users for any Redmine users who do not already exist in JIRA.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Users who interacted with the Redmine system will be created as active accounts in JIRA. Other users will be imported into a special group called “fredmine-import-unuse d-users” and will be deactivated.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Passwords from Redmine are not imported. Users from Redmine must have their passwords emailed to them the first time they log into JIRA.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• 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 starting the import.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• 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 open and list the users that can't be created.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Priority</th>
<th>Priority</th>
<th>You can configure the mapping of specific Redmine values to specific JIRA values.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Status</td>
<td>You can configure the mapping of specific Redmine values to specific JIRA values, provided you create your workflows in JIRA before running the importer.</td>
</tr>
<tr>
<td>• The JIRA status field is integral to JIRA workflow. To learn more, see What is Workflow.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• To create a JIRA workflow, see Configuring Workflow.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• To create a JIRA workflow scheme (which you can then associate with appropriate projects and Issue Types), see Activating workflow.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Component/s</th>
<th>Component/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>This mapping is hard-coded and cannot be changed.</td>
<td>This mapping is hard-coded and cannot be changed.</td>
</tr>
</tbody>
</table>
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.

**Archiving a Project**

**Request for interest**

Hi there,

Great to see your interest in archiving JIRA projects!

Atlassian is looking to build a JIRA Archiver solution that will give JIRA admins the ability to archive whole projects or a subset of issues and their configurations to a separate server. This solves the problem that many large Enterprise customers face, having old projects/issues that are not used anymore, however these projects/issues can not be deleted because they may be needed for future reference or for compliance reasons.

At the moment we are gathering names of interested customers who would like to be contacted when we have a solution ready for testing. If you would to be part of our JIRA Archiver early access program please send an email to roy@atlassian.com with the subject: **JIRA Archiver** and I will respond accordingly.

Cheers,

Roy Krishna
JIRA Product Management

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

**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 no one 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).
3. To prevent workflow transitions from happening you will need to update the workflow and add a condition to each transition. The conditions should check that a user has the Edit Issues permission.

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.

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.
   - Please note that deleting the Project will result in all the attachments also getting deleted from the JIRA A Home Directory. Please ensure that the attachments are copied to the test instance before deleting the project.

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.

Archiving issues

Archiving issues is also possible. The basic method would be to filter for issues that you want to archive then bulk move them into a separate project which can then be archived by using one of the methods above.
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.

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.
9. Generate new Server ID for the newly installed JIRA instance as described in the article Changing Server ID. This step is needed if you plan to create Application Links between the JIRA instances.

Integrating with a Source Control System

JIRA can be easily integrated with many popular source control systems:

- Integrating JIRA with FishEye
- 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.

Integrating JIRA with FishEye

Integration of JIRA with Atlassian FishEye allows you to track in JIRA the source code and changesets that correspond to your stories in JIRA, when you are using FishEye with your source-control repository. FishEye integration is implemented as a plugin that 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:
- Initial Configuration
- 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. (Optional) Subscribe JIRA to FishEye Gadgets
- Notes
Initial Configuration

You should check that JIRA is configured to accept remote API calls; in JIRA, go to Administration > System > General Configuration and turn on Accept remote API calls (under ‘Options’). A restart is not required. See JIRA options for more detail.

If you set up your Application Links prior to enabling either Remote API, please re-create both the JIRA and FishEye Application Links.

Note, we do not recommend the use of project links with FishEye 2.9 and later, if you have JIRA 5.0 or later as well as the latest version of the JIRA FishEye Plugin. This is because application links now provide all of the functionality previously available with project links. However, project links are retained in FishEye and Crucible for the following reasons:

- Setting up project links provides a way to restrict the scope of JIRA searches, which can provide performance benefits.
- Legacy configurations can continue to use project links without any need for changes.
- Third-party plugins may continue to rely on project links for their functionality.

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.

You require JIRA System Administration permissions in order to perform this procedure.

To create an application link between a JIRA and a FishEye/Crucible server:

1. Log in to JIRA as a user with the JIRA Administrators permissions.
2. Choose > Add-ons. Select Application Links in the Add-ons panel to open the configuration screen.
3. 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. Choose > Add-ons. Select FishEye Configuration to open the JIRA FishEye configuration screen.
3. Click Edit Primary Configuration.
4. 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.
5. Click Update.
6. Click Application Links Configuration at the bottom of the screen.
7. Click Trusted Applications in the Outgoing Authentication column for your FishEye/Crucible link.
8. Click Modify and enter the following information:
   - **IP Patterns**: Enter the IP addresses for your FishEye/Crucible instance, one per line:
     
     ```
     127.0.0.1
     172.20.5.95
     ```

9. Click Apply.
10. Click Incoming Authentication in the left menu.
11. Click Modify and enter the following information:
   - **IP Patterns**: Enter the IP addresses for your JIRA instance, one per line:
     
     ```
     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
     ```

12. Click Apply > 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 Project pages and tickets:

2. Choose `Projects`. (In this case, select the project that you want to associate with a FishEye repository/Crucible project.)
3. 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.
4. On the right-upper corner, choose Actions > Edit Permissions.
5. Locate the 'View Development Tools' 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. (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 > Add Gadget** in JIRA's top navigation bar.
2. The 'Gadget Directory' popup window will appear. Click **Gadget Subscriptions**.
3. The 'Gadget Subscriptions' popup window will appear. Click **Add Subscription**.
4. 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.
5. Click **Add Subscription**.
6. 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 behavior 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 Subversion**

JIRA's Subversion integration lets you see Subversion commit information relevant to each issue. Subversion integration can be implemented either by using Atlassian FishEye or the Subversion add-on. The FishEye integration offers greater scalability, insight and flexibility into your source code and related integration with JIRA, however both solutions allow you to link JIRA to related code changes in Subversion.

Commits will appear in this tab if the commit log mentions the issue key ('TEST-3' above).

**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).

**Integrating JIRA with ClearCase**

Atlassian FishEye has ended support for ClearCase and unfortunately there are no plans to put this back into the product. Following this there are not current ways to integrate ClearCase into the current version of JIRA as there are no ClearCase integration plugins that currently exist. We would instead encourage a move to a modern version control system such as GIT. For more information on GIT please see our Free Tutorials.

**Integrating with a Build Management System**

JIRA integrates with Bamboo, Atlassian's Continuous Integration server.

- Integrating JIRA with Bamboo

**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.

**Configuring Global Settings**

- Configuring Time Tracking
- Configuring JIRA Options
  - Configuring Advanced Settings
- 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 Issue Cloning
- Configuring Issue Linking
- Configuring the Whitelist
- Configuring Sub-tasks
- Managing Shared Filters
- Managing Shared Dashboards
- Linking to Another Application

**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.

ℹ️ 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. Choose **System > Issue Features > Time Tracking** to open the Time Tracking page.
3. Choose the 'Deactivate' 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

Time tracking add-ons for JIRA in the Atlassian Marketplace extend JIRA's time tracking power. Check them out here.

Enabling Time Tracking

To enable Time Tracking:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Choose **System > Issue Features > Time Tracking** to open the Time Tracking page.
3. Choose 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. Choose ![System] > **System**. Select **Issue Features > Time Tracking** 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**

   - **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.**

5. 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.

12. If it is needed to enter the 'Original Estimate' during issue creation or during issue editing, ensure that
the field 'Time Tracking' is added to the relevant screens associated with those operations. Refer Associating a Screen with an Issue Operation for more details.

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.

- With Legacy Mode enabled, if you change the Remaining Estimate field in a workflow post function the Original Estimate is also cleared. This issue is tracked at JIRA-25031

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 customized for use within your organization. These options can be accessed and edited on JIRA's 'General Configuration' page.

On this page:
- Editing JIRA's General Configuration
  - Settings
  - Internationalization
  - Options

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. Choose ☀️ > System. Select General Configuration to open the Administration page.
3. Scroll to the end of the page and click the Edit Configuration button to edit the three sections as described below:
   - Settings
   - Internationalization
   - Options

Screenshot 1: General Configuration
**Settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| **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.                                                                                      |
<p>| <strong>Mode</strong>              | JIRA can operate in two modes:                                                                                                              |
|                       | - Public — Anyone can sign themselves up with self-registration 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                                                                                                                             |</p>
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Authentication Attempts Allowed</td>
<td>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). <em>Default: 3 (for new installations of JIRA)</em></td>
</tr>
<tr>
<td>CAPTCHA on signup</td>
<td>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. <em>Default: ON</em></td>
</tr>
<tr>
<td>Base URL</td>
<td>The base URL of this JIRA installation. ![Warning] 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.</td>
</tr>
<tr>
<td>Email from</td>
<td>Specifies the From: header format in notification emails. Default is of the form &quot;John Doe (JIRA) <a href="mailto:jira@company.com">jira@company.com</a>&quot;. Available variables are <code>${fullname}</code>, <code>${email}</code> and <code>${email.hostname}</code>. Note that the actual address (e.g. '<a href="mailto:jira@company.com">jira@company.com</a>') cannot be specified here - it is determined by the mail server or individual project configuration.</td>
</tr>
<tr>
<td>Introduction</td>
<td>A short introduction message displayed on the dashboard. ![Warning] Also see the announcement banner, which is displayed on every JIRA page. You can include HTML, but ensure all tags are correctly closed.</td>
</tr>
</tbody>
</table>

**Internationalization**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
Indexing language

JIRA uses Lucene, a high-performance text search engine library, in full-text searches for issues stored in JIRA. This option is designed to enhance JIRA’s search indexing and issue searching features for issues entered in the languages available in this list. Hence, choose the language that matches the language used in your issues.

Choosing a specific language in this list has the following effects when conducting searches in JIRA (with respect to your chosen language):

- **Reserved words** in text fields will not be indexed.
- **Stemming of words** in all JIRA fields will be active.

If multiple languages are used in your issues (or you wish to disable the two effects above), choose Other.

ℹ️ 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 Internationalization.)

## 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 time zone by using the Time Zone setting in their user profile. (To choose the time format see Configuring the Layout and Design.)

ℹ️ 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

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| Allow users to vote on issues        | Controls whether voting is enabled in JIRA. Voting allows users to indicate whether 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                          |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum project name size</td>
<td>Controls the maximum number of characters allowed for a project name. Changing this value will not affect the names of existing projects. Default: <strong>80</strong></td>
</tr>
<tr>
<td>Maximum project key size</td>
<td>Controls the maximum number of characters allowed for a project key. Changing this value will not affect the keys of existing projects. You can set this to any value between 2 and 255, inclusive. Default: <strong>10</strong></td>
</tr>
<tr>
<td>Allow unassigned issues</td>
<td>When turned <strong>ON</strong>, the default assignee for the project is <strong>Unassigned</strong>. When turned <strong>OFF</strong>, issues must always be assigned to someone - by default, the assignee will be the <strong>Project Lead</strong> as defined for each project. Default: <strong>ON</strong></td>
</tr>
<tr>
<td>External user management</td>
<td>When turned <strong>ON</strong>, 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 <strong>project roles</strong>, and create/edit/delete <strong>user properties</strong>. Additionally, JIRA will not display options for users to change their <strong>password</strong>, or edit their <strong>profile</strong>. Generally you would only turn this <strong>ON</strong> if you are managing all your users from outside JIRA (e.g. using Crowd, Microsoft Active Directory or another LDAP directory). Default: <strong>OFF</strong></td>
</tr>
<tr>
<td>Logout confirmation</td>
<td>Controls whether to obtain user’s confirmation when logging out: <strong>NEVER COOKIE</strong> - prompt for confirmation if the user was automatically logged in (via a cookie). <strong>ALWAYS</strong> Default: <strong>NEVER</strong></td>
</tr>
<tr>
<td>Use gzip compression</td>
<td>Controls whether to compress the web pages that JIRA sends to the browser. It is recommended that this be turned <strong>ON</strong>, unless you are using mod_proxy. Default: <strong>OFF</strong></td>
</tr>
<tr>
<td>Accept remote API calls</td>
<td>Controls whether to allow remote client access (via XML-RPC or SOAP) to this JIRA installation, for authenticated users. Default: <strong>OFF</strong></td>
</tr>
<tr>
<td>User email visibility</td>
<td>Controls how users’ email addresses are displayed in the user profile page. - <strong>PUBLIC</strong> - email addresses are visible to all. - <strong>HIDDEN</strong> - email addresses are hidden from all users. - <strong>MASKED</strong> - the email address is masked (e.g. '<a href="mailto:user@example.com">user@example.com</a>' is displayed as 'user at example dot com'). - <strong>LOGGED IN USERS ONLY</strong> - only users logged in to JIRA can view the email addresses. Default: <strong>PUBLIC</strong></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 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 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 appropriate in JIRA, e.g. on Login panels and pages). Applies only if outgoing email is enabled. Can be used with or without the custom 'Contact Administrators Message' below. Users with the JIRA Administrators global permission (not JIRA System Administrators - see JIRA-27454 for details) will be notified as a result of this feature being used.  
**Default:** OFF |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact Administrators Message</td>
<td>Displays a custom message when users click the 'Contact Administrators' link (which appears when appropriate in JIRA, e.g. on Login panels and pages). 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.</td>
</tr>
</tbody>
</table>
| Use Gravatar for user avatars                | Enables users to use Gravatars in their user profile in stead of JIRA-specific avatars. Users will not be able to use JIRA-specific avatars if Gravatars are enabled, and vice versa.  
**Default:** OFF                                                                                                                                  |
| Auto Update Search Criteria                  | Enables search results to be automatically updated when criteria are modified in a basic search.                                                                                                           |
| Show product recommendations                 | Enables you to turn on/off recommendations for other Atlassian products on your JIRA site.                                                                                                                      
**Default:** ON                                                                                                                                      |
| Enable Atlassian analytics                   | Enables you to turn on/off Atlassian analytics; see Data collection policy for more information.                                                                                                          
**Default:** OFF                                                                                                                                      |

### 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. You must be a JIRA System Administrator to do this.

### Editing JIRA’s advanced settings

To access and edit options on the 'Advanced Settings' page:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Choose **System**. Select **General Configuration** to open the Administration page.  
   ![Keyboard shortcut: g + g + start typing general configuration](image)
3. Click the **Advanced Settings** button on the 'General Configuration' page to display this page:
4. Edit the value of a **Key/Property** by clicking its value on the right of the page and modifying the existing value.

Refer to the following **Topics** for more information about the **Keys** (above):

<table>
<thead>
<tr>
<th>Key</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira.attachments.number.of.zip.entries</td>
<td>Configuring the number of files shown in the content of ZIP-format files on issues</td>
</tr>
<tr>
<td>jira.clone.prefix</td>
<td>Configuring the cloned issue summary field prefix</td>
</tr>
<tr>
<td>jira.date-picker.java.format</td>
<td>Configuring date picker formats</td>
</tr>
<tr>
<td>jira.date-picker.javascript.format</td>
<td>Configuring date picker formats</td>
</tr>
<tr>
<td>jira.date.time-picker.java.format</td>
<td>Configuring date picker formats</td>
</tr>
<tr>
<td>jira.date.time-picker.javascript.format</td>
<td>Configuring date picker formats</td>
</tr>
<tr>
<td>jira.issue.actions.order</td>
<td>Changing the Default Order for Comments from Ascending to Descending</td>
</tr>
<tr>
<td>jira.projectkey.pattern</td>
<td>Changing the Project Key Format</td>
</tr>
<tr>
<td>jira.table.cols.subtasks</td>
<td>Configuring sub-task fields displayed on parent issues</td>
</tr>
<tr>
<td>jira.view.issue.links.sort.order</td>
<td>Configuring the order of linked issues displayed on the ‘view issue’ page</td>
</tr>
</tbody>
</table>

5. Click the **Update** button (which will appear in the **Operations** column on the right) to save the new value in the JIRA database.

**Please Note:**
Any changes you make to these properties/keys become effective immediately.
Click the General Settings button to return to the General Configuration page.

Related information

There are a handful of other advanced configuration options (which are of little interest to most JIRA system administrators) whose default values can be customized 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

<table>
<thead>
<tr>
<th>To Configure System Properties in Linux Installations,</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. From <code>&lt;jira-install&gt;/bin</code> (or <code>&lt;Tomcat-home&gt;/bin</code> for JIRA WAR installations), open <code>setenv.sh</code>.</td>
</tr>
<tr>
<td>2. Find the section <code>JVM_SUPPORT_RECOMMENDED_ARGS=</code></td>
</tr>
<tr>
<td>3. Refer to the list of parameters below.</td>
</tr>
</tbody>
</table>

Add all parameters in a space-separated list, inside the quotations.

Windows (starting from .bat file)

<table>
<thead>
<tr>
<th>To Configure System Properties in Windows Installations When Starting from the .bat File,</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. From <code>&lt;jira-install&gt;/bin</code> (or <code>&lt;Tomcat-home&gt;/bin</code> for JIRA WAR installations), open <code>setenv.bat</code>.</td>
</tr>
<tr>
<td>2. Find the section <code>set JVM_SUPPORT_RECOMMENDED_ARGS=</code></td>
</tr>
<tr>
<td>3. Refer to the list of parameters below.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Setting Properties for Windows Services via Command Line</th>
</tr>
</thead>
</table>
1. Identify the name of the service that JIRA is installed as in Windows (Control Panel > Administrative Tools > Services):

   ![JIRA Service Properties](image)

   In the above example, the **SERVICENAME** is: JIRA231112155942

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 you are running JIRA WAR).
4. Run:

   ```
   tomcat7w //ES//%SERVICENAME%
   ```

   In the above example, it would be **tomcat7w //ES//JIRA231112155942**
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`
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
-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

<table>
<thead>
<tr>
<th>Memory Property</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Xmx</td>
<td>These properties are pre-existing. See related pages for instructions.</td>
<td>Increasing JIRA Memory</td>
</tr>
<tr>
<td>-Xms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XX:MaxPermSize</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-XX:+PrintGCTimeStamps</td>
<td>Set these for Garbage Collection tuning.</td>
<td>Using Garbage Collection Logs to Analyze JIRA Performance</td>
</tr>
<tr>
<td>-XX:+HeapDumpOnOutOfMemoryError</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-agentlib:yjpagent=onexit=memory ,dir=/path/to/write/snapshots</td>
<td></td>
<td>Profiling Memory and CPU Usage with YourKit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mail Property</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Datlassian.mail.senddisabled</td>
<td>Set to 'true' to disable mail. In Linux setenv.sh, there is a pre-existing flag to uncomment.</td>
<td>Migrating JIRA to Another Server</td>
</tr>
<tr>
<td>-Datlassian.mail.fetchdisabled</td>
<td></td>
<td>Notifications Are Issued for Incorrect Issues</td>
</tr>
<tr>
<td>-Datlassian.mail.popdisabled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Dmail.debug</td>
<td>If set to &quot;true&quot;, logs statements related to mail</td>
<td>Configuring JIRA’s SMTP Mail Server to Send</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Notifications Creating Issues and Comments from</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Email</td>
</tr>
<tr>
<td>-Dmail.mime.decodetext.strict</td>
<td></td>
<td>Unable to Decode Mail Subject or Body when Creating Issue From Email</td>
</tr>
<tr>
<td>-Dmail.imap.auth.plain.disable</td>
<td>Authenticate Failed Error when Connecting to Exchange</td>
<td></td>
</tr>
<tr>
<td>-Dmail.imaps.auth.plain.disable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Dmail.imap.starttls.enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Dmail.mime.decodeparameters</td>
<td>Sets mail handler to work correctly with emails from RFC 2231-compliant mail clients.</td>
<td>Installing JIRA WAR-EAR JIRA 4.0 Upgrade Guide</td>
</tr>
<tr>
<td>-Dmail.smtp.localhost</td>
<td></td>
<td>Problems Sending Email from JIRA - EHLO requires domain address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Encoding Property</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
</table>

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
<table>
<thead>
<tr>
<th>Other Properties</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Dfile.encoding</td>
<td>Set to utf-8 for encoding consistency</td>
<td>Integrating JIRA with CVS and ViewCVS Characters Not Supported by ASCII are Being Displayed as Question Marks Internationalisation and Encoding Troubleshooting SQL Exception while Updating Issues or Importing Data in JIRA with MySQL Due to Encoding International Characters in Notification Email Subject Lines Are Being Replaced with Question Mark</td>
</tr>
<tr>
<td>-Duser.timezone</td>
<td>Incorrect Times Displayed in JIRA</td>
<td></td>
</tr>
<tr>
<td>-Dsvnkit.http.methods</td>
<td>Values include Basic, Digest, Negotiate, NTLM</td>
<td>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</td>
</tr>
<tr>
<td>-Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER</td>
<td>false</td>
<td>OutOfMemory Due to Tomcat Memory Leak JRA-10145</td>
</tr>
<tr>
<td>-ea/-da</td>
<td>Enable/Disable assertions</td>
<td>java.lang.AssertionError When Sending Mail Via SMTP</td>
</tr>
<tr>
<td>-Djava.net.preferIPv4Stack</td>
<td></td>
<td>SocketException to Announce 'Invalid argument' for an Available Port</td>
</tr>
<tr>
<td>-Djavax.net.ssl.trustStore</td>
<td></td>
<td>Connecting to SSL services Unable to Send Email 'javax.net.ssl.SSLException' Due to SMTP Server via SSL</td>
</tr>
<tr>
<td>-Djava.awt.headless</td>
<td>Ships with true by default. Allows thumbnail generation.</td>
<td></td>
</tr>
<tr>
<td>-Dhttp.proxyHost</td>
<td>Outbound Proxy Server hostname and port</td>
<td>How to Configure an Outbound HTTP Proxy for JIRA</td>
</tr>
<tr>
<td>-Dhttp.proxyPort</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Memory Property</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Xmx</td>
<td>These properties are pre-existing. See related pages for instructions.</td>
<td>Increasing JIRA Memory</td>
</tr>
<tr>
<td>-Xms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XX:MaxPermSize</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mail Property</td>
<td>Notes</td>
<td>Related Pages</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
<td>---------------</td>
</tr>
<tr>
<td>-Datlassian.mail.senddisabled</td>
<td>Set to 'true' to disable mail. In Linux setenv.sh, there is a pre-existing flag to uncomment.</td>
<td>Migrating JIRA to Another Server Notifications Are Issued for Incorrect Issues</td>
</tr>
<tr>
<td>-Datlassian.mail.fetchdisabled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Datlassian.mail.popdisabled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Dmail.debug</td>
<td>If set to &quot;true&quot;, logs statements related to mail</td>
<td>Configuring JIRA's SMTP Mail Server to Send Notifications Creating Issues and Comments from Email</td>
</tr>
<tr>
<td>-Dmail.mime.decodetext.strict</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Dmail.imap.auth.plain.disable</td>
<td>Authenticate Failed Error when Connecting to Exchange</td>
<td></td>
</tr>
<tr>
<td>-Dmail.imaps.auth.plain.disable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-Dmail.imap.starttls.enable</td>
<td>'javax.mail.MessagingException No login methods supported' Due to IMAP over SSL</td>
<td></td>
</tr>
<tr>
<td>-Dmail.mime.decodeparameters</td>
<td>Sets mail handler to work correctly with emails from RFC 2231-compliant mail clients.</td>
<td>Installing JIRA WAR-EAR JIRA 4.0 Upgrade Guide</td>
</tr>
<tr>
<td>-Dmail.smtp.localhost</td>
<td>Problems Sending Email from JIRA - EHLO requires domain address</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Encoding Property</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Dfile.encoding</td>
<td>Set to utf-8 for encoding consistency</td>
<td>Integrating JIRA with CVS and ViewCVS Characters Not Supported by ASCII are Being Displayed as Question Marks Internationalisation and Encoding Troubleshooting SQL Exception while Updating Issues or Importing Data in JIRA with MySQL Due to Encoding International Characters in Notification Email Subject Lines Are Being Replaced with Question Mark</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other Properties</th>
<th>Notes</th>
<th>Related Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Duser.timezone</td>
<td>Incorrect Times Displayed in JIRA</td>
<td></td>
</tr>
</tbody>
</table>
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. To access the values for editing:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose System > General Configuration to open the Administration page. See Configuring Advanced Settings for details.

Once any of these properties' values are changed, they become effective immediately.

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.
Example contents to demonstrate format

```
jira.projectkey.warning = testwarning
jira.projectkey.description = testdescription
```

In new JIRA installations, this file may not initially exist and if so, needs to be created manually. For more information about editing the `jira-config.properties` file see here: How to edit the `jira-config.properties` file

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 customized 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 customize 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 behavior differently to when they are specified in your `jira-config.properties` file with no value.

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 customized. 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.

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 customized 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.

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.

`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:


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
```

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**

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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.

File attachments are enabled by default. If you wish, you can configure the way JIRA handles attachments, or disable this feature altogether.

⚠ **Attachments are not stored in JIRA’s database and so will need to be backed up separately.**

**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).

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. Choose **‡ > System**. Select **Advanced > Attachments** 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 steps in the section below.

**Configuring create/delete attachment permissions**

1. Choose 📖 > **Issues**. Select **Permission Schemes** to open the Permission Schemes page, which displays a list of all permission schemes in your JIRA system and the projects that use each scheme.

Keyboard shortcut: **g + g +** start typing **permission schemes**
2. For each relevant permission scheme:
   a. Click the Permissions link associated with the relevant permission scheme to edit that scheme's permissions.

   ![Permission Scheme Table]

   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.

   ![Add New Permission]

   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 the number of files shown in the content of ZIP-format files 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 Issue Cloning

JIRA's issue cloning behavior can be modified by JIRA system administrators.

Configuring cloned issue linking behavior

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 behavior 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.

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. Choose system > system > issue linking to open the issue linking page. 

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. Choose system > system > issue linking to open the issue linking page. 

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 (running Confluence version 4.0 or later). 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.
   - 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 order of linked issues displayed on the 'view issue' page

JIRA system administrators can define the order in which linked issues are displayed in the Issue Links section on the 'view issue' page. This is done by editing the value of the jira.view.issue.links.sort.order property on JIRA's Advanced Settings page.

Specify the fields by which to sort issues in the Issue Links section on the 'view issue' page by entering the appropriate 'value' for each field in a comma-separated list. This property behaves similarly to a list of values specified after the ORDER BY keyword in JIRA Query Language (JQL), whereby sorting is conducted by the first and then subsequent fields specified in the list.

The jira.view.issue.links.sort.order property can accept the following individual field values: 'key', 'type', 'status', 'priority' and 'resolution'.

Configuring the Whitelist

JIRA administrators can choose to allow incoming and outgoing connections and content from specified sources by adding URLs to the whitelist.

JIRA will display an error if content has been added that is not from an allowed source, and prompt the user to add the URL to the whitelist.

Application Links are automatically added to the whitelist. You do need to manually add them.
Add allowed URLs to the whitelist

To add a URL to the whitelist:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Choose ![open system administration](icon) > **System**. Select **Security > Whitelist** to open the Whitelist page. **Keyboard shortcut: g + g + type wh**

3. On the **Whitelist** page, enter the URL or expression you want to allow.
4. Choose the **Type** of expression (see **Expression Types** below for examples).
5. Choose **Allow Incoming** if you need to allow CORS requests (see below).
6. Choose **Add**.

Your URL or expression appears in the whitelist.

To test that your whitelisted URL is working as expected you can enter a URL in the **Test a URL** field. Icons will indicate whether incoming and / or outgoing traffic is allowed for that URL.

### Expression Types

When adding a URL to the whitelist, you can choose from a number of expression types.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain name</td>
<td>Allows all URLs from the specified domain.</td>
<td><a href="http://www.example.com">http://www.example.com</a></td>
</tr>
<tr>
<td>Exact match</td>
<td>Allows only the specified URL.</td>
<td><a href="http://www.example.com/thispage">http://www.example.com/thispage</a></td>
</tr>
<tr>
<td>Wildcard Expression</td>
<td>Allows all matching URLs. Use the wildcard * character to replace one or more characters.</td>
<td>http://*example.com</td>
</tr>
<tr>
<td>Regular Expression</td>
<td>Allows all URLs matching the regular expression.</td>
<td>http(s)?://www\example.com</td>
</tr>
</tbody>
</table>

### Allow Incoming

**Allow Incoming** enables CORS requests from the specified origin. The URL must match the format `scheme://host[:port]`, with no trailing slashes (`:port` is optional). So `http://example.com/` would not allow CORS requests from the domain example.com.

### Disabling the whitelist

The whitelist is enabled by default. You can choose to disable the whitelist however this will allow all URLs, including malicious content, and is not recommended.

To disable the whitelist:

1. Log in as a user with the **JIRA System Administrators** global permission.
2. Choose ![open system administration](icon) > **System**. Select **Security > Whitelist** to open the Whitelist page.
Keyboard shortcut: g + g + type wh
3. On the Whitelist page, click the Turn off whitelist button.
4. Choose Confirm.

All URLs will now be allowed. Unless your instance is running in an environment without internet access, we do not recommend disabling the whitelist.

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).

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. Choose Issues. Select Issue Types > Sub-Tasks to open the Sub-Tasks page.
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-Tasks administration screen.

To enable sub-tasks:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Issues. Select Issue Types > Sub-Tasks to open the Sub-Tasks page.
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 at least one sub-task issue type must be defined in JIRA for users to be able to create sub-tasks.

Sub-task issue types can be customized on the Sub-Tasks administration page (described above). The Sub-Ta
sk s administration page also allows you to create, edit (i.e. the name, description or icon) and translate your Sub-Task issue types.

Creating a sub-task issue type

To create a new sub-task issue type:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🔄 > Issues. Select Issue Types > Sub-Tasks to open the Sub-Tasks page.
   💡 Keyboard shortcut: g + g + start typing sub-tasks
3. Click Add New Sub-Task Issue Type button to open the Add New Sub-Task Issue Type dialog box.
4. Complete the following:
   - Name — enter a short phrase that best describes your new sub-task issue type.
   - Description — enter a sentence or two to describe when this sub-task issue type should be used.
   - Icon URL — supply the path of a image that has been placed somewhere inside <jira-applica
   tion-dir>/images/icons of your JIRA Installation Directory or from an accessible URL.

Editing a sub-task issue type

To edit a sub-task issue type:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🔄 > Issues. Select Issue Types > Sub-Tasks to open the Sub-Tasks page.
   💡 Keyboard shortcut: g + g + start typing sub-tasks
3. Click the Edit link (in the Operations column) for the sub-task issue type that you wish to edit.
4. Edit the Name, Description and/or Icon as described above for Creating a sub-task issue type.

Deleting a sub-task issue type

You can only delete sub-task issue types through the Manage Issue Types page. For details, please see Deleting an Issue Type.

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 displayed on parent issues

JIRA system administrators can define which fields of sub-tasks are displayed in the Sub-Tasks section on the 'view issue' page of a parent issue (which contains one or more sub-tasks). This is done by editing the value of the jira.table.cols.subtasks property on JIRA's Advanced Settings page.

Specify which fields you want to show in the Sub-Tasks section of a parent issue's 'view issue' page by entering the appropriate 'value' for each field in a comma-separated list. The jira.table.cols.subtasks property can accept the values indicated in right-hand column of the IssueFieldConstants table on the Constant Field Values page (of JIRA's API documentation).

ℹ️ Please Note:

- The order of each value in this list determines the order of their representative fields in the Sub-Tasks section of a parent issue's 'view issue' page.
Managing Shared Filters

A filter is a saved issue search. JIRA users can create and manage their own filters (see Using 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.

On this page:
- Changing the Ownership of a Shared Filter
- Deleting a Shared Filter

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. Choose **User Management**. Select **Shared Filters** to open the **Search Shared Filters** page.
   
   ![Search Shared Filters](image)

   *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 'favorite'.

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. Choose > User Management. Select 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 'favorite'.

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 favorite 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.

RELATED TOPICS

- Saving Searches ('Issue Filters')
- Receiving Search Results via Email

Managing Shared Dashboards

A dashboard is a customizable 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.

On this page:

- Changing the Ownership of a Shared Dashboard
- Deleting a Shared Dashboard

Changing the Ownership of a Shared Dashboard

Before changing the ownership of a shared dashboard, ensure that you inform the shared dashboard's current
To change the ownership of a shared dashboard:

1. Log in as a user with the **JIRA Administrators** global permission.
2. Choose **Administration > User Management**. Select **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 'favorite'.
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 Customizing 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. Choose **Administration > User Management**. Select **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 ‘favorite’.
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 favorite is specified in this message box.
5. If you are happy to proceed, click the 'Delete' button to complete the action.

**RELATED TOPICS**
- Managing Multiple Dashboard Pages
- Customizing the Dashboard

**Linking to Another Application**

Application Links (sometimes called "AppLinks") is a bundled plugin that allows you to link Atlassian applications to each other. Linking two applications allows you to share information and access one application's functions and resources from within the other.

Linking JIRA to other applications allows you to include information from these systems in JIRA projects and issues. For example, if you link JIRA to Confluence, you can include pointers to wiki pages when creating or editing issues. Another common use case is to link FishEye with JIRA; this allows you to view source code and changesets that correspond to your stories in JIRA. In addition to Atlassian applications, you can also link to external applications; for example, you might use a plugin that allows you to share ZenDesk or Salesforce data via an application link.

1. Log in to JIRA as a user with the JIRA Administrators permissions.
2. Choose > Add-ons. Select Application Links in the Add-ons panel to open the configuration screen.
3. In the Application URL box, supply the URL of the application you want to link to and then select Create new link.
4. Use the wizard to finish configuring the link. If the application you are linking to does not have the Application Links plugin, you must supply additional information so the link can be set up. This information is required to set up a link with OAuth authentication; see Configuring OAuth for more information about that type of authentication.

When you complete the wizard, the Application Links plugin will create the link between your applications using the most secure authentication method that is supported between the two application types. After the link has been set up, it will appear on the "Configure Application Links" page. You can use this page to change the configuration of application links to make them more secure or to change the link settings:
   - To edit the settings of the application link (for example, to change the authentication type of the link), select Configure.
   - If you've set up multiple links to the same type of application (for example, multiple JIRA servers), you can use the Make Primary link to specify which application is the default instance. See Making a Primary
Link for Links to the Same Application Type for more information.

- After you've linked applications, you also connect the areas of those applications that contain information relating to your project or team (for example, you can connect a project's Confluence space with a JIRA project). These types of links are called **project links**.

### Server Administration

- Increasing JIRA Memory
- Using the Database Integrity Checker
- Precompiling JSP pages
- Logging and Profiling
- Restoring Data
- Optimizing Performance
- Backing Up Data
- Search Indexing
- Using robots.txt to hide from Search Engines
- Updating your JIRA License Details
- Viewing your System Information
- Monitoring Database Connection Usage
- Viewing JIRA's Instrumentation Statistics
- Generating a Thread Dump
- Finding the JIRA Support Entitlement Number (SEN)
- Performance Testing Scripts

### 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

**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**

To determine the JIRA usage patterns:

Choose ![icon] > System. Select Troubleshooting and Support > System Info to open the System Info page. Then 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:

  ![Windows Task Manager](image)

  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**

1. Identify the name of the service that JIRA is installed as in Windows (Control Panel > Administrative Tools > Services):

   ![Service Configuration](image)

   In the above example, the **SERVICENAME** is: JIRA120312230938

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 you are running the JIRA WAR distribution).
   For Example:

   ```
   cd C:\Program Files\Atlassian\JIRA\bin
   ```
4. For JIRA 5.1 or below:

   tomcat6w //ES//%SERVICENAME%

For JIRA 5.2 or above:

   tomcat7w //ES//%SERVICENAME%

In the above example, it would be tomcat6w //ES//JIRA120312230938

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:
-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.util.logging.config.file=/usr/local/jira/conf/logging.properties
-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.

**Using the Integrity Checker**

1. Log in as a user with the 'JIRA System Administrators' global permission.
2. Choose 🔄 > System. Select 'Troubleshooting and Support' > Integrity Checker 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.

1. **Select one or more integrity checks** from the list below to check for out of date information in the database.

   - **Check Issue Relations**
     - Check issue for Relation "ParentProject"
     - Check issue for Relation "RelatedOSWorkflowEntry"
     - Check that all Issue Links are associated with valid issues

   - **Check Search Request**
     - Check search request references a valid project

   - **Check for Duplicate Permissions**
     - Check the permissions are not duplicated

   - **Check Workflow Integrity**
     - Check workflow entry states are correct
     - Check workflow current step entries
     - Check JIRA issues with null status

   - **Check Field Layout Scheme Integrity**
     - Check field layout schemes for references to deleted custom fields

   - **Check for invalid filter subscriptions**
     - Check FilterSubscriptions for references to non-existent QuartzTriggers
     - Check FilterSubscriptions for references to non-existent SearchRequests
     - Check for existence of SimpleTriggers

2. **Check**

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:jaspc>` task, as shown:
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

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Logging levels

There are five logging levels available in log4j: 'DEBUG', 'INFO', 'WARN', 'ERROR' and 'FATAL'. Each logging level provides more logging information than 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 log4j<category>.com.atlassian categories.

Temporarily changing the logging level

1. Log in as a user with the 'JIRA System Administrators' global permission.
2. Choose Troubleshooting and Support > Logging & Profiling to open the Logging page, which lists all defined log4j categories (as package names) and their current logging levels.
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) Redbuild 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:
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. Choose **System** > **Troubleshooting and Support > Logging & Profiling** to open the Logging page, 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:

```xml
<filter>
    <filter-name>profiling</filter-name>
</filter>

<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 -->
    <param-name>activate.param</param-name>
    <param-value>jira_profile</param-value>
</init-param>

<init-param>
    <!-- specify the whether to start the filter automatically -->
    <param-name>autostart</param-name>
    <param-value>false</param-value>
</init-param>
```

3. Modify the `autostart` parameter to be **true** instead of **false**. That is:
3. <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.

To do this

Set `-Dmail.debug=true` and restart JIRA. Refer Setting Properties and Options on Startup for details on how to do this.

Output

In the logs, you should then see JavaMail initialize the first time a mail operation is run:

```java
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.SMTPSSLTransport,Sun Microsystems, Inc],
     com.sun.mail.smtp.SMTPTransport=javax.mail.Provider[TRANSPORT,smtp,com.s
        un.mail.smtp.SMTPTransport,Sun Microsystems, Inc],
     com.sun.mail.imap.IMAPSSLStore=javax.mail.Provider[STORE,imaps,com.sun.m
        ail.imap.IMAPSSLSStore,Sun Microsystems, Inc],
     com.sun.mail.pop3.POP3SSLSStore=javax.mail.Provider[STORE,pop3s,com.sun.m
        ail.pop3.POP3SSLSStore,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.IMAPSSLSStore,Su
        n Microsystems, Inc],
     imap=javax.mail.Provider[STORE,imap,com.sun.mail.imap.IMAPStore,Sun Mi
        crosystems, Inc],
     smtps=javax.mail.Provider[TRANSPORT,smtps,com.sun.mail.smtp.SMTPSSLS
        port,Sun Microsystems, Inc],
     pop3=javax.mail.Provider[STORE,pop3,com.sun.mail.pop3.POP3Store,Sun Mi
        crosystems, Inc],
     pop3s=javax.mail.Provider[STORE,pop3s,com.sun.mail.pop3.POP3SSLSStore,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-80
   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-Dispo
Related pages

- Logging and Profiling (JIRA 6.1) — 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.

- User access logging (JIRA 6.1) — How to track user actions with page access logging

- Where are the application server logs? (JIRA 6.1) — A decision tree for finding JIRA logs on your system

- Logging email protocol details (JIRA 6.1) — How to enable email protocol (SMTP, IMAP, POP) logging in JIRA

Restoring Data

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.

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:
- Before you begin
- 1. Disabling email sending/receiving
- 2. Restoring XML data
- 3. Restoring attachments

Before you begin

- 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.

- If you are restoring data from a **JIRA OnDemand site to an installed JIRA server**, please read Migrating from JIRA OnDemand to a JIRA Installed Site.

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.n.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 DISABLE_NOTIFICATIONS environment variable before starting JIRA (note, use startup.sh instead of startup.bat if you are not using Windows):
You could also try un-commenting the `DISABLE_NOTIFICATIONS=" -Datlassian.mail.senddisabled=true -Datlassian.mail.fetchdisabled=true -Datlassian.mail.popdisabled=true"` line from your `/bin/setenv.bat` file (`/bin/setenv.sh` if you are not using Windows) and then running startup.

2. Restoring XML data

<Warning> These instructions assume that you are restoring JIRA data from an XML backup. If you used native database tools to back up your JIRA data, the restore process will be tool-specific and so these instructions (i.e. stage 2 of 3) do not apply to you. </Warning>

1. Log in as a user with the 'JIRA System Administrators' global permission.


   <Keyboard shortcut>‘g’ + ‘g’ + type ‘rest’</Keyboard shortcut>

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.
   </Please note>

5. Click the 'Restore' button and wait while your JIRA data is restored.

   <Info> 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. </Info>

   <Info> 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. </Info>

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 must 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 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?

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 any custom field plugins (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 plugins 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.

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.

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.

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.

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.

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.
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 customized 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.

**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, log in as a user with the 'JIRA System Administrators' global permission.

1. Choose > System. Select Import & Export > Project Import to open the Project Import wizard page.
2. 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).
3. Copy the attachments from the path where you have backed up the attachments to the 'Backup Attachment Path' shown in the import window. This path is under the JIRA Home Directory of the instance. Please note that if file attachments are not enabled in your target JIRA instance you will not see the path to which you need to copy the attachments from the backup.

**Note:** You can choose to not copy the attachments to the '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

Project Import: Select Project to Import

The list of projects contains all projects present in the XML backup provided. Select the project you wish to import. The importer will attempt to automatically map the backup project's values to correct values in this instance of JIRA. Please make certain you have correctly configured the JIRA project in this instance (i.e. associated the correct schemes with the project, created any missing issue types, custom fields, etc.) See the documentation for full details of what needs to be done.

If a backup project cannot be imported the details will be displayed below:

Projects from Backup

- Demonstration Project

  - No project with key DEMO exists in this instance of JIRA. The importer will create a project with this key and the details of the backup project using the default schemes.

<table>
<thead>
<tr>
<th>Project</th>
<th>Demonstration Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key:</td>
<td>DEMO</td>
</tr>
<tr>
<td>Description:</td>
<td>Welcome to the administration of your demonstration project! This is where you can view and change how the project is configured. Use the tabs on the left to navigate to different project settings.</td>
</tr>
<tr>
<td>Lead:</td>
<td>admin</td>
</tr>
<tr>
<td>URL:</td>
<td></td>
</tr>
<tr>
<td>Default Assignee:</td>
<td>Project Lead</td>
</tr>
<tr>
<td>Issues:</td>
<td>6</td>
</tr>
<tr>
<td>Components:</td>
<td>0</td>
</tr>
<tr>
<td>Versions:</td>
<td>0</td>
</tr>
</tbody>
</table>

Next | Cancel

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.

Optimizing Performance

For more information about optimising performance in JIRA, please refer to our Crashes and Performance Issues Troubleshooting knowledge base article.

Back 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

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 while JIRA is active.

Caveat: if you are migrating your instance or performing an upgrade, please create an XML backup per the directions in this guide. Native tools run a much higher risk of failure and/or corruption if used for migration or disaster-recovery purposes.

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. Choose 🚀 > System. Select Import & Export > Backup System to open the Backup JIRA data page. Keyboard shortcut: 'g' + 'g' + type 'backup'

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...
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 +` start typing services

3. In the Add Service form towards the end 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) or 1440 minutes (24 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 is displayed.
5. Complete the following items on this page:
   - 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.
   - If the **Use Default Directory** check box is displayed, see the note below.

6. 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'.

   **Note:** JIRA will create the 'corrupted' directory if required - you do not need to create it.

### About custom backup directories

The **Use Default Directory** check box (not shown in screenshot above) is for legacy JIRA installations (prior to JIRA 4.2), which have backup services that use custom directories.

If you are using JIRA 5.1.0 or earlier, the **Use Default Directory** will always be displayed, as the option of using custom directories has been deprecated. If you are using JIRA 5.1.1 or later, the **Use Default Directory** check box will only be displayed if you upgraded from a version of JIRA prior to 4.2 and you are editing an existing backup service which used a custom directory.

- If you are not using a legacy backup service with a custom directory, select the the **Use Default Directory** check box. If you do not, your backup service may not work correctly.
- If you are using a legacy backup service with a custom directory, you can choose between using the default directory or your custom directory (cannot be edited). Note, if you choose the default directory option, you will not be able to 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.

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.

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.

See Re-Indexing after Major Configuration Changes for more information on when you should re-index.

Re-indexing JIRA

1. Log in as a user with the 'JIRA Administrators' global permission.
2. Choose System > Advanced > Indexing to open the Re-Indexing page.
   
3. This page allows you to choose one of the following two re-indexing options:
   • Background re-index — This will re-index all issues in the background.
   • Lock JIRA and rebuild index — This will delete and rebuild all indices, including the comment and change history indices.

Screenshot: Re-indexing JIRA
Which re-indexing option should I use?

The Background re-index option should be used in the majority of circumstances, particularly following changes to the configuration. It will generally take significantly longer to perform than the Lock JIRA and rebuild index option, but it allows JIRA to remain usable while it is being done. There will however be a performance impact on JIRA as a whole. We recommend that you perform this option during a low usage period. The actual impact of running the Background re-index option will depend upon the customer's particular hardware and software installation as well as how many issues are in the system.

The Lock JIRA and rebuild index should be used when:

- the indices are corrupt, which may be caused by a system or disk failure or
- it is more important to have the re-index completed quickly than to have JIRA continuously available. The Lock JIRA and rebuild index option may be in the order of twice as fast as a background re-index.

The following table summarises the differences between the two options:

<table>
<thead>
<tr>
<th>Background re-index</th>
<th>Lock JIRA and rebuild index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slower to complete.</td>
<td>Faster to complete (may be up to twice as fast).</td>
</tr>
<tr>
<td>JIRA can be used by users during re-index.</td>
<td>JIRA cannot be used by users during re-index.</td>
</tr>
<tr>
<td>Can be cancelled at any time.</td>
<td>Cannot be cancelled once started.</td>
</tr>
</tbody>
</table>

NFS is not supported. See this page for details: Supported Platforms.

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).

#### Editing robots.txt

JIRA (version 3.7 and later) installs the following robots.txt file at the root of the JIRA webapp:

```bash
# 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](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. Choose System > License to view your existing JIRA license details.
   ✅ 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.

**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. Choose System > System. Select Troubleshooting and Support > System Info 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
Warnings

Any warnings about known issues with your configuration will be displayed here.

System Info

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base URL</td>
<td>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.</td>
</tr>
<tr>
<td>System Date</td>
<td>The JIRA server's system date.</td>
</tr>
<tr>
<td>System Time</td>
<td>The JIRA server's system time.</td>
</tr>
<tr>
<td>Current Working Directory</td>
<td>For a description of the JIRA Working Directory, please see Important Directories and Files.</td>
</tr>
<tr>
<td>Java Version</td>
<td>The JIRA server's Java version.</td>
</tr>
<tr>
<td>Java Vendor</td>
<td>The JIRA server's Java vendor.</td>
</tr>
<tr>
<td>JVM Version</td>
<td>The JIRA server's JVM version.</td>
</tr>
<tr>
<td>JVM Vendor</td>
<td>The JIRA server's JVM version.</td>
</tr>
<tr>
<td>JVM Implementation Version</td>
<td>The JIRA server's JVM implementation version.</td>
</tr>
<tr>
<td>Java Runtime</td>
<td>The JIRA server's Java runtime environment.</td>
</tr>
<tr>
<td>Java VM</td>
<td>The JIRA server's Java Virtual Machine.</td>
</tr>
<tr>
<td>User Name</td>
<td>The operating system login name which JIRA runs under.</td>
</tr>
<tr>
<td>User Timezone</td>
<td>The JIRA server's timezone.</td>
</tr>
<tr>
<td>User Locale</td>
<td>The JIRA server's locale. Unless the default language is modified in JIRA's General Configuration, the User Locale will dictate the default language.</td>
</tr>
<tr>
<td>System Encoding</td>
<td>The JIRA server's system encoding.</td>
</tr>
<tr>
<td>Operating System</td>
<td>The JIRA server's operating system.</td>
</tr>
<tr>
<td>OS Architecture</td>
<td>The JIRA server's operating system architecture (e.g. i386).</td>
</tr>
<tr>
<td>Application Server Container</td>
<td>The application server in which your JIRA instance is running (see Supported Platforms for a list of supported application servers).</td>
</tr>
<tr>
<td>Database type</td>
<td>The type of database to which your JIRA instance is connected (see Supported Platforms for a list of supported databases).</td>
</tr>
<tr>
<td><strong>Database JNDI address</strong></td>
<td>The JNDI address of the database to which your JIRA instance is connected. (For more details, see Connecting JIRA to a Database.)</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Database URL</strong></td>
<td>The URL of the database to which your JIRA instance is connected. (For more details, see Connecting JIRA to a Database.)</td>
</tr>
<tr>
<td><strong>Database version</strong></td>
<td>The version of the database to which your JIRA instance is connected (see Supported Platforms for a list of supported database versions).</td>
</tr>
<tr>
<td><strong>Database driver</strong></td>
<td>The driver which your JIRA instance is using to connect to its database. (For more details, see Connecting JIRA to a Database.)</td>
</tr>
<tr>
<td><strong>External user management</strong></td>
<td>'ON' / 'OFF' indicates whether JIRA’s users are being managed externally or internally to JIRA (e.g. via Crowd).</td>
</tr>
<tr>
<td><strong>Crowd integration</strong></td>
<td>'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.</td>
</tr>
<tr>
<td><strong>JVM Input Arguments</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Modified Files</strong></td>
<td>A list of any files in your JIRA installation that have been modified as part installation or customisation of JIRA.</td>
</tr>
<tr>
<td><strong>Removed Files</strong></td>
<td>A list of any files that have been removed from your JIRA installation.</td>
</tr>
</tbody>
</table>

**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.

<table>
<thead>
<tr>
<th><strong>Setting</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total Memory</strong></td>
<td>The total amount of memory allocated to the JVM that is available to this instance of JIRA. For more details, see Increasing JIRA Memory.</td>
</tr>
<tr>
<td><strong>Free Memory</strong></td>
<td>The amount of free JVM memory currently available to this instance of JIRA.</td>
</tr>
<tr>
<td><strong>Used Memory</strong></td>
<td>The amount of JVM memory currently being used by this instance of JIRA.</td>
</tr>
<tr>
<td><strong>Total PermGen Memory</strong></td>
<td>The total amount of PermGen (Permanent Generation) memory available to this instance of JIRA.</td>
</tr>
<tr>
<td><strong>Free PermGen Memory</strong></td>
<td>The amount of free PermGen (Permanent Generation) memory currently available to this instance of JIRA.</td>
</tr>
</tbody>
</table>
### JIRA Info

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uptime</td>
<td>The period of time since your JIRA instance was last started.</td>
</tr>
<tr>
<td>Edition</td>
<td>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'.)</td>
</tr>
<tr>
<td>Version</td>
<td>The version of JIRA you are running. (Note: for the latest version, please see JIRA Releases.)</td>
</tr>
<tr>
<td>Build Number</td>
<td>The build number of your JIRA version. This is generally only useful to Atlassian's support engineers.</td>
</tr>
<tr>
<td>Build Date</td>
<td>The date on which your JIRA version was built. This is generally only useful to Atlassian's support engineers.</td>
</tr>
<tr>
<td>Atlassian Partner</td>
<td>Indicates whether your distribution of JIRA was built by an Atlassian partner company. Blank indicates that it was built directly by Atlassian.</td>
</tr>
<tr>
<td>Installation Type</td>
<td>Indicates whether JIRA has been installed as a 'recommended' distribution or as a 'WAR' distribution.</td>
</tr>
<tr>
<td>Server ID</td>
<td>This number is calculated automatically by JIRA, based on your license number.</td>
</tr>
<tr>
<td>Last Upgrade</td>
<td>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.</td>
</tr>
<tr>
<td>Installed Languages</td>
<td>A list of all language packs available within the JIRA system. (Note: to install additional languages, see Translating JIRA.)</td>
</tr>
</tbody>
</table>
### Default Language

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Language</td>
<td>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.</td>
</tr>
</tbody>
</table>

### 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.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date Purchased</td>
<td>The date on which this system's JIRA license was originally purchased. Note: you can verify this information by visiting <a href="http://my.atlassian.com">http://my.atlassian.com</a></td>
</tr>
<tr>
<td>License Type</td>
<td>For information about the different types of JIRA licences, please see <a href="http://www.atlassian.com/software/jira/licensing.jsp">http://www.atlassian.com/software/jira/licensing.jsp</a></td>
</tr>
<tr>
<td>Maintenance Period End Date</td>
<td>For information about JIRA support and maintenance, please see <a href="http://www.atlassian.com/software/jira/licensing.jsp">http://www.atlassian.com/software/jira/licensing.jsp</a></td>
</tr>
<tr>
<td>Maintenance Status</td>
<td>For information about JIRA support and maintenance, please see <a href="http://www.atlassian.com/software/jira/licensing.jsp">http://www.atlassian.com/software/jira/licensing.jsp</a></td>
</tr>
<tr>
<td>Support Entitlement Number (SEN)</td>
<td>For information about JIRA support and maintenance, please see <a href="http://www.atlassian.com/software/jira/licensing.jsp">http://www.atlassian.com/software/jira/licensing.jsp</a></td>
</tr>
</tbody>
</table>

### Configuration Info

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attachments Enabled</td>
<td>'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.</td>
</tr>
<tr>
<td>Issue Voting Enabled</td>
<td>'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.</td>
</tr>
<tr>
<td>Issue Watching Enabled</td>
<td>'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.</td>
</tr>
</tbody>
</table>
### 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 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 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.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issues</td>
<td>The number of issues that have been created in this JIRA system.</td>
</tr>
<tr>
<td>Projects</td>
<td>The number of projects that have been created in this JIRA system.</td>
</tr>
<tr>
<td>Custom Fields</td>
<td>The number of custom fields that have been created in this JIRA system.</td>
</tr>
<tr>
<td>Workflows</td>
<td>The number of workflows that have been created in this JIRA system.</td>
</tr>
<tr>
<td>Users</td>
<td>The number of user IDs that have been created in this JIRA system.</td>
</tr>
<tr>
<td>Groups</td>
<td>The number of groups that have been created in this JIRA system.</td>
</tr>
</tbody>
</table>

---

### File Paths

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location of JIRA Home</td>
<td>The path to your JIRA Home Directory. For information about changing the location, see Setting your JIRA Home Directory.</td>
</tr>
<tr>
<td>Location of entityengine.xml</td>
<td>The path to your Entity Engine. For database-specific information about configuring your entityengine.xml file, see Connecting JIRA to a Database.</td>
</tr>
</tbody>
</table>
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.

Monitoring Database Connection Usage
JIRA provides a view of its database connection usage. This provides information on the activity of the connection pool, as well as the frequency of reads/writes to the database. You can use this information to tune your database connections for better performance.

The instructions on this page describe how to navigate to the database connection usage information in the JIRA administration console, and how to interpret the information. If you want to make changes to your database connection pool settings using this information, see this related topic: Tuning Database Connections.

On this page:
- Accessing the Database Monitoring Page
- Interpreting the database monitoring graphs

Related pages:
- Tuning Database Connections
- Enterprise Resources

Accessing the Database Monitoring Page
To access the Database Monitoring page:

1. Log in as a user with the **JIRA System Administrators** global permission.

2. Choose ➔ Add-ons. Select **Database Monitoring** to display the Database Monitoring page.

   | Keyboard shortcut: g + g + start typing database monitoring |

**Database Monitoring**

This page allows you to monitor JIRA’s database connection usage. You can use this information to tune the size of JIRA’s connection pool and to diagnose performance problems. See [Connecting JIRA to a Database](#) for more info.

### Connection Pool

The 'Connection Pool' graph shows the activity in the connection pool for the last 6 hours.

- This graph shows the number of active and idle connections, as well as the maximum and minimum for the period.
- The scale of the vertical axis is equal to the maximum number of connections.
- The readings are averages over a period of 5 minutes.

This information can help you to optimise database connection usage. For example, if the number of active connections is consistently or frequently near to the maximum available, then you may need to raise the

### Reads / Writes

The 'Reads / Writes' graph shows the cumulative reads and writes for the last 6 hours.

- The graph shows the number of reads and writes, as well as the maximum and minimum for the period.
- The readings are averages over a period of 5 minutes.

This information can help you to optimise database read and write performance. For example, if the number of reads or writes is consistently or frequently near to the maximum available, then you may need to increase the

---

Interpreting the database monitoring graphs

**Connection Pool graph**

The 'Connection Pool' graph shows the activity in the connection pool for the last 6 hours.

- This graph shows the number of active and idle connections, as well as the maximum and minimum for the period.
- The scale of the vertical axis is equal to the maximum number of connections.
- The readings are averages over a period of 5 minutes.

This information can help you to optimise database connection usage. For example, if the number of active connections is consistently or frequently near to the maximum available, then you may need to raise the
maximum connections available in the pool. Conversely, if the number of active connections is consistently low compared to the maximum available, then you may want to lower the maximum connections available in the pool. For more information on how to tune database connections, see Tuning Database Connections.

Reads / Writes graph

The ‘Reads / Writes’ graph shows the frequency of reads and writes to the database over a period of time. It can be helpful to correlate database usage with connection pool usage. Whenever JIRA needs to access (i.e. read from or write to) the database, a database connection is required. If there are regular spikes in the reads / writes, you may need to consider raising the maximum connections available in the pool.

Viewing JIRA's Instrumentation Statistics

JIRA provides an Instrumentation page, which displays a variety of statistics on a wide range of internal properties within JIRA that have been 'instrumented' (i.e. recorded) for presentation through JIRA's administration area.

This page is mostly useful to help Atlassian Support provide assistance with your support queries, especially if they ask you to quote the statistics of one or more properties listed on this page.

To access JIRA's 'Instrumentation' page:

1. Log in as a user with the JIRA System Administrators global permission.
2. Choose System. Select Troubleshooting and Support > Instrumentation to display the Instrumentation page.
   ✔ Keyboard shortcut: g + g + start typing instrumentation
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
<th>Invocation</th>
<th>Time (ms)</th>
<th>CPU (nanos)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cache.CachingFieldConfigContextPersister evictionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.CachingFieldConfigContextPersister hitCount</td>
<td>Counter</td>
<td>1,314</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.CachingFieldConfigContextPersister loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.CachingFieldConfigContextPersister loadSuccessCount</td>
<td>Counter</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.CachingFieldConfigContextPersister missCount</td>
<td>Counter</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.CachingFieldConfigContextPersister size</td>
<td>Gauge</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.CachingFieldConfigContextPersister totalLoadTime</td>
<td>Counter</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager evictionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager hitCount</td>
<td>Counter</td>
<td>14,533</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager loadSuccessCount</td>
<td>Counter</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager missCount</td>
<td>Counter</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager size</td>
<td>Gauge</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultFieldLayoutManager totalLoadTime</td>
<td>Counter</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager evictionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager hitCount</td>
<td>Counter</td>
<td>1,935</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager loadSuccessCount</td>
<td>Counter</td>
<td>602</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager missCount</td>
<td>Counter</td>
<td>602</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager size</td>
<td>Gauge</td>
<td>602</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultIssueLinkManager totalLoadTime</td>
<td>Counter</td>
<td>4,701</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager evictionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager hitCount</td>
<td>Counter</td>
<td>17,824</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager loadSuccessCount</td>
<td>Counter</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager missCount</td>
<td>Counter</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager size</td>
<td>Gauge</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultPermissionSchemeManager totalLoadTime</td>
<td>Counter</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager evictionCount</td>
<td>Counter</td>
<td>38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager hitCount</td>
<td>Counter</td>
<td>713,077</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager loadSuccessCount</td>
<td>Counter</td>
<td>43</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager missCount</td>
<td>Counter</td>
<td>43</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager size</td>
<td>Gauge</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.DefaultUserPropertyManager totalLoadTime</td>
<td>Counter</td>
<td>17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager evictionCount</td>
<td>Counter</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager hitCount</td>
<td>Counter</td>
<td>34,914</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager loadSuccessCount</td>
<td>Counter</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager missCount</td>
<td>Counter</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager size</td>
<td>Gauge</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.JiraOsgiContainerManager totalLoadTime</td>
<td>Counter</td>
<td>134</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache directives evictionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metric</td>
<td>Type</td>
<td>Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>---------------</td>
<td>----------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.directives.hitCount</td>
<td>Counter</td>
<td>419,353</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.directives.loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.directives.loadSuccessCount</td>
<td>Counter</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.directives.missCount</td>
<td>Counter</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.directives.size</td>
<td>Gauge</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.evictionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.hitCount</td>
<td>Counter</td>
<td>419,353</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.loadExceptionCount</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.loadSuccessCount</td>
<td>Counter</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.missCount</td>
<td>Counter</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.size</td>
<td>Gauge</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache.VelocityTemplateCache.totalLoadTime</td>
<td>Counter</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>concurrent users</td>
<td>Gauge</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>db.conns</td>
<td>Operation</td>
<td>112,352</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>db.conns.borrowed</td>
<td>Gauge</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>db.reads</td>
<td>Operation</td>
<td>103,234</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>db.writes</td>
<td>Operation</td>
<td>4,669</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dbcp.maxActive</td>
<td>Gauge</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dbcp.numActive</td>
<td>Gauge</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>entity.customfields.total</td>
<td>Gauge</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>entity.groups.total</td>
<td>Gauge</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>entity.issues.total</td>
<td>Gauge</td>
<td>301</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>entity.projects.total</td>
<td>Gauge</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>entity.users.total</td>
<td>Gauge</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>entity.workflows.total</td>
<td>Gauge</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>http.session.objects</td>
<td>Gauge</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>http.sessions</td>
<td>Gauge</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>index.writes</td>
<td>Operation</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>issue.index.reads</td>
<td>Operation</td>
<td>96</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.class.loaded.current</td>
<td>Gauge</td>
<td>26,029</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.class.loaded.total</td>
<td>Counter</td>
<td>27,594</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.class.unloaded.total</td>
<td>Counter</td>
<td>1,585</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.gc</td>
<td>Operation</td>
<td>194</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.memory.heap.committed</td>
<td>Gauge</td>
<td>659,767,296</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.memory.heap.used</td>
<td>Gauge</td>
<td>554,776,504</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.memory.nonheap.committed</td>
<td>Gauge</td>
<td>146,669,568</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.memory.nonheap.used</td>
<td>Gauge</td>
<td>148,505,104</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.system.up.time</td>
<td>Gauge</td>
<td>1,058,358,864</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.block.count</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.block.time</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.time</td>
<td>Counter</td>
<td>323,250,000,000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.user.time</td>
<td>Counter</td>
<td>259,125,000,000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.wait.count</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.wait.time</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>jmx.thread.cpu.wait.time</td>
<td>Counter</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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.

### 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 you get the error "Not enough storage is available to process this command", download the 'psexec' utility from here, then run the following command using it:
  ```
psexec -s jstack <pid> >> threaddumps.txt
  ```
- 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-standalone/conf/logging.properties
   -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/common/endorsed
   -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:

- Searching, Indexing, and Filters Troubleshooting
- Poor Performance Due to Limited Database Connection Pooling
- JIRA Deadlocks when Running Tomcat 6.0.24
- OutOfMemory or Poor Performance due to XML View of a Filter
- JIRA Performance Tuning
- JIRA Crashes Due to OutOfMemoryError Java heap space

**Finding the JIRA Support Entitlement Number (SEN)**

There are three ways to find your Support Entitlement Number (SEN).

See [Finding Your Support Entitlement Number](#) in the support space for more general information about how Atlassian Support uses this number.

**Method 1: Check in the JIRA administration interface**

Access the JIRA license page, as described on [Updating your JIRA License Details](#). The JIRA license page will show your Support Entitlement Number (SEN).

**Method 2: Check my.atlassian.com**

Your Support Entitlement Number is available from the licenses page after logging in to [http://my.atlassian.com](http://my.atlassian.com):

**Method 3: Check your Atlassian invoice**

Your Support Entitlement Number (SEN) also appears on the third page of your Atlassian Invoice.
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. Furthermore, the performance testing scripts which you can download from Atlassian's public Maven repository (via the link on this page) are no longer supported.

This page is provided for your information only and using it is done so at your own risk. Instead of using these scripts, we would recommend our JIRA Performance Testing with Grinder page.

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

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.

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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](https://jmeter.apache.org/).

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:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira.host</td>
<td>localhost</td>
<td>The hostname or address of the JIRA installation.</td>
</tr>
</tbody>
</table>
### User Control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>create.users.enable</td>
<td>true</td>
<td>Create users in the target JIRA installation. Use false if you already have the users created elsewhere.</td>
</tr>
<tr>
<td>browseissue.max</td>
<td>250</td>
<td>The number of users to be created for browsing the JIRA installation (aka &quot;browseissue&quot; users).</td>
</tr>
<tr>
<td>createissue.max</td>
<td>250</td>
<td>The number of users to be created for creating issues (aka &quot;createissue&quot; users).</td>
</tr>
<tr>
<td>editissue.max</td>
<td>250</td>
<td>The number of users to be created for editing issues (aka &quot;editissue&quot; users).</td>
</tr>
<tr>
<td>search.max</td>
<td>250</td>
<td>The number of users to be created for searching issues (aka &quot;search&quot; users).</td>
</tr>
<tr>
<td>useraction.max</td>
<td>250</td>
<td>The number of users to be created for browsing user information (aka &quot;useraction&quot; users).</td>
</tr>
<tr>
<td>browseissue.groupname</td>
<td>none</td>
<td>The group to which &quot;browseissue&quot; users will be placed. Use none for no group.</td>
</tr>
<tr>
<td>createissue.groupname</td>
<td>jira-developers</td>
<td>The group to which &quot;createissue&quot; users will be placed. Use none for no group.</td>
</tr>
<tr>
<td>editissue.groupname</td>
<td>jira-developers</td>
<td>The group to which &quot;editissue&quot; users will be placed. Use none for no group.</td>
</tr>
<tr>
<td>search.groupname</td>
<td>none</td>
<td>The group to which &quot;search&quot; users will be placed. Use none for no group.</td>
</tr>
<tr>
<td>useraction.groupname</td>
<td>jira-developers</td>
<td>The group to which &quot;useraction&quot; users will be placed. Use none for no group.</td>
</tr>
</tbody>
</table>
### Project Control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>create.projects.enable</td>
<td>true</td>
<td>Create projects. Use false if you want to use existing projects (in existing data).</td>
</tr>
<tr>
<td>project.max</td>
<td>20</td>
<td>The number of projects to create in the system.</td>
</tr>
</tbody>
</table>

### Issue Control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>create.issues.enable</td>
<td>true</td>
<td>Creates issues in the target JIRA installation. Use false if you do not want the test to create sample issues.</td>
</tr>
<tr>
<td>issue.max</td>
<td>3000</td>
<td>The number of issues to be created.</td>
</tr>
<tr>
<td>issue.comment.enable</td>
<td>true</td>
<td>Controls whether or not comments are added to issues.</td>
</tr>
<tr>
<td>issue.comment.max</td>
<td>10</td>
<td>If issue.comment.enable is true, then the number of actual comments created on an issue is chosen randomly between 0 and this value.</td>
</tr>
<tr>
<td>issue.close</td>
<td>true</td>
<td>Controls whether or not issues will be closed automatically after being created.</td>
</tr>
<tr>
<td>issue.close.percentage</td>
<td>60</td>
<td>If issue.close is enabled, then this value defines the percentage of issues closed.</td>
</tr>
<tr>
<td>issue.setupload.threads</td>
<td>10</td>
<td>The number of threads used for creating the issues.</td>
</tr>
<tr>
<td>issue.setupload.pause</td>
<td>50</td>
<td>The amount of time (in milliseconds) for which a simulated user will 'sleep' between each request during issue creation.</td>
</tr>
<tr>
<td>resource.dir</td>
<td>resources</td>
<td>The directory which contains the CSV data resources.</td>
</tr>
</tbody>
</table>

### Test Output

Once you have chosen your target settings, run JMeter and you should get output similar to the following:
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 behavior if the test. By default, the test has the following behavior 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**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira.host</td>
<td>localhost</td>
<td>The hostname or address of the JIRA installation.</td>
</tr>
<tr>
<td>jira.port</td>
<td>8000</td>
<td>The network port that the JIRA installation is running on.</td>
</tr>
<tr>
<td>jira.context</td>
<td>/</td>
<td>JIRA webapp context.</td>
</tr>
<tr>
<td>admin.user</td>
<td>admin</td>
<td>Administrator username.</td>
</tr>
<tr>
<td>admin.pass</td>
<td>admin</td>
<td>Administrator password.</td>
</tr>
<tr>
<td>script.base</td>
<td>.</td>
<td>The location of the performance tests. This should only be set if you run</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the tests from outside the <code>scripts</code> directory.</td>
</tr>
<tr>
<td>script.runtime</td>
<td>1800</td>
<td>The amount of time to run for (in seconds).</td>
</tr>
<tr>
<td>resource.dir</td>
<td>resources</td>
<td>The subdirectory which contains the resource CSV files. Replace this if you</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wish to customize the backend data.</td>
</tr>
</tbody>
</table>

**Edit Issue**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>editissue.threads</td>
<td>5</td>
<td>The number of simultaneous ‘Edit Issue’ users to simulate.</td>
</tr>
<tr>
<td>editissue.pause</td>
<td>15000</td>
<td>The pause between each ‘Edit Issue’ user request (in milliseconds).</td>
</tr>
<tr>
<td>workflow.matchname</td>
<td>(Close</td>
<td>A regular expression to match the workflow to action.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>workflow.matchname</td>
<td>Resolve</td>
<td></td>
</tr>
<tr>
<td>editworkflow.percentage</td>
<td>20</td>
<td>The percentage of ‘Edit Issue’ user requests that will attempt to change the issue workflow.</td>
</tr>
<tr>
<td>addcomment.percentage</td>
<td>60</td>
<td>The percentage of ‘Edit Issue’ user requests that will attempt to add a comment to an issue.</td>
</tr>
<tr>
<td>editissue.percentage</td>
<td>20</td>
<td>The percentage of ‘Edit Issue’ user requests that will attempt to edit an issue.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Default</td>
<td>Explanation</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>editissue.issuestoown</td>
<td>5</td>
<td>The number of issues the test attempts to assign to an 'Edit Issue' user.</td>
</tr>
</tbody>
</table>

**User Actions**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>useraction.threads</td>
<td>1</td>
<td>The number of simultaneous 'User Action' users to simulate.</td>
</tr>
<tr>
<td>useraction.pause</td>
<td>40000</td>
<td>The pause between each 'User Action' user request (in milliseconds).</td>
</tr>
<tr>
<td>createfilter.percentage</td>
<td>10</td>
<td>The percentage of 'User Action' user requests that will attempt to create a filter.</td>
</tr>
<tr>
<td>viewwatches.percentage</td>
<td>10</td>
<td>The percentage of 'User Action' user requests that will attempt to 'view watches'.</td>
</tr>
<tr>
<td>viewvotes.percentage</td>
<td>10</td>
<td>The percentage of 'User Action' user requests that will attempt to view votes.</td>
</tr>
</tbody>
</table>

**Browse Issues and Projects**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>browseissue.threads</td>
<td>40</td>
<td>The number of simultaneous 'Browse Issue' users to simulate.</td>
</tr>
<tr>
<td>browseissue.pause</td>
<td>3000</td>
<td>The pause between each 'Browse Issue' user request (in milliseconds).</td>
</tr>
<tr>
<td>userprofile.percentage</td>
<td>10</td>
<td>The percentage of 'Browse Issue' user requests that will attempt to browse a user profile.</td>
</tr>
<tr>
<td>browsedashboard.percentage</td>
<td>20</td>
<td>The percentage of 'Browse Issue' user requests that will attempt to browse the dashboard.</td>
</tr>
<tr>
<td>dashboard.category</td>
<td>allprojects</td>
<td>The project category for project status gadget requests.</td>
</tr>
</tbody>
</table>

**RSS**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>browserss.threads</td>
<td>10</td>
<td>The number of simultaneous 'RSS' users to simulate.</td>
</tr>
<tr>
<td>browserss.pause</td>
<td>60000</td>
<td>The pause between each 'RSS' user request (in milliseconds).</td>
</tr>
</tbody>
</table>

**Create Issues**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
</table>
### Search For Issues

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>search.threads</td>
<td>2</td>
<td>The number of simultaneous 'Search' users to simulate.</td>
</tr>
<tr>
<td>search.pause</td>
<td>15000</td>
<td>The pause between each 'Search' user request (in milliseconds).</td>
</tr>
</tbody>
</table>

**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 in 2013 by Atlassian. Licensed under a
Creative Commons Attribution 2.5 Australia License

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:
  0 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 (125671229128)
... 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
JIRA 6.1 Documentation

created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.

-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.

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. This page provides an overview of the mechanisms available for extending JIRA.

1 JIRA Add-ons: For information on installing or enabling existing add-ons, please read the Managing JIRA Add-ons documentation. To learn about creating your own add-ons, see developing add-ons with the Atlassian Plugin SDK.

Note that an add-on that specifically plugs into the architecture of an Atlassian application such as JIRA is sometimes called a plugin, although the terms ‘plugin’ and ‘add-on’ are often used interchangeably.

<table>
<thead>
<tr>
<th>Custom Field Types</th>
<th>JIRA comes with various custom field types defined. New types can be written and plugged into JIRA. See the How to create a new Custom Field Type tutorial for more information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Formats</td>
<td>JIRA comes with many options to change the look and feel of features in the system. User formats are a feature that can be customized by add-ons. You can write your own User Format add-on to change the display of user details in JIRA, e.g. display a profile picture. See the User Format Plugin Module for more information.</td>
</tr>
<tr>
<td>Gadgets</td>
<td>New gadgets can be created by writing an XML descriptor file, packaged as an Atlassian plugin. See Writing an Atlassian Gadget for more information.</td>
</tr>
<tr>
<td>Reports</td>
<td>JIRA comes with various reports built-in. Using the plugin system, new reports can be written, providing new ways of viewing and summarizing JIRA’s data.</td>
</tr>
</tbody>
</table>
| 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 provides hooks where you can plug in your own behavior:  
  - 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 How to create Custom Workflow Elements for JIRA guide for details on how to write your own workflow post-functions, conditions and validators. Once written, these can be packaged as plugins and reused. |
### Issues and Projects

One the 'View Issue' page, some issue information (comments, change history) is displayed. Likewise, the 'Browse Project' page contains separate sections, listed on the far left, for different types of project information:

**Demonstration**

**Overview**

**Summary**

Welcome to the administration of your demonstration project!

This is where you can view and change how the project is configured. Use the tabs on the left to navigate to different project settings.

**Issues: 30 Day Summary**

By writing a plugin, you can add new issue or project sections (that will be listed in the left panel) 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 Services overview for general information. For building RPC clients, check out the creating a JIRA SOAP Client and Creating an XML-RPC Client tutorials. New RPC endpoints can also be added to JIRA as plugins - see the RPC Endpoint Plugin Module.

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 Add-ons

About Add-ons

An add-on is an installable component that supplements or enhances the functionality of JIRA in some way. For example, the JIRA Calendar Plugin is an add-on that shows the due dates for issues and versions in calendar format. Other add-ons are available for connecting JIRA to Bamboo, developing for JIRA, and accessing Atlassian support from JIRA.

JIRA comes with many pre-installed add-ons (called system add-ons). You can install more add-ons, either by acquiring an add-on from the Atlassian Marketplace or by uploading an add-on from your file system. This means that you can install add-ons that you have developed yourself. For information about developing your own add-ons for JIRA, see the JIRA Developer documentation.

To enable various JIRA Gadgets (for example, the Text Gadget), please refer to this section: Configuring the Default Dashboard.

On this page:
- About Add-ons
- About the Universal Plugin Manager

You may notice that the terms 'add-on' and 'plugin' both appear in the Atlassian documentation and tools. While the terms are often used interchangeably, there is a difference. A plugin is a type of add-on that can be installed into an Atlassian host application. Plugins are what developers create with the Atlassian SDK. But there are other types of add-ons as well. For example, the JIRA client is an add-on that runs as a separate program rather than as a plugin to JIRA. This documentation uses the term 'add-on' most often.

About the Universal Plugin Manager

The Universal Plugin Manager (UPM) is itself an add-on that you use to administer add-ons from the JIRA Administration Console. UPM works across Atlassian applications, providing a consistent interface for administering add-ons in JIRA, Confluence, Crucible, Fisheye, Stash or Bamboo.

UPM comes pre-installed in recent versions of all Atlassian applications, so you do not normally need to install it yourself. However, like other add-ons, the UPM software is subject to regular software updates. Before administering add-ons in JIRA, therefore, you should verify your version of the UPM and update it if needed.

You can update UPM, or any add-on, from the UPM's own add-on administration pages. In addition to updating UPM, you can perform these tasks from the administration pages:
• Install or remove add-ons
• Configure add-on settings
• Discover and install new add-ons from the Atlassian Marketplace
• Enable or disable add-ons and their component modules, including "safe mode"

If the add-on request feature is enabled in your Atlassian application, non-administrative users can also discover add-ons on the Atlassian Marketplace. Instead of installing the add-ons, however, these users have the option of requesting the add-ons from you, the administrator of the Atlassian application.

For more information on administering the add-on request feature or performing other common add-on administration tasks, see the Universal Plugin Manager documentation. For an end-user's view of requesting add-ons in JIRA, see Requesting Add-ons.

Using the Issue Collector

What is an 'issue collector'?

The issue collector allows you to easily embed a JIRA feedback form into your own web site. This form is typically accessed by clicking a 'trigger' tab exposed along the edge of pages in your web site.

When used by people visiting your web site click this trigger tab and submit the resulting JIRA feedback form, an issue is conveniently created in JIRA.

Visitors to your web site do not require a user account in JIRA to use the JIRA feedback form.

Accessing JIRA’s issue collectors

In JIRA, issue collectors are configured (and hence organised) on a per-project basis.

To access all issue collectors configured in JIRA:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🚀 > Add-ons. Select Issue Collectors to open the Issue Collectors page, which shows a list of all existing issue collectors in your JIRA system.
3. Click the name of a project to access a more detailed list of issue collectors belonging to that project or click the name of an issue collector to access detailed information about it. On the issue collector page (containing detailed information), you can access:
   • An activity graph, showing the number of issues created via this issue collector (Y-axis) on a daily basis (X-axis).
   • A list of recent issues in reverse chronological order, which have been created via this issue collector.

On this page:
• What is an 'issue collector'?
• Accessing JIRA's issue collectors
• Adding an issue collector
• Embedding an issue collector into your web site
• Editing an issue collector
• Copying an issue collector
• Disabling or deleting an issue collector
• Known Limitations

Related pages:
• Advanced Use of the JIRA Issue Collector

To access issue collectors belonging to a specific project:
1. Log in to JIRA as a project administrator or a user with the JIRA Administrators global permission. A project administrator is someone who has the Administer Project project-specific permission, but not necessarily the JIRA Administrators global permission.

2. Choose 🔄 > Projects.
   ✔️ Keyboard shortcut: g + g + start typing project

3. On the left of the Project Summary page, click the Issue Collectors tab. The Issue Collectors page is displayed, listing any issue collectors that have already been set up in your project.

   ![Issue Collectors](image)

4. Click the name of an issue collector to access detailed information about it — in particular, its recent activity and details on how to embed the issue collector into your web site.

Adding an issue collector

1. Log in to JIRA as a project administrator or a user with the JIRA Administrators global permission.

2. Choose 🔄 > Projects.
   ✔️ Keyboard shortcut: g + g + start typing project

3. On the left of the Project Summary page, click the Issue Collectors tab. The Issue Collectors page is displayed, listing any issue collectors that have already been set up in your project.

4. Click the Add Issue Collector button to open the Add Issue Collector page.

5. In the top section of the Add Issue Collector page, specify the following:

<table>
<thead>
<tr>
<th>Name</th>
<th>Specify the name of the issue collector, as you want it to appear throughout the JIRA user interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Specify a description for the issue collector. This description will appear adjacent to the Name of your issue collector, throughout the JIRA user interface.</td>
</tr>
<tr>
<td>Issue Type</td>
<td>Select the type of issue that you want created in JIRA when visitors to your web site submit your issue collector's JIRA feedback form.</td>
</tr>
<tr>
<td>Issue Reporter</td>
<td>Specify the username that will be the default reporter of JIRA issues created when visitors to your web site submit your issue collector's JIRA feedback form.</td>
</tr>
</tbody>
</table>

---

*Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](https://creativecommons.org/licenses/by/2.5/au/).*
Match Reporter?

Select either of the following:

- **Always use Issue Reporter** — select this option to ensure that the default Issue Reporter you specify above, will always be the reporter of issues created by submission of the JIRA feedback form on your web site.

- **Attempt to match user session of submitter or submitter email address** — select this option if you want the reporter of an issue created by submission of the JIRA feedback form on your web site, to be a JIRA user:
  - Who is logged in to JIRA when they submit a JIRA feedback form on your web site (in the same browser session).
  - Who's email address matches the email address specified in the E-mail field of the JIRA feedback form.

Please note that if the JIRA user does not have the Create Issues project permission in your JIRA project, the default Issue Reporter you specify above will be used as the issue's reporter.

Collect Browser Info

Select this option to collect meta-information about your browser's statistics, which will be incorporated into issues created by submission of the JIRA feedback form on your web site.

6. In the middle section of the Add Issue Collector page (entitled Trigger), specify the following:

<table>
<thead>
<tr>
<th>Trigger Text</th>
<th>Specify a short, brief phrase that will appear on the trigger tab on your web site.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger Style</td>
<td>Choose the style in which trigger tab will appear on your web site. Custom will not display a trigger but will add additional javascript to the generated script, so you can create a custom trigger on your web page.</td>
</tr>
</tbody>
</table>

7. In the lower section of the Add Issue Collector page (entitled Issue Collector Form), specify the following:
Choose from the options provided. Typically, your choice would reflect the type of issue being created (i.e. chosen above). You can choose:

- A predefined template for your JIRA feedback form — either **Got Feedback?** or **Raise a Bug**.
- **Custom** to create a custom JIRA feedback form, which allows you to specify your own wording on the dialog box as well as add or remove other fields on the form and change their positions on the form.
- Please note that if a field is **required** for the chosen issue type but that field has:
  - No specified a default value, the field will automatically appear on the form. This field’s position can be changed on the form, although it cannot be removed.
  - A default value but the field is not added to the form, then the field’s default value is used when an issue is created via the issue collector.
- **Not all fields of types of fields can be added to the form, since some fields cannot be displayed to anonymous users.** The fields types that can be displayed are:
  - **Standard Fields**: Summary, Description, Components, AffectsVersion, Environment, Priority, Attachment
  - **Custom Field Types**: Date Time, Radio Buttons, Multi-Checkbox, Multi-Select, Number, Select List, URL field, Version Picker, Cascading Select, Project Picker, Single Version Picker, Text Field, Free Text Field

8. Click the **Submit** button to save your changes.

**Embedding an issue collector into your web site**

After clicking the **Submit** button (above) to save your new issue collector, a page containing code snippets is displayed. Use the code and information provided to embed your new issue collector into your web site.

If you accidentally click away from this page, you can easily retrieve the information that was on it by accessing your issue collector’s details (above) and scrolling to the end of the page.
Editing an issue collector

Editing an issue collector should not require any changes to web pages that include the issue collector, unless you change the Trigger Style to or from a custom trigger. Changing the Trigger Style to or from a custom trigger will change the generated javascript, so you may need to change what you embed in any web page that includes the issue collector.

To edit an existing issue collector:

1. Log in to JIRA as a project administrator or a user with the JIRA Administrators global permission.
2. Access the relevant project's list of issue collectors (above).
3. In the Operation dropdown for the issue collector you would like to edit, select Edit to open the Edit Issue Collector page.
4. Update the issue collector, as desired.
5. Click Update to save your changes.

Copying an issue collector

Copying an issue collector will create an entirely new issue collector and will not affect any existing issue collectors. You will need to embed it in whatever web pages you would like, just as if you had created a new issue collector.

To copy an existing issue collector:

1. Log in to JIRA as a project administrator or a user with the JIRA Administrators global permission.
2. Access the relevant project's list of issue collectors (above).
3. In the Operation dropdown for the issue collector you would like to edit, select Copy to open the Add Issue Collector page.
4. All the information from the copied issue collector will be the same as the copied issue collector with the exception of the name (which will be "Copy of " + the original name of the copied issue collector.
5. Update the issue collector, if desired.
6. Click Submit to save your changes.

Disabling or deleting an issue collector

To disable or delete an issue collector:

1. Access the relevant project's list of issue collectors (above).
2. On the list of the project's issue collectors, click Disable or Delete to respectively disable or delete the associated issue collector.

   While an issue collector is disabled, its trigger tabs will still be visible on pages of your web site(s) to which the issue collector code has been added until a user refreshes the page. However, clicking these triggers results in a message indicating that the issue collector is currently out of action.

Known Limitations

Placing the Issue Collector plugin within a frameset will not close the prompt window automatically.

This is a known limitation for Issue Collector plugin and has been tracked at

[JIRA-29886 - Issue Collector Cannot Be Closed When Placed Inside a Frameset (Resolved)]

Advanced Use of the JIRA Issue Collector

Customizing the JIRA issue collector

The JIRA issue collector can be used without any additional JavaScript beyond the single line generated in the issue collector administration screens in JIRA. However, you can also customize the JIRA issue collector in a number of different ways:

- Set up a custom trigger, so the feedback form launches from a different link or button than the packaged triggers provided.
- Set the default values of fields for your users, using JavaScript.
- Specify the values of fields on the issue, which are not shown in the feedback form.

This page assumes you are already familiar with Using the Issue Collector.
Warning: The JavaScript exposed by the issue collector is not considered a stable API and may change with new JIRA releases.

On this page:
- Customizing the JIRA issue collector
  - Setting up a custom trigger
- Adding the custom trigger function manually
- Setting field values from JavaScript
- Embedding multiple issue collectors
- Embedding the Issue Collector
  - Full Source Code
  - Is localization of an issue collector possible?

Related pages:
- Using the Issue Collector

Setting up a custom trigger

Configuring your collector to use a custom trigger

If you want to use a different trigger, or button, to launch the issue collector on your website, configure your issue collector as described below:

1. Add a new issue collector, or edit an existing issue collector.
2. Scroll down to section Trigger and select the option 'Custom'.
3. You don't need to set any Trigger Text as this will be overridden by your custom trigger.

Screenshot: Using a custom trigger for an issue collector

Adding the issue collector script for a custom trigger

The issue collector script generated by JIRA for adding a custom trigger is slightly different to the script generated for the standard triggers, because it includes the JavaScript function for the custom trigger.

Customization of the issue collector is done by creating/extending the global object ATL_JQ_PAGE_PROPS. This allows you to add a custom trigger, set default values for fields and more.

Note: In JIRA 5.1 (and version 1.1 of the Issue Collector plugin), the issue collector administrative interface let you define the custom trigger function UI, and you did not need to include it in the JavaScript on the page. In version 1.2 of the Issue Collector, the custom trigger JavaScript is a part of the generated JavaScript that you should copy and paste into your web page.

The code snippet below shows a sample HTML page with the generated issue collector JavaScript.

In the example below, we've added a simple button in HTML, and made that button launch the issue collector. This is done simply by replacing 'myCustomTrigger' in the generated JavaScript with the HTML id of the button ('feedback-button')
<head>
  <!-- We pasted the generated code from the Issue Collector here, after choosing a custom trigger -->

  <!-- This is the script for the issue collector feedback form -->


  <!-- This is the script for specifying the custom trigger. We've replaced 'myCustomTrigger' with 'feedback-button' -->

  <script type="text/javascript">
    window.ATL_JQ_PAGE_PROPS = {
      "triggerFunction": function(showCollectorDialog) {
        // Requires that jQuery is available!
        jQuery("#feedback-button").click(function(e) {
          e.preventDefault();
          showCollectorDialog();
        });
    }
  }

</head>

<body>

<h2>JIRA Issue Collector Demo</h2>
<a href="#" id="feedback-button" class='btn btn-primary btn-large'>Report feedback</a>

</body>
Adding the custom trigger function manually

The custom trigger JavaScript will be included in the JavaScript generated by the Issue Collector. However, this section provides details on how you could do it without pasting in the additional lines of generated JavaScript.

To add a custom trigger, add the property `triggerFunction` in the global object `ATL_JQ_PAGE_PROPS`. `triggerFunction` needs to be defined as a function and takes one argument which is the function for displaying the issue collector.

You can invoke the issue collector from any element on your page by adding a click handler in `triggerFunction` as shown below. In this example, we will be calling the issue collector from our `#feedback-button` anchor tag defined in the above HTML markup. You can assign multiple triggers for the same issue collector by adding more click handlers.

```javascript
window.ATL_JQ_PAGE_PROPS = $.extend(window.ATL_JQ_PAGE_PROPS, {

    // ==== custom trigger function ====
    triggerFunction : function( showCollectorDialog ) {
        $('#feedback-button').on( 'click', function(e) {
            e.preventDefault();
            showCollectorDialog();
        });

        // add any other custom triggers for the issue collector here
    }
});
```

The `triggerFunction` will be invoked by the Issue Collector after the $(document).ready() phase.

Setting field values from JavaScript

Setting field values

The issue collector gives you the option to set field values for any of the fields on the issue type. This is done by adding the property `fieldValues` in the global object `ATL_JQ_PAGE_PROPS`. There are different methods for setting default values for different field types. The code samples below show a visual representation of a field in JIRA and its relevant markup, and how to set a default value for that field type. Use a DOM inspection tool such as Firebug in the JIRA Issue Create Screen to extract the field names and values relevant to your issue collector. Please note that the Issue Collector is not supposed to be a replacement for the JIRA REST API. If you require a more customized solution, make use of the JIRA REST API to create JIRA issues from external websites. The JIRA Travel App is a good example of how you can build a front end interface with JIRA as the back end.
Visible fields (setting default field values)

If you set the value of a field that is visible on the issue collector feedback form, the fields will already be filled in with that value when the form opens.

Hidden fields

There might be cases where you might want to set a field value without actually displaying the field on the Issue Collector. In this case, simply use the same method as above to set the field values via JavaScript. The fields will not be shown as they were not added in the form template but their values will still be present in issues created with the Issue Collector.

JavaScript for Setting field values

Setting field values is done by specifying field name / value pairs within the "fieldValues" block of window.ATL_JQ_PAGE_PROPS. If you already have a custom trigger defined, you can simply add to the definition of window.ATL_JQ_PAGE_PROPS like the example below.

Note the names of the fields are always the names of the field in the JIRA Create Issue Screen, not any overridden names you may have provided in the issue collector form.

```javascript
window.ATL_JQ_PAGE_PROPS = $.extend(window.ATL_JQ_PAGE_PROPS, {
    // ==== custom trigger function ====
    triggerFunction : function( showCollectorDialog ) {
        $('#feedback-button').on( 'click', function(e) {
            e.preventDefault();
            showCollectorDialog();
        });
    },
    // ==== we add the code below to set the field values ==== 
    fieldValues: {
        summary : 'Feedback for new website designs',
        description : 'The font doesn\'t quite look right',
        priority : '2'
    }
});
```

Examples of how to set specific field types

Text field example

Setting the value for a text field, like the issue Summary, is straightforward. Here's the markup for a text field like Summary in the issue Collector (you do not need to add this, this is simply to show the representation that the Issue Collector contains):

```html
<div class="field-group">
    ...
    <input class="text long-field" id="summary" name="summary" type="text" value="">
    ...
</div>
```

And here's how you set the value of the field in JavaScript:
Select list example with issue priority

Setting the value for a select list field, such as the issue priority, requires a little more effort, because you need to know the HTML element id for the choice you want to select. Here's the markup for the Priority field in the issue Collector (you do not need to add this, this is simply to show the representation that the Issue Collector contains):

```html
<div class="field-group">
...
<input id="priority-field" class="text aui-ss-field ajs-dirty-warning-exempt" autocomplete="off">
...
<select class="select" id="priority" name="priority" style="display: none; 
multiple="multiple">
<option class="imagebacked" data-icon="/images/icons/priority_blocker.gif" value="1">Blocker</option>
<option class="imagebacked" data-icon="/images/icons/priority_critical.gif" value="2">Critical</option>
...
</select>
...
</div>
```

And here's how you set the value of the field in JavaScript:

```json
fieldValues : {
  'priority' : '2'
}
```

Multi-select or checkboxes example

Setting the value for a multi-select (like the Browser field) or checkbox requires that you provide an array of values. Like the select list, you need to know the values to set, by looking at the markup on the Create Issue Screen.
Custom fields

Setting a value for a custom field is exactly the same as any other field in JIRA. Since multiple custom fields can share the same name, custom fields will be referenced by "customfield_" + the Id of the custom field in JIRA. This ID can be seen in the HTML markup for the Create Issue Screen in JIRA, but can also be determined by looking at the URLs on the custom fields screen in JIRA administration. Here's what the JavaScript would look like for setting a custom field whose ID in JIRA was 11111:

```javascript
fieldValues : { 'customfield_11111' : 'San Francisco' }
```

Cascading selects

Setting a value for a cascading select is done in two steps - one for the parent value and one for the child. Below is an example of setting the value of a cascading select field.

```javascript
fieldValues : { 'customfield_12345' : 'Australia', 'customfield_12345:1' : 'Sydney' }
```

Special Case Fields

Environment field

By default the Issue Collector puts user context such as the URL, User-Agent and screen resolution in the environment field. There might be cases where you wish to include more information in the environment field. In this case, you can add the property `environment` in the global object `ATL_JQ_PAGE_PROPS`. This allows you to add key value pairs that will appear on the environment field in the JIRA issue.
Restricted fields

Some fields that require a user to be logged into JIRA cannot be set through JavaScript. Assignee is an example of a field that cannot be set via JavaScript.

Dynamic functions

Environment and fieldValues properties can also be a function returning a JSON object that will be executed immediately when the collector trigger is shown (not just before opening the collector form). This might come in handy when you might wish to capture contextual information relevant to the user.
Embedding multiple issue collectors

If you want to have two different forms appear on the same web page, you will need to create two different issue collectors in JIRA. To set custom triggers, or set field values on those issue collectors requires a few changes to your page:

1. Include the generated JavaScript for both of your issue collectors in the page.
2. Find the id of each collector. This can be done one of two ways:
   a. The parameter of the script is "collectorId=<8 character id>. That's the ID you want.
   b. Go to the Issue Collector page in the Admin section and click on the Issue Collector you wish to embed. Copy the collectorId from the URL.

   https://<JIRA_URL>/secure/ViewCollector!default.jspa?projectKey=<PROJECT_KEY>&collectorId=<copy this part here>

Then, create separate namespaces for each of the issue collectors in the ATL_JQ_PAGE_PROPS object.

```javascript
window.ATL_JQ_PAGE_PROPS = $.extend(window.ATL_JQ_PAGE_PROPS, {
    '<collectorId_1>' : {
        triggerFunction: // define trigger function
        fieldValues: { // define field values here
        },
    },
    '<collectorId_2>' : {
        triggerFunction: // define trigger function
        fieldValues: { //define field values here
        }
    }
});
```

Embedding the Issue Collector

Embedding the Issue Collector in your Confluence Site

The Issue Collector can be embedded into Confluence using the HTML Include Macro. Note that using the HTML Include Macro would require you to embed the Issue Collector code separately on each page.

The Issue Collector was previously embeddable in Confluence via a User Macro, allowing you to create a re-usable Issue Collector macro that other Confluence users can embed into their pages. This option is currently unavailable due to a known bug:

- Some JavaScripts are not executed if included in User Macro.

Embedding the Issue Collector is not currently supported in Confluence OnDemand.

JIRA

The Issue Collector can be embedded in the Announcement Banner on a JIRA page by embedding the above script and HTML markup for your custom trigger in the Announcement Banner configuration screen. If you wish to change the location of your custom trigger, this can be easily done via jQuery. The following snippet shows how you can add the custom trigger onto the footer of all JIRA pages.

You cannot embed an Issue Collector in your JIRA OnDemand instance since HTML markup is disabled for the Announcement Banner.
Please note that embedding the Issue Collector requires you to enable HTML markup for the Announcement Banner.

**Full Source Code**

This source code shows how to embed two different issue collectors on the same page with custom triggers.

```html
<body>

<h2>JIRA Issue Collector Demo</h2>
<a href="#" id="feedback-button" class='btn btn-primary btn-large'>Report feedback</a>

<!-- JIRA Issue Collector - append this at the bottom of <body> -->

<!-- We will customize JIRA in the following script tag -->

<script type="text/javascript">
  // safely use jQuery here since the issue collector will load it for you
  $(document).ready(function() {

    window.ATL_JQ_PAGE_PROPS = $.extend(window.ATL_JQ_PAGE_PROPS, {

      // ==== feedback collector ====
      '<collectorId_1>': {

        // === custom trigger function ===
        triggerFunction : function( showCollectorDialog ) {
          $('#feedback_button').click( function(e) {
            e.preventDefault();
            showCollectorDialog();
          });
        },

        // === default and hidden field values ===
        fieldValues : {

          // default values
          summary : 'Feedback for new website designs'
        }
      }
    });
  });
</script>

```
, description: 'The font doesn\'t quite look right'

// hidden field value
, priority: '2'

// === custom trigger function ===

triggerFunction: function( showCollectorDialog ) {
  $('##bug_button').click( function(e) {
    e.preventDefault();
    showCollectorDialog();
  });
}

// === additional environment details ===

, environment: function() {

  var env_info = {};

  if ( window.ADDITIONAL_CUSTOM_CONTEXT ) {
    env_info[ 'Additional Context Information' ] = window.ADDITIONAL_CUSTOM_CONTEXT;
  }

  return env_info;
}

// === default field values ===

, fieldValues: function() {

  var values = {};

  var error_message = $('.error_message');
  if ( error_message.length !== 0 ) {

    // record error message from the page context rather than asking the user to enter it
    values[ 'summary' ] = error_message.children('.summary').text();
    values[ 'description' ] = error_message.children('.description').text();

  }

  return values;
}

});
Is localization of an issue collector possible?

You can create an issue collector 100% localized to the default language of your JIRA instance. Beyond that, complete localization of the issue collector is not possible.

The strings and text in the issue collector feedback form of the issue collector is a combination of:

1. The issue collector strings set by the JIRA Administrator
2. Either the default language setting for JIRA, or the language preference of the user if they are logged in to JIRA.
   - All users will see the names of the fields as they are set by the JIRA Administrator. These are not affected by the default language of JIRA, and are not affected by the default language of logged in JIRA users.
   - All users will see the field descriptions as they are set in the JIRA Administration UI.
   - For everything else:
     - Anonymous users will see everything else in the default JIRA language.
     - Logged in users will see everything else in the feedback form in the language specified by their JIRA profile.

Because of the above, you cannot create a single issue collector that will present itself entirely in the language of the end user.

However, if you want to create an issue collector that will present itself to anonymous users in the default language of your JIRA instance, you should:

1. Use the custom feedback template for the Issue Collector
2. Change the field labels in JIRA, and the labels for name and email, to the words you want to use in the default JIRA language.

The language setting of the browser will not impact the text in the feedback form.

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

Listener Interfaces

JIRA has the following concrete Listeners (which extend the base JiraListener interface):
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 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. Choose > System. Select Advanced > Listeners 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 which are passed as an array of String objects. The `init` method is given a `Map` with the configured values (the JavaDoc is outdated). The `com.atlassian.jira.event.listeners.DebugParamListener` class is a good example of doing this with two parameters.

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. Choose `🌐 > System`. Select Advanced > Services to open a page showing all the configured 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.TypeService`. See Sample services for provided service class names.
   - **Delay** — the delay (in minutes) between service runs.

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.

5. After completing the remaining options on the Edit Service page, click the Update button to save your new service’s options.

Editing service properties

To edit a service’s properties:

1. Log in as a user with the JIRA System Administrators global permission.

2. Choose `🌐 > System`. Select Advanced > Services to open a page showing all the configured 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. Choose `🌐 > System`. Select Advanced > Services to open a page showing all the configured services.
services.

- **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](https://my.atlassian.com)):

```
<source-installation-directory>/jira-project/jira-components/jira-core/src/main/java/com/atlassian/jira/service/services/export
```

#### 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.

> In Linux, the input-file and output-file properties are relative to your system's `/` folder. Please be careful to select a path that JIRA will have access to.

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](https://my.atlassian.com)):

```
<source-installation-directory>/jira-project/jira-components/jira-core/src/main/java/com/atlassian/jira/jelly/service
```

#### 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](https://jelly.apache.org) 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. Choose "System" > System. Select Advanced > Jelly Runner to open the Jelly Runner page. Keyboard shortcut: 'g' + 'g' + '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

Restricting Jelly

To remove the interface for pasting scripts in:
1. Edit atlassian-jira/secure/admin/views/jelly_runner.jsp
2. Add the disabled attribute to the textarea, e.g.
   ```html
   <ui:textarea label="text('admin.jellyrunner.jelly.script.xml')"
   name="script" rows="40" cols="80" disabled="true" />
   ```

   This prevents text being pasted into the Jelly Runner page. Note that this is only an interface change and it will be still possible to run Jelly scripts by submitting an HTTP request with the right content.

Writing a Jelly script

Jira Add Actors to Default Project Role
Jira Add Actors to Project Role
Jira Add Comment
Jira Add Component
Jira Add Field To Screen
Jira Add Permission
Jira Add User To Group
Jira Add Version
Jira Assign Issue
Jira Attach File
Jira Create Custom Field
Jira Create Group
Jira Create Issue

On this page:
- Enabling Jelly
- Running a Jelly script
- Restricting Jelly
- Writing a Jelly script
- Beta Tags
- Sample scripts
- `jira:CreatePermissionScheme`
- `jira:CreateProject`
- `jira:CreateProjectRole`
- `jira:CreateUser`
- `jira:DeleteProjectRole`
- `jira:GetDefaultRoleActors`
- `jira:GetProjectRole`
- `jira:GetProjectRoleActors`
- `jira:IsProjectRoleNameUnique`
- `jira:LinkIssue`
- `jira:Login`
- `jira:RemoveActorsFromDefaultProjectRole`
- `jira:RemoveActorsFromProjectRole`
- `jira:RemoveUser`
- `jira:RunSearchRequest`
- `jira:SelectComponentAssignees`
- `jira:TransitionWorkflow`
- `jira:UpdateProjectRole`

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.

<table>
<thead>
<tr>
<th>Special Character</th>
<th>Escaped equivalent</th>
</tr>
</thead>
</table>

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
The list of currently available tags:

**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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>This is the id of the project role.</td>
</tr>
<tr>
<td>actors</td>
<td>string</td>
<td></td>
<td>A comma delimited list of either users or groups</td>
</tr>
<tr>
<td>actortype</td>
<td>string</td>
<td></td>
<td>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.</td>
</tr>
</tbody>
</table>

**Examples**

**Adding a list of default users or groups to a project role**

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>This is the id of the project role.</td>
</tr>
<tr>
<td>actors</td>
<td>string</td>
<td></td>
<td>This a comma delimited list of either user names or group names</td>
</tr>
<tr>
<td>actortype</td>
<td>string</td>
<td></td>
<td>This defines the 'actor' type. Currently this field can contain either 'atlassian-user-role-actor' for users, or 'atlassian-group-role-actor' for groups.</td>
</tr>
<tr>
<td>projectkey</td>
<td>string</td>
<td></td>
<td>This is the key of the project you wish to add users or groups to for the specified role.</td>
</tr>
</tbody>
</table>

### Examples

**Adding a list of users or groups to a project role**

```xml
<jira:AddActorsToProjectRole projectroleid="1" actors="jira-administrators,jira-users" projectkey="MKY" actortype="atlassian-group-role-actor" />
```

**jira:AddComment**

This function adds a comment to an Issue.

### Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>issue-key</td>
<td>string</td>
<td></td>
<td>The issue to add the comment to (required).</td>
</tr>
<tr>
<td>commenter</td>
<td>string</td>
<td>Currently logged in user</td>
<td>Username of the user to make the comment (Must have browse and comment permissions).</td>
</tr>
<tr>
<td>comment</td>
<td>string</td>
<td></td>
<td>Comment to be added to the issue (required).</td>
</tr>
<tr>
<td>groupLevel</td>
<td>string</td>
<td>none</td>
<td>Name of group that can see this comment. NOTE: If this is specified you can not specify the roleLevel parameter.</td>
</tr>
</tbody>
</table>
### Examples

#### Create comment

```xml
<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

```xml
<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

<table>
<thead>
<tr>
<th>roleLevel</th>
<th>string</th>
<th>none</th>
<th>Name or Id of Project Role that can see this comment. NOTE: If this is specified you can not specify the groupLevel parameter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>created</td>
<td>string</td>
<td>Current Date/Time</td>
<td>Date/Time the Comment was created in format yyyy-MM-dd hh:mm:ss.0</td>
</tr>
<tr>
<td>updated</td>
<td>string</td>
<td>Current Date/Time</td>
<td>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.</td>
</tr>
<tr>
<td>editedBy</td>
<td>string</td>
<td>Currently logged in user</td>
<td>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.</td>
</tr>
<tr>
<td>tweakIssueUpdateDate</td>
<td>boolean</td>
<td>true</td>
<td>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.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Default Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>project-key</td>
<td>string</td>
<td></td>
<td>The key of the project you want to add the component to (not required if nested inside atag).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>Name of the component (required).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td></td>
<td>Description of the component.</td>
</tr>
<tr>
<td>componentLead</td>
<td>string</td>
<td></td>
<td>The username of the Component's lead. Leave blank for no lead.</td>
</tr>
</tbody>
</table>

Examples

Create Component

```xml
<jira:AddComponent project-key="ABC" name="Comp 1" description="Comp 1 description"/>
```

Create Component in a Project

```xml
<jira:CreateProject key="ABC" name="A Project" lead="logged-in-user">
  <jira:AddComponent name="Comp 1"/>
</jira:CreateProject>
```

Create Component with a Component Lead

```xml
<jira:AddComponent project-key="ABC" name="Comp 1" description="Comp 1 with lead" componentLead="user-name"/>
```

$jira:AddFieldToScreen$

Adds a field to a specific tab on a screen. Can also specify in which position to insert the field.

Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ffldId</td>
<td>string</td>
<td></td>
<td>Field ID of the field to add (required), e.g. &quot;description&quot;, &quot;duedate&quot;, etc.</td>
</tr>
<tr>
<td>screen</td>
<td>string</td>
<td></td>
<td>Screen ID or Name (required), e.g. &quot;1&quot; or &quot;Default Screen&quot;.</td>
</tr>
</tbody>
</table>
### Examples

#### Add Fields to a Screen

```xml
<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

```xml
<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 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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>schemeld</td>
<td>string</td>
<td>If not nested in a CreatePermissionScheme tag, specifies the scheme Id to add the permission to (0 is the default permission scheme).</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>permissions</td>
<td>required string</td>
<td>A comma-separated list of permissions to grant: String — Permission</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Project — Administer projects</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Browse — Browse projects</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Create — Create issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit — Edit issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ScheduleIssue — Schedule issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Move — Move issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Assign — Assign issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Assignable — Assignable user</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolve — Resolve issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Close — Close issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ModifyReporter — Modify reporter</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Comment — Add comments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommentEditAll — Edit all comments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommentEditOwn — Edit own comments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommentDeleteAll — Delete all comments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommentDeleteOwn — Delete own comments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Delete — Delete issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Work — Work on issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WorklogEditAll — Edit all worklogs</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WorklogEditOwn — Edit own worklogs</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WorklogDeleteOwn — Delete own worklogs</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WorklogDeleteAll — Delete all worklogs</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Link — Link issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Attach — Create attachments</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AttachDeleteAll — Delete all attachments</td>
<td></td>
</tr>
</tbody>
</table>
### 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](#))
Grant issue reporters the ability to edit/delete their own issues, in a new permission scheme

```xml
<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>
```

Granting a group selector custom field’s members the ability to assign/be assigned the issue.

```xml
<jiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:AddPermission schemeId="10164" type="groupCF" groupCF="customfield_10000" permissions="Assign,Assignable" />  
</jiraJelly>
```

Make projects using default permission scheme visible to certain users

```xml
<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>
```

```xml
<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>
```

### jira:AddUserToGroup

Makes a user a member of a Group. Adds the username and/or group name into the context if specified.

**Attributes**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>string</td>
<td></td>
<td>Username to add to Group (required if not in atag).</td>
</tr>
<tr>
<td>group-name</td>
<td>string</td>
<td></td>
<td>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.</td>
</tr>
</tbody>
</table>

Username is set in the context if specified in the tag. Group name is set in the context if specified in the tag.
Examples

Add User to Group

```xml
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:AddUserToGroup username="new-user" group-name="new-group"/>
</JiraJelly>
```

Add New User to Group

```xml
<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

```xml
<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

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>project-key</td>
<td>string</td>
<td></td>
<td>The key of the project you want to add the component too (not required if nested inside atag).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>Name of the version (required).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td></td>
<td>The description of the version.</td>
</tr>
<tr>
<td>releaseDate</td>
<td>string</td>
<td></td>
<td>The release date of the version.</td>
</tr>
</tbody>
</table>
schedule | string | Schedule of the version.

**Examples**

**Create a Version**

```xml
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:AddVersion project-key="ABC" name="Ver 1"/>
</JiraJelly>
```

**Create a Version in a Project**

```xml
<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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td></td>
<td>Key of the issue to assign.</td>
</tr>
<tr>
<td>assignee</td>
<td>string</td>
<td></td>
<td>User to assign issue to.</td>
</tr>
</tbody>
</table>

**Examples**

**Create and assign issue**

```xml
<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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td></td>
<td>Key of the issue to attach the file to. (Required)</td>
</tr>
<tr>
<td>filepath</td>
<td>string</td>
<td></td>
<td>Path (on the server) of the file to attach. (Required)</td>
</tr>
</tbody>
</table>
### Option

<table>
<thead>
<tr>
<th>Option</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>skip</td>
<td>string</td>
<td>do not attach file if a file with this name is already attached.</td>
</tr>
<tr>
<td>override</td>
<td>string</td>
<td>overwrite existing attached file</td>
</tr>
<tr>
<td>add</td>
<td>string</td>
<td>add the file as another attachment</td>
</tr>
</tbody>
</table>

### Created

<table>
<thead>
<tr>
<th>Created</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Date/Time</td>
<td>string</td>
<td>Date/Time the attachment was created, in format yyyy-MM-dd hh:mm:ss.0 (Optional)</td>
</tr>
</tbody>
</table>

### Examples

#### Adding an attachment

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldType</td>
<td>string</td>
<td></td>
<td>Field type as appears as the key in the plugin descriptor</td>
</tr>
<tr>
<td>fieldScope</td>
<td>string</td>
<td></td>
<td>One of global, project or issuuetype</td>
</tr>
<tr>
<td>fieldName</td>
<td>string</td>
<td></td>
<td>Name of custom field</td>
</tr>
<tr>
<td>projectKey</td>
<td>string</td>
<td></td>
<td>Key of the related project. Only valid for scope &quot;project&quot;</td>
</tr>
<tr>
<td>issueType</td>
<td>string</td>
<td></td>
<td>Issue type. Only valid for scope &quot;issuetype&quot;</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td></td>
<td>Description of the field to be displayed when adding a value</td>
</tr>
<tr>
<td>searcher</td>
<td>string</td>
<td>A valid related custom field searcher List of valid system searchers...</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>------------------------------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>
|          |        | • textsearcher
|          |        | • exacttextsearcher
|          |        | • daterange
|          |        | • datetimerange
|          |        | • exactnumber
|          |        | • numberrange
|          |        | • versionsearcher
|          |        | • projectsearcher
|          |        | • userpickersearcher
|          |        | • userpickergroupssearcher
|          |        | • grouppickersearcher
|          |        | • selectsearcher
|          |        | • radiosearcher
|          |        | • cascadingselectsearcher
|          |        | • multiselectsearcher
|          |        | • checkboxsearcher
|          |        | • labels

| customFieldIdVar | string | The name of the variable to place the new custom field. |

### Examples

**Create Cascading Custom Field**

The `jira:AddCustomFieldSelectValue` subtag can be used to add values for select lists. They can also be nested for Cascading Select Lists.

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group-name</td>
<td>string</td>
<td></td>
<td>Name of group to create (required).</td>
</tr>
</tbody>
</table>

### Context Variables

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jelly.group.name</td>
<td>string</td>
<td>Name of group being created.</td>
</tr>
</tbody>
</table>

### Examples

#### Create Group

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>project-key</td>
<td>string</td>
<td></td>
<td>Key of the project to add the issue to (required if not nested in atag).</td>
</tr>
<tr>
<td>issueType</td>
<td>string</td>
<td>First issue type</td>
<td>The string name of the Issue Type this issue should be created for (e.g. Bug).</td>
</tr>
<tr>
<td>summary</td>
<td>string</td>
<td></td>
<td>Summary of the issue being created (required).</td>
</tr>
<tr>
<td>priority</td>
<td>string</td>
<td>First priority</td>
<td>The string name of the Priority (e.g. Major).</td>
</tr>
<tr>
<td>components</td>
<td>string</td>
<td></td>
<td>The string name of the Component.</td>
</tr>
<tr>
<td>versions</td>
<td>string</td>
<td></td>
<td>The string name of the Affected Version.</td>
</tr>
<tr>
<td>fixVersions</td>
<td>string</td>
<td></td>
<td>The string name of the Fix For Version.</td>
</tr>
<tr>
<td>assignee</td>
<td>string</td>
<td></td>
<td>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 &quot;-1&quot; for Automatic assignment.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>reporter</td>
<td>string</td>
<td>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 JIRA-129 for further explanation. (Broken? See JIRA-5620.)</td>
<td></td>
</tr>
<tr>
<td>environment</td>
<td>string</td>
<td>Description of the environment.</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Detailed description of the issue.</td>
<td></td>
</tr>
<tr>
<td>duedate</td>
<td>string</td>
<td>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:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- jira.date_picker.java.format to value yyyy-MM-dd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- jira.date_picker.javascript.format at to value %Y-%m-%d</td>
<td></td>
</tr>
</tbody>
</table>

See Changing the Due Date Input Format for more information about changing these values.
<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>created</td>
<td>string</td>
<td>Current Date/Time Date/Time the Issue was created in format yyyy-MM-dd hh:mm:ss.0</td>
</tr>
<tr>
<td>updated</td>
<td>string</td>
<td>Current Date/Time Date/Time the Issue was updated in format yyyy-MM-dd hh:mm:ss.0</td>
</tr>
<tr>
<td>issueIdVar</td>
<td>string</td>
<td>The name of the variable to place the ID of the new Issue.</td>
</tr>
<tr>
<td>issueKeyVar</td>
<td>string</td>
<td>The name of the variable to place the Key of the new Issue.</td>
</tr>
<tr>
<td>duplicateSummary</td>
<td>string</td>
<td>Setting this attribute to 'ignore' will allow Issue with the same summary to be created.</td>
</tr>
<tr>
<td>security-level</td>
<td>string</td>
<td>Sets the security level of an issue. Value is the name of a level, e.g. 'Secret'.</td>
</tr>
</tbody>
</table>

**Examples**

**Create Issue**

```xml
<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.

```xml
<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`

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>long</td>
<td>ID of the custom field with the customfield_prefix</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>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)</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>Key is used for multi-dimensional data. Currently, only Cascading Selects supports its use. Omit to specify the value of parent, use &quot;1&quot; as the value for child.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td><strong>deprecated</strong> Name of the custom field.</td>
</tr>
</tbody>
</table>

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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
</table>
### Name

**name**  
**required string**  
Name of the permission scheme.

### Description

**description**  
**string**  
Permission scheme description.

### Context Variables

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jelly.permission.scheme.id</td>
<td>string</td>
<td>Id of the created permission scheme</td>
</tr>
</tbody>
</table>

### jira:CreateProject

This tag creates a new project in JIRA and places the project id in the context.

#### Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td></td>
<td>The project key used to create Issue Keys (required).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>The name of the project (required).</td>
</tr>
<tr>
<td>lead</td>
<td>string</td>
<td></td>
<td>The username of the user that is the project lead (required).</td>
</tr>
<tr>
<td>url</td>
<td>string</td>
<td></td>
<td>The URL of the site for this project.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td></td>
<td>The description of this project.</td>
</tr>
</tbody>
</table>

### Context Variables

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jelly.project.id</td>
<td>string</td>
<td>Id of the Project that was created.</td>
</tr>
<tr>
<td>jelly.project.key</td>
<td>string</td>
<td>Key of the Project that was created.</td>
</tr>
</tbody>
</table>

### Examples

#### Create Project

```xml
<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
<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>The name for the project role you will be creating</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td></td>
<td>The description for the project role you will be creating</td>
</tr>
</tbody>
</table>

### Context Variables

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jelly.role.id</td>
<td>Long</td>
<td>The id of the project role</td>
</tr>
<tr>
<td>jelly.role.name</td>
<td>string</td>
<td>The name of the project role</td>
</tr>
<tr>
<td>jelly.role.description</td>
<td>string</td>
<td>The description of the project role</td>
</tr>
</tbody>
</table>

### Examples

#### Creating a new project role

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>string</td>
<td></td>
<td>Username of the user being created (required).</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td></td>
<td>User's password. If the password field is left blank, a random password will be auto-generated.</td>
</tr>
<tr>
<td>confirm</td>
<td>string</td>
<td></td>
<td>Confirmation of users password (required).</td>
</tr>
<tr>
<td>fullname</td>
<td>string</td>
<td></td>
<td>Descriptive name of the user (required).</td>
</tr>
<tr>
<td>email</td>
<td>string</td>
<td></td>
<td>Email address of the user (required).</td>
</tr>
<tr>
<td>sendEmail</td>
<td>boolean</td>
<td>false</td>
<td>If provided, specifies whether to send a confirmation email.</td>
</tr>
</tbody>
</table>

### Context Variables

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jelly.new.username</td>
<td>string</td>
<td>Username of the user being created.</td>
</tr>
</tbody>
</table>

Examples

**Create User**

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>The id of the project role you want to delete.</td>
</tr>
<tr>
<td>confirm</td>
<td>string</td>
<td></td>
<td>To delete the project role this value must be set to 'true'.</td>
</tr>
</tbody>
</table>

Examples

Deleting a project role from JIRA

```xml
<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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>The id of the project role you want to query</td>
</tr>
<tr>
<td>var</td>
<td>string</td>
<td></td>
<td>The name of the variable you wish to have the returned role actors placed into</td>
</tr>
</tbody>
</table>

Examples

Returning a List of role actors and iterating over the users in each of these actors.
jira:GetProjectRole

This tag will return the project role with the given id.

Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>The id of the project role you want</td>
</tr>
<tr>
<td>var</td>
<td>string</td>
<td></td>
<td>The name of the variable you wish to have the project role assigned to</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectkey</td>
<td>string</td>
<td></td>
<td>The key of the project you want to query</td>
</tr>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>The id of the project role you want to query</td>
</tr>
<tr>
<td>var</td>
<td>string</td>
<td></td>
<td>The name of the variable you want the returned 'role actors' object assigned to</td>
</tr>
</tbody>
</table>

Examples

Return a list of users for a given 'Role Actors' object

```
<jiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:GetDefaultRoleActors projectroleid="1" var="roleactors">
    <core:forEach var="actor" items="${roleactors.users}">
      ${actor.name}
    </core:forEach>
  </jira:GetDefaultRoleActors>
</jiraJelly>
```
<JiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib"
xmlns:core="jelly:core">
  <jira:GetProjectRoleActors projectkey="MKY" projectroleid="1" var="roleactors">
    <core:forEach var="actor" items="${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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>The name of the project role</td>
</tr>
<tr>
<td>var</td>
<td>string</td>
<td></td>
<td>The name of the variable you want the returned result assigned to.</td>
</tr>
</tbody>
</table>

**Examples**

Determining if a project role is unique.

```xml
<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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td></td>
<td>The key of the issue to link from (origin of link - required)</td>
</tr>
<tr>
<td>linkKey</td>
<td>string</td>
<td></td>
<td>The key of the issue to link to (destination of link - required)</td>
</tr>
<tr>
<td>linkDesc</td>
<td>string</td>
<td>linkDesc is taken from the 'Inward Description' or the 'Outward Description' of the link. (required)</td>
<td></td>
</tr>
</tbody>
</table>

**Examples**

Create a Link between two existing issues
Create two issues and link them

```xml
<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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>string</td>
<td></td>
<td>Username of the user to log in.</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td></td>
<td>Password of the user to log in.</td>
</tr>
</tbody>
</table>

**Context Variables**

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jelly.user</td>
<td>User</td>
<td>User logged in.</td>
</tr>
<tr>
<td>jelly.username</td>
<td>string</td>
<td>Username of the User logged in.</td>
</tr>
</tbody>
</table>

**Examples**

Login a user in with username and password and set in context

```xml
<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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
</table>
### jira:RemoveActorsFromDefaultProjectRole

This tag will remove a list of role actors from a given project role for a given project.

#### Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>The id of the project role you wish to remove members from</td>
</tr>
<tr>
<td>actors</td>
<td>string</td>
<td></td>
<td>A comma delimited list of users or groups you wish to remove from the project role</td>
</tr>
<tr>
<td>projectkey</td>
<td>string</td>
<td></td>
<td>The key of the project the project role is associated with</td>
</tr>
<tr>
<td>actortype</td>
<td>string</td>
<td></td>
<td>The type of 'actor' you are working with. Currently the available options are 'atlassian-group-role-actor' or 'atlassian-user-role-actor'</td>
</tr>
</tbody>
</table>

#### Examples

**Removing a list of groups from a default project role**

```xml
<jiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib">
  <jira:RemoveActorsFromDefaultProjectRole projectroleid="1"
    actors="jira-administrators, jira-users"
    actortype="atlassian-group-role-actor" />
</jiraJelly>
```
<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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>Username of the user to remove (required).</td>
</tr>
</tbody>
</table>

**Examples**

Remove User

```xml
<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.

```xml
[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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterid</td>
<td>int</td>
<td></td>
<td>The id of the filter which will be used to run the search request.</td>
</tr>
<tr>
<td>size-var</td>
<td>string</td>
<td></td>
<td>The variable that will hold the number of issues returned from the search request.</td>
</tr>
<tr>
<td>var</td>
<td>string</td>
<td></td>
<td>The variable that will hold the issues returned from the search request.</td>
</tr>
</tbody>
</table>

Examples

Running a search request and iterating through the keys of the returned issues

```xml
<jiraJelly xmlns:jira="jelly:com.atlassian.jira.jelly.JiraTagLib"
  xmlns:core="jelly:core">
  <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

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>project-key</td>
<td>string</td>
<td></td>
<td>The key of the project you want to add the component to (required).</td>
</tr>
<tr>
<td>componentName</td>
<td>string</td>
<td></td>
<td>Name of the component (required).</td>
</tr>
</tbody>
</table>
| assigneeType     | string  |               | Default assignee type (required). **Assignee Types:**
  - projectDefault
  - componentLead
  - projectLead
  - unassigned |

Examples

Select a Component Assignee
**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 associated 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**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>string</td>
<td>Currently logged in user</td>
<td>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).</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td></td>
<td>The key of the issue to execute the transition on.</td>
</tr>
<tr>
<td>workflowAction</td>
<td>string</td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>resolution</td>
<td>string</td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td><strong>assignee</strong></td>
<td>string</td>
<td>The username of the user to assign an issue to during the transition. The &quot;user&quot; 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 &quot;-automatic-&quot; to let JIRA assign the issue to the default assignee.</td>
<td></td>
</tr>
<tr>
<td><strong>fixVersions</strong></td>
<td>string</td>
<td>A comma separated list of version ids or names to set as &quot;fix for&quot; versions during the transition. The &quot;user&quot; executing the transition must have permissions to set &quot;fix for&quot; versions if this attribute is supplied. Please note that the transition must expect the &quot;fix for&quot; 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.</td>
<td></td>
</tr>
<tr>
<td><strong>comment</strong></td>
<td>string</td>
<td>The comment to add to the issue during the transition. The &quot;user&quot; 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.</td>
<td></td>
</tr>
<tr>
<td><strong>groupLevel</strong></td>
<td>string</td>
<td>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 cannot specify the roleLevel parameter.</td>
<td></td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Default Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>--------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>projectroleid</td>
<td>int</td>
<td></td>
<td>The id of the project role you want to query</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td></td>
<td>The name you want the project role updated with</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td></td>
<td>The description you want the project role updated with</td>
</tr>
</tbody>
</table>

**Examples**

**Updating a project role**

```xml
<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
  - schemeld (required)
  - security (required)
  - type (required)
- AddIssueSecurityLevel
  - name (required)
  - description (required)
  - Output
    - jelly.issue.scheme.level.id
- CreateIssueSecurityScheme

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
• 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
  • schemeld (required)
  • confirm (required))
• RemoveProject
  • pld (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:
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. -->
<j:script var="script" ns-type="script" script="true">
  <!-- 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="${key}-managers" permissions="Edit,Assign,Affordable,Resolve,Close,Delete" type="group"/>
    <jira:AddPermission group="${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,Affordable,Comment" type="group"/>
    <jira:AddPermission group="help-desk-group" permissions="Browse,Assign,Affordable,Comment" type="group"/>
    <jira:AddPermission group="webadmin-group" permissions="Browse,Assign,Affordable,Comment" type="group"/>
    <jira:AddPermission group="unix-admin-group" permissions="Browse,Assign,Affordable,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.
<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](http://support.atlassian.com/jira/toolkit) is useful in conjunction 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](https://plugins.atlassian.com)

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](http://support.atlassian.com). See the [JIRA Toolkit documentation](http://support.atlassian.com/jira/toolkit) for details.

### Developer Guides

Please refer to the [JIRA Developer Documentation](http://docs.atlassian.com).

### Building JIRA from Source

Please be aware that the content on this page is is not included in our [Atlassian Support Offerings](https://www.atlassian.com/software/support) and consequently Atlassian [can not guarantee providing any support for it](https://www.atlassian.com/software/support). Please be aware that this material is provided for your information only and using it is done so at your own risk.

If you're looking to browse through the source code for reference, you can access it if you have a license.
Commercial users at any level receive access to JIRA’s source code (note, evaluation license holders are not permitted 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.

⚠️ There is a known issue where the Workflow Designer source is missing from the JIRA source code which is being tracked under JIRA-29615 - Workflow Designer source is missing from the source code delivered with JIRA source code folder (Open)

- please keep this in mind when building JIRA from source.

### 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

---

**Building a JIRA WAR file from a JIRA Source release**

1. Ensure you have JDK 1.7 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:**
5. Add Maven's bin directory to your path. For example,

**On Windows:**

```bash
> set PATH=%M2_HOME%\bin
```

You can set this via My Computer >> Properties >> Advanced >> Environment Variables again if you wish.

**On Mac/Linux:**

```bash
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.

<table>
<thead>
<tr>
<th>Libraries</th>
<th>Maven groupId and artifactId</th>
<th>Version</th>
<th>Download URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>activation</td>
<td>javax.activation:activation</td>
<td>1.0.2</td>
<td><a href="http://repository.jboss.org/nexus/service/local/repositories/deprecated/content/javax/activation/activation-1.0.2.jar">http://repository.jboss.org/nexus/service/local/repositories/deprecated/content/javax/activation/activation-1.0.2.jar</a></td>
</tr>
<tr>
<td>jms</td>
<td>javax.jms:jms</td>
<td>1.1</td>
<td><a href="http://repository.jboss.org/nexus/content/groups/public-jboss/javax/jms/jms-1.1/">http://repository.jboss.org/nexus/content/groups/public-jboss/javax/jms/jms-1.1/</a></td>
</tr>
<tr>
<td>jmxri and jmxtools</td>
<td>com.sun.jmx:jmxri and com.sun.jdmk:jmxtools</td>
<td>1.2.1</td>
<td><a href="http://www.oracle.com/technetwork/java/javase/tech/download-jsp-141676.html">http://www.oracle.com/technetwork/java/javase/tech/download-jsp-141676.html</a> (Download 'JMX 1.2.1 Reference Implementation')</td>
</tr>
<tr>
<td>Library</td>
<td>Group</td>
<td>Version</td>
<td>URL</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>jta</td>
<td>jta:jta</td>
<td>1.0.1B</td>
<td><a href="http://www.oracle.com/technetwork/java/javaee/jta/index.html">http://www.oracle.com/technetwork/java/javaee/jta/index.html</a> (Download 'Class Files 1.0.1B')</td>
</tr>
<tr>
<td>mail</td>
<td>javax.mail:mail</td>
<td>1.3.2</td>
<td><a href="http://www.oracle.com/technetwork/java/javamail-1-3-2-138617.html">http://www.oracle.com/technetwork/java/javamail-1-3-2-138617.html</a></td>
</tr>
<tr>
<td>ojdbc6</td>
<td>com.oracle:ojdbc6</td>
<td>11.2.0.2.0</td>
<td><a href="http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-12010-090769.html">http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-12010-090769.html</a> (Download 'ojdbc6.jar' under 'Oracle Database 11g Release 2 (11.2.0.2.0) JDBC Drivers')</td>
</tr>
</tbody>
</table>

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 .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
     jmxri.jar
     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.zip file.
   - On Mac/Linux:
     cd $HOME/Downloads
     unzip jndi-1_2.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:
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:

```bash
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:

```bash
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=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
```

   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:
   ```bash
   C:\atlassian-jira-4.3-source\> build.bat
   ```

   On Mac/Linux:
   ```bash
   > 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-dist-4.3.war

   An unpacked version of your JIRA's 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 6.0 or 7.0

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.

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/check/SearchRequestRelationCheck.class
```

To deploy this class, place it in:

```
<jira-install>/WEB-INF/classes/com/atlassian/jira/appconsistency/integrity/check/SearchRequestRelationCheck.class
```
...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

Where can I find the API documentation?

- The API documentation can be found here: [https://developer.atlassian.com/static/](https://developer.atlassian.com/static/) (see JIRA section).
- Information about each of the remote APIs (REST, XML-RPC, JSON-RPC) can be found here: [JIRA Remote API Reference](https://developer.atlassian.com/static/).
- Please note, the API documentation is not available for download.

Other resources
We also recommend that you refer to the developer resources available via this page: Development Resources.

Each of the "Preparing for..." pages provide helpful information for each JIRA version, such as change reports like this one: JIRA stable (Java) API changes for 6.0 (Clirr report).

Managing Webhooks

Webhooks are user-defined HTTP POST callbacks. They provide a lightweight mechanism for letting remote applications receive push notifications from JIRA, without requiring polling. For example, you may want any changes in JIRA bugs to be pushed to a test management system, so that they can be retested.

JIRA supports webhooks for all standard issue-related events (i.e. not custom events), e.g. issue_created, issue_assigned, etc. Issue events are pushed to a specified URL in JSON format. You can constrain the scope of the webhook by specifying the desired issue events and/or JQL.

Please read the JIRA Webhooks Overview page which contains detailed information on JIRA webhooks. This page only contains instructions on how to use the Webhooks user interface in the JIRA administration console.

On this page:
- Viewing webhooks in JIRA
- Managing webhooks via the JIRA administration console
- Extending your webhook
- Known issues

Viewing webhooks in JIRA

To view webhooks in JIRA,

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🌐 > System. Select Advanced > Webhooks to open the Webhooks page, which shows a list of all existing webhooks.
   🔄 Keyboard shortcut: g + g + start typing webhooks
3. Click the summary of the webhook in the left ‘Webhooks’ column to display the details of the webhook.

Managing webhooks via the JIRA administration console

Webhooks can be registered (created), edited and deleted via the JIRA administration console, REST calls or the installation of a "Remote App". The instructions on this page describe how to manage webhooks via the JIRA administration console. Please see the JIRA Webhooks Overview page for information on the other methods.

To manage webhooks in JIRA,
1. Log in as a user with the **JIRA Administrators** global permission.

2. Choose **”> System.** Select **Advanced > Webhooks** to open the Webhooks page, which shows a list of all existing webhooks.

   **Keyboard shortcut:** `g + g +` start typing **webhooks**

3. Create/register, edit or delete webhooks as desired.
   - **Create** — Click **Add a Webhook**, enter the webhook details and click **Create**.
   - **Edit** — Select the desired webhook in the summary list on the left, click **Edit**, update the webhook details and click **Save**.
   - **Delete** — Select the desired webhook in the summary list on the left, and click **Delete** in the webhook details.
   - **Disable** — Select the desired webhook in the summary list on the left, and click **Disable** in the webhook details. Disabling a webhook prevents the webhook from firing.
   - **Enable** (disabled webhooks only) — Select the desired disabled webhook in the summary list on the left, and click **Enable** in the webhook details.

**Tips for managing a webhook**

- **Constraining the webhook to a set of issues/issue events** — By default, a webhook will fire for **all events of all issues** in a JIRA installation. You can constrain this to a set of issues and a set of issue events (or no issue events). To do this, specify the desired set of issues (via a JQL statement) and the set of issue events, when you create or edit a webhook. For example, the webhook in the screenshot above will only fire when bugs with an affected version of "1.0" are updated.

- **Omitting the JSON from the POST** — If you are integrating with a system where only a POST URL is required to trigger the action, and having a body with data breaks the integration, you can configure the webhook so that it does not send the JSON with the POST. To do this, select the **Exclude details** check box when editing your webhook.

- **Determining the changes in an 'Issue Updated' event** — A changelog is sent with the 'Issue Updated' event (excepting the **issue_generic_event**). You can listen for this event and check the changelog for the changes.

**Extending your webhook**

*Trigger your webhook from a workflow post function*

See **Configuring Workflows**.

Please note, if a webhook is associated with a post-function, you cannot be delete the webhook. You must disassociate it from the post-function first.

*Add an issue key variable to your webhook*

You can add an issue key variable to your webhook by inserting `$issue.key` in your webhook URL. When the webhook fires for an issue, the issue key will be substituted for the variable in the URL.

For example, say you have the following web hook URL with the `$issue.key` variable:

http://example.com/bamboo/rest/api/latest/queue/BAMBOO/${issue.key}

If the web hook is fired for a JIRA issue with the key **JIRA-1234**, the URL that the web hooked is posted to will be:

http://example.com/bamboo/rest/api/latest/queue/BAMBOO/JIRA-1234

**Known issues**

- Post function web hooks will not fire if added to the **Create Issue** workflow transition. We recommend that you configure your web hook to fire from the **issue_created** event instead.

- If you are using webhooks in JIRA OnDemand, there is a known issue that restricts the port numbers that can be specified in the webhook URL. If you do not follow the guidelines below, then the webhook will not trigger.
  - If you are using HTTP and specify a port number in the webhook URL, the port number must be 80. Note, if you do not specify a port number, the webhook will work.
  - If you are using HTTPS and specify a port number in the webhook URL, the port number must be
443. Note, if you do not specify a port number, the webhook will work.

### JIRA Installation and Upgrade Guide

The pages listed below contain information on installing and upgrading JIRA:

- JIRA Requirements
- Supported Platforms
- Installing JIRA
- Running the Setup Wizard
- Connecting JIRA to a Database
- Upgrading JIRA
- Migrating JIRA to Another Server
- Establishing Staging Server Environments for JIRA
- Important Directories and Files
- Tomcat security best practices
- Customizing Your JIRA Installation
- Deployment Planning Activity

You can also skip the installation process by using JIRA OnDemand.

**OnDemand**

- No installation required, get started in 5 minutes
- Option to migrate to your own server later
- Choose from a set of supported add-ons to install
- Free technical support during your trial
- We won’t call you unless you want us to

**Start My Free Trial**

### 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 Virtualizing JIRA (JIRA on VMware).**

- JIRA Client/Server Software Requirements
  - 1. Browser
  - 2. Java
  - 3. Application Server
  - 4. Database
  - 5. Notes
- JIRA Server Hardware Recommendations
  - JIRA Server Hardware Recommendation for Evaluation
  - JIRA Server Hardware Recommendation for Production
- Next Steps
No hardware? No problem! Try using JIRA OnDemand.

- No installation required, get started in 5 minutes
- Option to migrate to your own server later
- Choose from a set of supported add-ons to install

JIRA Client/Server Software Requirements

Please read the Supported Platforms page for JIRA, which lists the required server and client software supported by JIRA 6.1, 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, 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).

5. Notes

- Virus checking software are a common cause of performance problems. In particular, Symantec must be uninstalled from the server that you want to install JIRA on, as it is known to dramatically reduces JIRA performance (even stopping the services does not prevent it from slowing JIRA down).
  For more information, see this knowledge base article: Crashes and Performance Issues Troubleshooting

### JIRA Server Hardware Recommendations

**JIRA typically will not perform well in a tightly constrained, shared environment - examples include an A WS micro.t1 instance.** Please be careful to ensure that your choice of hosting platform is capable of supplying sustained processing and memory capacity for the server.

#### JIRA Server Hardware Recommendation for Evaluation

During evaluation, JIRA will run well on any reasonably fast workstation computer (eg. something purchased within the last two years). 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.

There are two ways to evaluate JIRA:

1. Start immediately with JIRA OnDemand and then migrate to a local production server later, or simply continue to use JIRA OnDemand.
2. Install JIRA on a local computer and then migrate this to a production server later.

#### JIRA Server Hardware Recommendation for Production

The hardware required to run JIRA in production depends on a number of different JIRA configurations (eg. projects, issues, custom fields, permissions, etc) as well as the maximum number of concurrent requests that the system will experience during peak hours. Here are some general guide lines:

- For a small number of projects (10-20) with 1,000 to 5,000 issues in total and about 100-200 users, a recent server (multicore CPU) with 2 GB of available RAM and a reasonably fast hard drive (7200rpm or faster) should cater for your needs.
- For a greater number of issues adding more memory will help. We have reports that having 2GB of RAM to JIRA is sufficient for instances with around 200,000 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 multicore CPU machine will increase the concurrency of processing the requests and therefore speed up the response time for your users.
- For reference we have a server that has a 2 Intel(R) Xeon(R) CPU E5520 @ 2.27GHz (16 logical cores) with 32GB of RAM. This server runs Apache, various monitoring systems, and two JIRA instances:
  - Our public JIRA site that has approximately: 145,000 issues, 255,000 comments, 120 custom fields, and 115 projects.
  - Our support JIRA site that has approximately: 285,000 issues, 2,500,000 comments, 75 custom fields, and 22 projects.

Please note that performance heavily depends on your dimensions and your usage pattern, much more than what is simply covered here. Therefore we have written a guide on the different methods you can use to scale JIRA in your environment.

A quick note that your JIRA database's size is predominantly dominated by these three large tables: change items, comments and issues stored in your JIRA instance. Also, the type of custom fields and the values they hold may also increase the size of your JIRA database, eg. a free text custom field that is on every issue with grow the database size if the value of that field is large.

**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 the JAVA_HOME
   - Linux-based computers
   - Windows-based computers
3. Confirming that Java Works
4. 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 "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b15)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)
```

2. Setting the 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.

The default path for the bundled JRE with JIRA is C:\Program Files\Atlassian\JIRA\jre. If using a 32-bit install in 64-bit system, this would be C:\Program Files (x86)\Atlassian\JIRA\jre.

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 "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b15)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)
```
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 lists the supported platforms for JIRA 6.1 only. If a particular platform or a particular platform's version is not noted on this page, then *we do not support it for JIRA 6.1.*

**Not using JIRA 6.1?** The information below does not apply to you. See the following pages instead:

- Supported Platforms for JIRA 6.0
- Supported Platforms for JIRA 5.2
- Documentation for older JIRA versions

Further information:

- 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.
- 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.

Supported platforms for JIRA 6.1

<table>
<thead>
<tr>
<th>Supported platform(s)</th>
<th>Supported version(s)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Java platforms</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle JDK / JRE (formerly Sun JDK / JRE)</td>
<td>1.7</td>
<td>• JIRA is a pure Java-based application and should run on any supported operating system, provided that the JDK / JRE requirements are satisfied.</td>
</tr>
<tr>
<td><strong>Operating systems</strong></td>
<td></td>
<td>• Please see here for information on Anti-Virus in JIRA</td>
</tr>
<tr>
<td>Microsoft Windows</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Linux / Solaris | • JIRA is a pure Java-based application and should run on any supported operating system, provided that the JDK / JRE requirements are satisfied.  
• 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.  
• 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.  
• NFS mounts are not supported due to Lucene requirements. Please see the Index Writer docs for further info. |
<p>| Virtualisation | VMware | • Please read our <a href="guide">Virtu&lt;br&gt;alizing JIRA (JIRA on&lt;br&gt;VMware)</a> guide for&lt;br&gt;information on the&lt;br&gt;required configuration&lt;br&gt;of VMWare.&lt;br&gt;• We are unable to&lt;br&gt;provide any support&lt;br&gt;for VMWare itself.&lt;br&gt;• All of the operating&lt;br&gt;systems listed in the&lt;br&gt;'Operating systems'&lt;br&gt;rows above are&lt;br&gt;supported for&lt;br&gt;VMware. |
| Application Servers | Apache Tomcat | • 7.0.29&lt;br&gt;• 6.0.32 | • Deploying multiple&lt;br&gt;Atlassian applications&lt;br&gt;in a single Tomcat&lt;br&gt;container is not supported. Please&lt;br&gt;see <a href="guide">Deploying Multiple Atlassian&lt;br&gt;Applications in a Single Tomcat&lt;br&gt;Container</a> for reasons&lt;br&gt;why we do not&lt;br&gt;support this configuration.&lt;br&gt;• Whilst JIRA may&lt;br&gt;work with other&lt;br&gt;versions of Apache&lt;br&gt;Tomcat 6 &amp; 7, it has&lt;br&gt;not been tested on&lt;br&gt;them and may cause&lt;br&gt;problems. We&lt;br&gt;recommend using&lt;br&gt;these versions. |
| Databases | Oracle | • 11G with Oracle&lt;br&gt;11.2.x drivers | • Using Advanced&lt;br&gt;Compression Option&lt;br&gt;(ACO) is not supported. |
| | MySQL | • 5.x (excluding 5.0) with the JDBC&lt;br&gt;Connector/J 5.1 | • The MySQL driver is&lt;br&gt;no longer bundled with JIRA, as of&lt;br&gt;JIRA 5.2. Please see&lt;br&gt;the <a href="guide">JIRA 5.2 Upgrade&lt;br&gt;Notes</a> for reasons&lt;br&gt;why we do not bundle&lt;br&gt;this driver.&lt;br&gt;• MySQL 5.0.x is not&lt;br&gt;supported as it is&lt;br&gt;essentially EOL and&lt;br&gt;only covered under&lt;br&gt;Oracle Sustaining&lt;br&gt;Support.&lt;br&gt;• We recommend&lt;br&gt;running MySQL in&lt;br&gt;strict mode. |</p>
<table>
<thead>
<tr>
<th>Database</th>
<th>Supported Versions</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostgreSQL</td>
<td>9.0 with the PostgreSQL Driver 9.0.x, 8.4 with the PostgreSQL Driver 8.4.x, 8.3 with the PostgreSQL Driver 8.4.x</td>
<td></td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>2008 with the JTDS 1.2.4 driver, 2008 R2 with the JTDS 1.2.4 driver, 2005 with the JTDS 1.2.4 driver</td>
<td>Express Editions are not supported.</td>
</tr>
<tr>
<td>HSQLDB</td>
<td>Supported for evaluation use only</td>
<td>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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Web Browsers</th>
<th>Supported Versions</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>Latest stable version supported</td>
<td>Minimum screen resolution of 1024 x 768 (when browsers are maximised). Please refer to our Patch Policy on fixing browser issues.</td>
</tr>
<tr>
<td>Microsoft Internet Explorer</td>
<td>8.0, 9.0, 10.0</td>
<td>Minimum screen resolution of 1024 x 768 (when browsers are maximised). Please refer to our Patch Policy on fixing browser issues. 'Compatibility View' is not supported.</td>
</tr>
<tr>
<td>Mozilla Firefox</td>
<td>Latest stable version supported</td>
<td>Minimum screen resolution of 1024 x 768 (when browsers are maximised). Please refer to our Patch Policy on fixing browser issues.</td>
</tr>
<tr>
<td>Platform/Functionality</td>
<td>JIRA end of support</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Project keys with characters other than letters, numbers and the underscore</td>
<td>From JIRA 6.1 (announcement)</td>
<td></td>
</tr>
<tr>
<td>CVS/ViewCVS integration</td>
<td>From JIRA 6.1 (announcement)</td>
<td></td>
</tr>
<tr>
<td>“Voters and Watchers” and “Labels” plugins</td>
<td>From JIRA 6.1 (announcement)</td>
<td></td>
</tr>
</tbody>
</table>

**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 new versions of databases, web browsers, etc, are released, the cost of supporting multiple platforms grows exponentially, making it harder to provide the level of support our customers have come to expect from us. Therefore, we no longer support platform versions marked as end-of-life by the vendor, or very old versions that are no longer widely used.

**On this page (most recent announcements first):**

- Deprecated plugins for JIRA (01 August 2013)
- Deprecated source control systems for JIRA (8 July 2013)
- End of support for characters other than letters, numbers and the underscore in JIRA's project key (13 May 2013, updated 6 August 2013)
- Deprecated Java platforms for JIRA (18 January 2013)
- End of support for project key format configuration (22 November 2012)
- Deprecated application servers for JIRA (27 August 2012)
- Deprecated databases for JIRA (24 July 2012)
- 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 plugins for JIRA (01 August 2013)

This section announces the end of Atlassian support for the Voters and Watchers and Labels plugins. Atlassian will not fix bugs related to either the "Voters and Watchers" or the "Labels" plugins past the support end date. Note, these plugins are not bundled with JIRA. In case of both of these plugins, the respective functionality has been largely replaced as part of the JIRA core since version 4.2 (Voters and Watchers) and 4.1 (Labels).

We will stop supporting the following plugins:

• From JIRA 6.1, due in the second half of 2013, JIRA will no longer support the "Voters and Watchers" or the "Labels" plugins.

We made this decision in order to reduce our plugin support, to reduce testing time and help us speed up our ability to deliver market-driven features. If you have questions or concerns regarding this announcement, please email eol-announcement at atlassian dot com.

"Voters and Watchers" and "Labels" plugins End of Support Notes:

• JIRA 6.0 will be the last major version of JIRA to officially support the "Voters and Watchers" and "Labels" plugins.
• JIRA 6.0.x and earlier versions should continue to work with the "Voters and Watchers" and "Labels" plugins.
• JIRA 6.1 will not be tested with the "Voters and Watchers" and "Labels" plugins.

Deprecated source control systems for JIRA (8 July 2013)

This section announces the end of Atlassian support for CVS/ViewCVS. At the support end date, all functionality related to CVS/ViewCVS integration will be removed from JIRA. Also, Atlassian will not fix bugs related to CVS/ViewCVS integration past the support end date. Note, this feature was never available in JIRA OnDemand.

We will stop supporting the following source control system:

• From JIRA 6.1, due in the second half of 2013, JIRA will no longer include nor support CVS/ViewCVS integration.

We made this decision in order to reduce our source control system support, to reduce testing time and help us speed up our ability to deliver market-driven features. If you have questions or concerns regarding this announcement, please email eol-announcement at atlassian dot com.

CVS/ViewCVS End of Support Notes:

• JIRA 6.0 will be the last major version of JIRA to officially include and support CVS integration.
• JIRA 6.0.x and earlier versions should continue to work with CVS/ViewCVS.
• CVS/ViewCVS integration is available as an unsupported open source plugin for JIRA 6.1.x and later: http://bitbucket.org/atlassian/jira-cvs-plugin/overview.
• The JIRA CVS plugin has been created from the JIRA source code, so JIRA 6.1.x and later should work with CVS/ViewCVS without any additional setup.
• JIRA 6.1 will not be tested with CVS/ViewCVS.
• If you need an alternative that is supported by Atlassian, Atlassian's FishEye integrates with JIRA and supports CVS repository access.

End of support for characters other than letters, numbers and the underscore in JIRA’s project key (13 May 2013, updated 6 August 2013)

This section announces the end of support for characters other than letters, numbers and the underscore in JIRA’s project key in downloadable JIRA. Note, project key configuration was never allowed in JIRA OnDemand.

From JIRA 6.1, due in the second half of 2013, we will only support customized project keys that meet all of the conditions specified below:

• Only letters, numbers or the underscore character can be used.
• The first character must be a letter from the Modern Roman Alphabet. All other letter characters in the key
must also be from the Modern Roman Alphabet.
• The first and all other letter characters must be in upper case.

Examples of supported keys: PRODUCT_2013; R2D2; MY_EXAMPLE_PROJECT.
Examples of unsupported keys: 2013PROJECT (first character is not a letter); PRODUCT-2012 (hyphens are not supported).

We understand that some customers may be in the position where their current project keys do not meet the above requirements. We want to help these customers migrate to a supported format. To do this, we will implement the highly-voted feature request: Editing Project Keys (JIRA-2703). We have started work on this and should complete it for the JIRA 6.1 release. Please note, we will continue to provide the same level of support for your current customized project keys until this feature has been implemented.

Why are you only supporting letters, numbers and the underscore character for customized project keys?

We are only supporting letters, numbers and the underscore character, as supporting every possible character for customized project keys is a challenging task for JIRA plugins, integration with other Atlassian products, as well as core JIRA functionality. Hence, we will only be supporting the most commonly used characters. From your feedback and our research, letters and numbers are commonly used for customized project keys. Also, the underscore is by far the most commonly used separator character, with hyphens a distant second. However, we will not be supporting hyphens as they interfere with the separator that JIRA uses for issue numbers.

End of support notes:

• Please note that our previous announcement for the end of project key format configuration in JIRA 6.0 no longer applies. This is largely due to the great feedback provided by you, our customers.
• In JIRA 6.0, we will continue to support customized project keys, however we recommend that customers only use customized project keys if they meet the conditions specified above.
• Classic Boards in JIRA Agile are not compatible with the supported project key format noted above. You must use the default project key format. If you are not using the Classic Boards, you can change to a supported project key format.

Deprecated Java platforms for JIRA (18 January 2013)

This section announces the end of Atlassian support for Java 6 (JRE and JDK 1.6). End of support means that Atlassian will not fix bugs in Java 6 (JRE and JDK 1.6) past the support end date.

We will stop supporting the following Java platform:

• From JIRA 6.0, due towards the middle of 2013, JIRA will no longer support Java 6 (JRE and JDK 1.6).

We are ending support for Java 6 (JRE and JDK 1.6), as Oracle has announced the end of public updates for Java 6: Java SE 6 End of Public Updates Notice. We are committed to helping our customers understand this decision and to assist you in upgrading JIRA, (we recommend the JIRA Installer for the upcoming 6.0 release, which will bundle its own Java 7 environment).

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.

Java 6 (JRE and JDK 1.6) End of Support Notes:

• JIRA 5.2 will be the last major version of JIRA to officially support Java 6 (JRE and JDK 1.6).
• JIRA 5.2.x and earlier versions will continue to work with Java 6 (JRE and JDK 1.6). However, we will not fix bugs affecting Java 6 (JRE and JDK 1.6) past the support end date.
• JIRA 6.0 will not be tested with Java 6 (JRE and JDK 1.6).

End of support for project key format configuration (22 November 2012)

⚠️ This announcement no longer applies. Please see this announcement regarding project key format customization instead.
This section announces the end of Atlassian support for project key format configuration in downloadable JIRA (note, project key configuration was never allowed in JIRA OnDemand). End of support means that Atlassian will not fix bugs (past the support end date) that occur due to customization of the project key.

- From JIRA 6.0, due in the first half of 2013, JIRA will no longer support project key format configuration.

We are ending support for project key format configuration, as changing the product key format will break JIRA plugins, integration with other Atlassian products, as well as core JIRA functionality. We are committed to helping our customers understand this decision and guiding them back to use the standard project key format.

Deprecated application servers for JIRA (27 August 2012)

This section announces the end of Atlassian support for Tomcat 5.5.x for JIRA WAR. 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 5.2, due towards the end of 2012, JIRA will no longer support Tomcat 5.5.x.

We are ending support for Tomcat 5.5, as the Apache Tomcat team has announced the end of life (EoL) for version 5.5 as the 30th September 2012 (see announcement). We are committed to helping our customers understand this decision and assisting them in migrating to Tomcat 6.0.32 or later, 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.

**Tomcat 5.5.x End of Support Notes:**

- JIRA 5.1 will be the last major version of JIRA to officially support Tomcat 5.5.x.
- Tomcat 6.0.32 will be supported in JIRA 5.2. We are planning on adding support for Tomcat 7.0.29 in JIRA 5.2.
- JIRA 5.1.x and earlier versions will continue to work with Tomcat 5.5.x. However, we will not fix bugs affecting Tomcat 5.5.x past the support end date.
- JIRA 5.2 will not be tested with Tomcat 5.5.x.

Deprecated databases for JIRA (24 July 2012)

This section announces the end of Atlassian support for certain database versions for JIRA. End of support means that Atlassian will not fix bugs related to certain database versions past the support end date.

We will stop supporting the following database versions:

- From JIRA 5.2, due towards the end of 2012, JIRA will no longer support Postgres 8.2.

We are ending support for Postgres 8.2, as the PostgreSQL Global Development Group has already announced that the end of life (EoL) for version 8.2. We are committed to helping our customers understand this decision and to assist you in migrating to a different 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.

**PostgreSQL End of Support Notes:**

- The PostgreSQL Global Development Group's support for PostgreSQL 8.2 ended on December 2011 (see the PostgreSQL versioning policy).
- JIRA 5.1 will be the last major version of JIRA to officially support PostgreSQL 8.2.
- PostgreSQL 8.3 and 8.4 will be supported in JIRA 5.2.
- JIRA 5.1.x and earlier versions will continue to work with PostgreSQL 8.2. However, we will not fix bugs affecting PostgreSQL 8.2 past the support end date.
- JIRA 5.2 will not be tested with PostgreSQL 8.2.

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 last 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**

<table>
<thead>
<tr>
<th>Web Browsers</th>
<th>Support End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safari 4</td>
<td>When JIRA 4.3 releases (due in Quarter 1 2011)</td>
</tr>
<tr>
<td>Firefox 3.0.x</td>
<td>When JIRA 4.3 releases (due in Quarter 1 2011)</td>
</tr>
</tbody>
</table>
**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

<table>
<thead>
<tr>
<th>Application Servers</th>
<th>Support End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>JBoss 4.2.2</td>
<td>When JIRA 4.1 releases, due late Q1 2010</td>
</tr>
<tr>
<td>Oracle WebLogic 9.2</td>
<td>When JIRA 4.2 releases, due Q3 2010</td>
</tr>
<tr>
<td>IBM WebSphere 6.1</td>
<td>When JIRA 4.2 releases, due Q3 2010</td>
</tr>
</tbody>
</table>
• 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 customizations 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 dated 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

<table>
<thead>
<tr>
<th>Java Platform</th>
<th>Support End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Platform 5 (JDK/JRE 1.5)</td>
<td>When JIRA 4.2 releases, due Q3 2010</td>
</tr>
</tbody>
</table>

• 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

<table>
<thead>
<tr>
<th>Web Browsers</th>
<th>Support End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Explorer 6</td>
<td>When JIRA 4.2 releases (target Q3 2010)</td>
</tr>
</tbody>
</table>

- **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 this 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 customize 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:

```bash
> 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 customizing 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 Optimizing 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 file 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

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 and Install the 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.

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 customize 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' and no password, 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.
- Depending on how big the JIRA instance may become, you may need to increase the maximum number of files available on the operating system. This is further covered in our Loss of Functionality due to Too Many Open Files Error KB - please review it for further information.

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.
**********************************************
```

To start JIRA using the service, you can execute the /etc/init.d/jira script.

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:

- \[\text{http://<computer_name_or_IP_address>:<HTTP_port_number>}\]

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 \[\text{logs/catalina-YYYY-MM-DD.log}\] 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 customizing 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 Optimizing 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.

ℹ️ 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`
   ℹ️ 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.

ℹ️ 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 the instructions on this page.

Before you begin

Please ensure that you have installed Java and set JAVA_HOME. Also refer to the Supported Platforms page for details about which Java (as well as other) platforms are supported by JIRA.

⚠️ 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.

On this page:
- 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

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.
      ℹ️ 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.
      ℹ️ 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
To set this, do one of the following:

- Edit the `jira-application.properties` file and set the value of the 'jira.home' property to the desired location for your JIRA Home Directory (this location should be something different than the application directory, or you may run into problems later). If you are specifying this location's path on Windows, use double backslashes (``") between subdirectories. For example, `X: \path\to\JIRA\Home`

  - If you define an UNC path in Microsoft Windows, be sure to double escape the leading backslash: `\\machinename\path\to\JIRA\home`

- 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):
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 customizing 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 Optimizing 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.

<table>
<thead>
<tr>
<th>Recommended distributions</th>
<th>WAR distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Require minimal setup</td>
<td>Requires manual configuration</td>
</tr>
<tr>
<td>Pre-packaged with the Apache Tomcat application server</td>
<td>Requires building and deployment to an existing application server installation</td>
</tr>
<tr>
<td>Include the JIRA Configuration Tool</td>
<td>Does not include the JIRA Configuration Tool</td>
</tr>
<tr>
<td>Recommended for all users</td>
<td>Suitable only for system administrators</td>
</tr>
</tbody>
</table>

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 or 7.0

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:
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).

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.
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:**

```xml
<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 the 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 or 7.0

This guide describes how to install the JIRA WAR distribution on Tomcat 6.0 or Tomcat 7.0, 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 recommended 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 6.0/7.0:

- 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 internationalized 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
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 Customizing your JIRA installation directory files

(This section is optional and recommended for experts only.)

2.1.1 How to customize files in your JIRA installation directory

If you wish to customize 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 customize in your JIRA Installation Directory, the more difficult it will be to upgrade JIRA or migrate JIRA to another server, as your customizations 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>` 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.

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
2.2 JIRA Home

- Edit the `jira-application.properties` file and set the value of the 'jira.home' property to the desired location for your JIRA Home Directory (this location should be something different than the application directory, or you may run into problems later). If you are specifying this location's path on Windows, use double backslashes ("\") between subdirectories. For example, `X:\path\to\JIRA\Home`

  If you define an UNC path in Microsoft Windows, be sure to double escape the leading backslash: `\\machinename\path\to\JIRA\home`

- 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 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` directory 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 the relevant archive depending on your Tomcat version (see below), 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).

<table>
<thead>
<tr>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>jcl-over-slf4j-x.y.z.jar</td>
</tr>
<tr>
<td>jul-to-slf4j-x.y.z.jar</td>
</tr>
<tr>
<td>log4j-x.y.z.jar</td>
</tr>
<tr>
<td>slf4j-api-x.y.z.jar</td>
</tr>
<tr>
<td>slf4j-log4j12-x-y-z.jar</td>
</tr>
</tbody>
</table>

**5. Configure JIRA's context in Tomcat**

A JIRA 'context' now needs to be set up in Tomcat. To do this, add the below configuration to the Tomcat server.xml file. The value of docBase will need to be modified to point to the JIRA WAR file.

```xml
<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

```n

- If installing JIRA 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-5.x.war).
- If installing on Linux, ensure the path does not use any special characters.

These details can also be configured in conf/Catalina/localhost/jira.xml, however they cannot be configured in both server.xml and jira.xml. If they are, it will cause significant problems with the JIRA instance.

**6. Modify Tomcat’s server.xml to handle internationalized characters correctly**

In order for JIRA to correctly display internationalized 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:

1. 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:
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
</table>
| 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 server.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/Catalina/localhost/jira.xml file instead of the conf/server.xml file of your Tomcat installation directory? Although this is fine, be sure that you do not also have details in the server.xml file 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 JIRA-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.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, then please consider installing one of the 'recommended' distributions of JIRA.

**User-contributed notes**

Do you have experiences to share with Tomcat 6.0.x/7.0.x and JIRA? We welcome your thoughts. Please see the user-contributed Tomcat 6.0.x and 7.0.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**

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**

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 do the following:

- Configure your JIRA Home Directory
- Configure your database connection
- Tune your database connection
- Configure the webserver, including the TCP ports that JIRA runs through and SSL configuration.
On this page:
- Starting the JIRA Configuration Tool
- Configuring the JIRA Home Directory
- Configuring the database connection
- Configuring JIRA's web server
- Tuning JIRA's database connections

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.
- Whenever you configure or reconfigure JIRA's server-level settings using this tool, JIRA must be restarted so it can recognise these changes.

Starting the JIRA Configuration Tool

The JAVA_HOME environment variable must be set to use the JIRA configuration tool. If it has not been set already, follow the instructions in Installing Java to set it.

To start the JIRA configuration tool:
- **Windows**: Open a command prompt and run config.bat in the bin sub-directory of the JIRA Installation Directory.
- **Linux/Unix**: Open a console and execute config.sh in the bin sub-directory of the JIRA Installation Directory.

This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.

The JIRA configuration tool can be run with a graphical user interface or via a command-line interface. The following sections show the graphical user interface, but the functionality is the same regardless of the interface.

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.
3. Click 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.

Screenshot: JIRA configuration tool — 'JIRA Home' tab
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

Screenshot: JIRA configuration tool — 'Database' tab
Configuring JIRA’s web server

The JIRA configuration tool can also be used to configure JIRA’s web server, specifically the TCP ports and the SSL configuration. Follow the relevant instructions linked below:

- Changing JIRA’s TCP Ports
- Running JIRA over SSL or HTTPS

Screenshot: JIRA configuration tool — ‘Web Server’ tab
Tuning JIRA's database connections

For more information about the functionality of the Advanced tab, see Tuning Database Connections.

Screenshot: JIRA configuration tool — 'Advanced' tab
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.

ℹ️ 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 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 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:
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 7.0.
- 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:

   `tomcat7 //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 `regedit32.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>*

   For JIRA 5.2 and above run these `tomcat7` commands, but for JIRA 5.1 or below run the `tomcat` 6 command instead

   - To increase the maximum memory JIRA can use (the default will already be 256MB), run:

     `tomcat7 //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:

     `tomcat7 //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:

     `tomcat7 //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 theVerbose GC (garbage collection) logging, execute the following command in the command prompt:

```
tomcat7 //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:

```
tomcat7 //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 `tomcat7 //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](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:

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

- Java 6 is not supported by JIRA 6.0 and later. 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/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**

The JIRA Setup Wizard helps you configure the basic settings for your new JIRA server. These include the
server language, database, license key, administrator accounts and mail notifications.

To get started, access your new JIRA server in a browser, after you have installed JIRA. Your server will be available at the following URL, if you are using the default port: http://<jira-server-name>:8080.

*The JIRA Setup Wizard will only display the first time after you install JIRA. Once you have completed it, you cannot run it again. However, every setting configured in the Setup Wizard can be configured via the JIRA administration console.*

**On this page:**
- Step 1. Configure the language and database
- Step 2. Configure the application properties
- Step 3. Specify your license key
- Step 4. Set up the administrator account
- Step 5. Configure email notifications
- Next steps

### Step 1. Configure the language and database

**Screenshot: Configuring the server language and database connection for a new JIRA server**

#### Set the server language

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.

#### Configure a database for JIRA

Choose between connecting JIRA to the bundled database or your own database.

<table>
<thead>
<tr>
<th>Database Connection</th>
<th>Recommended for</th>
<th>Instructions</th>
<th>Notes</th>
</tr>
</thead>
</table>

Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](https://creativecommons.org/licenses/by/2.5/au/).
<table>
<thead>
<tr>
<th>Bundled database</th>
<th>Evaluations only</th>
<th>Go to the next step. The bundled HSQL database will be automatically configured by the Setup wizard.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• HSQL databases are prone to corruption and are <strong>not suitable for use in a production environment</strong>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Your own database</th>
<th>Production use</th>
<th>1. Choose a database. See our list of supported databases first.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2. Configure the database connection. If you need help, see the guides on <a href="#">Connecting JIRA to a Database</a>. Note, the fields displayed on this screen are identical to those on the JIRA Configuration Tool.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Your external database must be a newly-created (or empty) database.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Database connection pool — You cannot configure your database connection pool size through the Setup Wizard. You can do this subsequently using the <a href="#">JIRA Configuration Tool</a> or manually (described on each specific database configuration guide).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MySQL database — The MySQL driver is not bundled with JIRA (see <a href="#">Connecting JIRA to MySQL</a>). You need to copy the driver into the lib folder of your JIRA installation and restart JIRA/JIRA service before completing the Setup Wizard.</td>
</tr>
</tbody>
</table>

---

**Step 2. Configure the application properties**

*Screenshot: Configuring the application properties for a new JIRA server*
Configure the Title, Mode and Base URL

<table>
<thead>
<tr>
<th>Setting</th>
<th>Instructions</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Title</td>
<td>Choose a title that helps identify your installation and its purpose.</td>
<td>• The application title will be displayed on the JIRA login page and the dashboard.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• After you have completed the Setup Wizard, you may also want to configure the logo and color scheme of your installation.</td>
</tr>
<tr>
<td>Mode</td>
<td>Choose a mode that suits how you use your issue tracker.</td>
<td>• Setting the mode to public enables public signup. Note, that allowing anyone to sign up can cause you to exceed the user limit on your JIRA license.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A public issue tracker can be useful for gathering feedback and bug reports directly from customers. A private issue tracker may be more suitable for tracking the development progress of your team.</td>
</tr>
<tr>
<td>Base URL</td>
<td>Specify the base URL that users will use to access your JIRA site.</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>

- 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. 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.

Further information:

- If you need to change these settings after setting up JIRA, you can configure them via the JIRA administration console. For details, see Configuring JIRA Options.
- JIRA will store your automated backups, file attachments and indexes in your JIRA Home Directory.

(Optional) Import data into your JIRA installation

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 data link. The 'Import Existing Data' page will display — see Restoring Data for further instructions.

Please also note:

- 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.

Step 3. Specify your license key

Screenshot: Creating a my.atlassian.com account and license key

![Specify Your License Key](image)

Screenshot: Generating a license key for an existing my.atlassian.com account
You are required to enter a JIRA license key before you can use JIRA. If you don't have a JIRA license key, you can get the Setup Wizard to create an evaluation license for you, as shown above. Evaluation license keys will allow you to use a fully functional installation of JIRA for 30 days.

License keys for Atlassian products are linked to your account at my.atlassian.com. If you don't have a my.atlassian.com account, you can create one and get the Setup Wizard to create an evaluation license for you, as shown above.

Some customers have encountered Javascript and CSS errors on the 'Specify Your License Key' page. See this knowledge base article: JIRA Setup Wizard Fails with Errors on Licence Key Setup Page.

Step 4. Set up the administrator account

Screenshot: Setting up a JIRA system administrator account
Enter the details for the administrator account for the JIRA installation. The account will be granted the JIRA system administrator permission.

You can create additional JIRA system administrator and JIRA administrator accounts after you have set up JIRA.

Step 5. Configure email notifications

Screenshot: Setting up email notifications for JIRA

This step is optional. You can configure email notifications after you have set up JIRA if you wish.

If you want to configure email notifications at this stage, you will need to set up a connection to a mail server. See this page for further instructions: Configuring JIRA's SMTP Mail Server to Send Notifications

✅ Congratulations, you have completed setting up your new JIRA installation!

Next steps

The Welcome screen will lead you through creating demonstration project for your new JIRA installation, or you can opt to set it up yourself.

For more information on getting started with JIRA, see the JIRA 101 guide.

Detailed information on using and administering JIRA can be found in the rest of the JIRA documentation:

- JIRA User's Guide
Next Steps - Creating a Project

If you have already created a project via the 'Welcome' screen, you can skip this step.

This page tells you how to add a new project or configure an existing project, if you are a JIRA administrator. You'll need to navigate to the project administration screen for this:

Choose 🔄 > Projects.

- To create a new project, choose the Add Project button and follow the wizard. See Creating a project below for details.
- To configure an existing project, choose the project name from the Project list and configure the settings. See Configuring a project below for details.

On this page:
- Creating a project
- Simple Issue Tracking project
- Software Development project
- Configuring a project
- A note about project administrators

Related pages:
- Migrating from Other Issue Trackers

Creating a project

To create a new project in JIRA:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose Projects > Create Project.
3. Choose the type of project that you want to create.
4. Enter the project details. Note, choose the project key carefully, as it is not a simple task to change it after the project is created.
5. Choose Submit to add the new project.

Notes:
- Choosing JIRA Classic or Project Management creates the default JIRA project.
- The project key will be used as the prefix of this project's issue keys (e.g. 'TEST-100'). Choose one that is descriptive and easy to type.
- The project lead is a unique project role. Choose the person who manages the project as the project lead. 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.
This project provides you with a quick and easy way to get JIRA up and running for simple issue tracking. For details on working with this project, see Simple Issue Tracking project.

Software Development project

This project provides you with a template to use for software development. For details on working with this project, see Software Development project.

Configuring a project

**To configure a project in JIRA:**

1. Log in as a user with the JIRA Administrators global permission.
2. Choose 🗄 > Projects.
   - Keyboard shortcut `g + g` and then start typing the name of your project.
3. Use the tabs on the left to navigate between the different project settings. Read the sections below for a description of each setting.
   - **Project details**
   - **Issue types**
   - **Workflows**
   - **Screens**
   - **Fields**
   - **Settings**
   - **Roles**
   - **Versions**
   - **Components**
   - **Permissions**
   - **Notifications**
   - **Development tools**

**Project details**

**To edit the project's details:**

1. Click **Edit Project** at the top of the **Project Summary** page.
2. In the resulting **Edit Project** dialog box, edit the following fields:
   - **Name** — type a descriptive name. This can be changed later if you wish.
   - **Key** — Click **edit key** to change the project key. Read this page before you edit the project key: Ed iting a Project Key.
   - **URL** — an optional URL associated with this project, e.g. pointing to project documentation.
   - **Project Avatar** — an image (48x48 pixels) that represents the project. You can either use the default image, i.e.:
     ![Default Avatar](image)
     or choose a different image. The process for choosing a project avatar is similar to that for choosing a user avatar. If you prefer not to use an image for your project, 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.

To edit the project Category, enter the project Administration section by clicking a project name and then click the link next to the **Category** field (located under the project name). JIRA can search for all the issues in a particular project category, and can display projects sorted by the project category, but a project category is not part of a project hierarchy. JIRA does not support sub-projects or parent projects. In addition a JIRA project can only belong to one category.

If no categories exist, click the **Add** link on the following **No Project Category** 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 behavior: each field can be required/optional, rich text/plain text, hidden/visible. You define this behavior 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**

  - **Application Links** (Configure Project Links) — if you have linked your JIRA instance to other Atlassian applications, like Confluence, FishEye or other JIRA instances, you will be able to link this JIRA project to areas of those applications that contain information relating to your project or team. For example, Confluence spaces, FishEye repositories, JIRA projects (in another JIRA instance), etc. This allows you to take advantage of integration points between these applications. See [Linking to Another Application](#) for information about application links and project links.

**Roles**

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.
    
    By default, new projects also have their 'Default Assignee' set to 'Unassigned.' You can change this here if you want to set it to be a specific role, i.e. 'Project Lead.'
  - **Project Roles** — members are users/groups who fulfill 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.

⚠️ Please note, the Default Notification Scheme (shipped with JIRA) is associated with all new projects by default. This means that if you have an outgoing (SMTP) mail server set up, that email notifications will be sent as soon as there is any activity (e.g. issues created) in the new project.

**Development tools**

The Development tools section gives you an overview of the development tools that are connected and which users can use the integration features between them:

- **View permission** - This section lists which users can see the development tools integration features (like the Create Branch link) on the view issue screen, as well as other development-related information, like commits, reviews and build information. This ability is controlled by the "View Development Tools" project permission.
- **Applications** - This section shows which development tools are connected to JIRA via application links and are eligible to use the development tool features in JIRA.

**A note about project administrators**

A project administrator in JIRA is someone who has the project-specific Administer Projects project permission, but not necessarily the JIRA Administrator global permission.

Without the JIRA Administrator global permission, however, project administrators can do the following:

- Edit the project name
- Edit the project description
- Edit the project avatar image
- Edit the project URL
- Edit the project lead
- Edit project role membership
- Define project components
- Define project versions
- View, but not select nor edit the project's schemes (notification scheme, permission scheme, etc)
Next Steps - Adding Users

On this page:
- Viewing users
- Adding users
  - Who can see a user?
  - Creating a user
  - Inviting users
- 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
- Changing a username
- Adding a property to a user
- Deactivating a user
- Deleting a user
- Notes

Viewing users

To view a list of JIRA users:

1. Log in as a user with the JIRA Administrators global permission.
2. Choose > User Management.
   Keyboard shortcut: g + g + start typing users

3. To restrict the list of users shown, use the Filter form at the top of the page.
4. To view details and login information about a user in the list, click their Username or Email Address.

Adding users

Users can be created via any of the following methods:

- Add the user directly into JIRA — see Creating a user below. You can create one user at a time, using
this method.
- Invite users via email — see Inviting Users below. You can invite multiple users at the same time, using this method.
- Allow users to sign up — see Enabling Public Signup.
- Automatically create users when issue/comment creation emails are received from unknown email addresses — You can use a mail handler to allow JIRA to create issues or comments via emails received. The handler can also be configured to create new users based on the sender's email address. See Creating Issues and Comments from Email.
- Connect to an Internal Directory with LDAP Authentication — see Copying Users on First Login.

ℹ️ Please Note: If you have a user limited license (e.g. starter license) and have reached your user limit, any further users created will not have permission to log in to JIRA.

Who can see a user?
Any JIRA user can see another JIRA user's full name and username, for example, see Mark Lassau. The email address visibility is controlled by a configuration item; see “User email visibility” on the Configuring JIRA Options documentation page.

Group membership and the ability to edit users is only available to administrators.

Any JIRA user can see their own details, including group memberships, update their own password, and change certain user preferences (for example, time zone and language).

Creating a user

To create a user:

1. Open the User browser (see Viewing Users above) and click the Create User button to open the 'Create New User’ dialog box.
2. Enter the Username, 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.

Inviting users

You can invite one or more users to JIRA via email. Note, JIRA's SMTP mail server must be configured to send notifications before you can invite users via email.

To invite users to JIRA:

1. Open the User browser (see Viewing Users above) and click the Invite Users button to open the 'Invite Users' dialog box.
2. Enter the email addresses of the users that you want to invite. Enter each address on a new line or separate addresses using commas.
   Note, you cannot invite users by sending an invitation to a mailing list.
3. Click the Send button to send the invitations.
   - Each invitation can only be used to create a user under the email address that it was sent to, and can only be used once.
   - Each invitation will expire seven days after the day it was sent.
   - Your user license count will not be affected until users accept the invitation and the users are created.
   - Users that are created via the invitation will be added to the 'jira-users' group.

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.
This will display two lists; the one on the left shows all Available Groups, and the one on the right shows the Current Groups to which the user currently belongs.

2. Choose a group(s) and click the Join selected groups or Leave selected groups buttons to add/remove the user from the selected groups.

ℹ️ 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 assign a user to a project role:

1. 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:

<table>
<thead>
<tr>
<th>Project</th>
<th>Administrators</th>
<th>Developers</th>
<th>Observers</th>
<th>RequesterOnly</th>
<th>Users</th>
<th>Viewport Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angry Nerds</td>
<td>☑️</td>
<td>☐</td>
<td>☐</td>
<td>☑️</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>iPhone Dev</td>
<td>☑️</td>
<td>☐</td>
<td>☐</td>
<td>☑️</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

2. 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

To change 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.
2. In the resulting form, make the required changes the user’s Full Name and/or Email address.
   ⚠️ Do not clear the Active check box unless you want to deactivate this user.
3. Click Update to confirm the change.

Changing a user’s password

To change 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. Choose Actions > Set Password.
3. Enter and confirm the new password.
4. Click the Update button.

Changing a username

ℹ️ This feature is only available for downloadable instances of JIRA. It is not available in JIRA On
Demand.

JIRA Administrators can edit any **Username** in the JIRA Internal Directory (this is often referred to as the "rename user" feature). This ability is important to have if you wish to connect JIRA to an LDAP directory that does not follow the same username conventions. You also may want to do this if a staff member wishes to change their surname. Once the **Username** is changed in the internal directory, all parts of the JIRA interface display the updated **Username**.

**Note:** The **Username** is for the JIRA Internal Directory, and should not be confused with the user's display name, or **Full Name**, in the JIRA system.

There are some important exceptions that will prevent you from using this feature, of which you should be aware:

- Only **JIRA Administrators** can perform this function.
- JIRA cannot update **external** users – for example, users that are coming from an LDAP server or Crowd instance – it can only update users stored in the JIRA Internal Directory. (However, JIRA can update JIRA users stored in an "Internal Directory with LDAP Authentication.")
- If you are using your JIRA instance as a JIRA User Server for other applications, e.g., Confluence, you will not be able to use this feature. If you aren't sure about this, check under **User Management > JIRA User Server** to confirm that no external applications have been configured to use JIRA as a Crowd Server.

### Using a JIRA User Server for other applications

When JIRA is being used as a **User Server** for other applications, e.g. Confluence, we don't allow a user rename on the JIRA server as this would not be recognised on the other server. (And the other application would think that user was deleted and a new user was added.)

If you are happy to accept this behavior, then you can set a flag to allow the rename, as documented in the Knowledge Base article: **Cannot rename users despite upgrading/installing JIRA 6**

We are hoping to add the ability to detect renames from a remote Crowd or JIRA server sometime soon, see JIRA-32200.

### To change a username:

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. Choose **Actions > Edit Details**.
3. Edit the **Username**.
4. Click the **Update** button.

### Adding a property to a user

A 'Property' is an extra piece of information about a user that you can store in JIRA. A Property consists of a **Key** of your choice (eg. 'Phone number', 'Location') plus a corresponding **Value** (eg. '987 654 3210', 'Level Three'). Other than adding property data to the specified user, User Properties do not have an effect anywhere else in the project. Plugins, however, can frequently use this data.

### 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. Choose **Actions > Edit Properties**. The **Edit User Properties** screen will be displayed:
3. Enter the new **Key** and its **Value**, then click the **Add** button.

### Deactivating a user

JIRA administrators can 'deactivate' a JIRA user, which disables that user's access to JIRA. This avoids the need for a JIRA administrator to delete the user's account from the system.

This feature is useful when a JIRA user leaves an organisation because a deactivated user's history of JIRA activity is preserved on the system. If a user with a deactivated JIRA account rejoins the organisation at some point in the future, their JIRA user account can be re-activated.
To deactivate a user account:

1. Locate the user in the User browser (see Viewing Users above) and click their Edit link in the Operations column.
2. In the resulting form, clear the Active check box.
3. Click Update to confirm the change.
   
   To re-enable the user again, repeat the steps above but instead, select the Active check box.

While a JIRA user account has been deactivated, that user:

- Will no longer be able to log in to JIRA.
- Cannot be assigned issues or added as a watcher to issues (whenever issues are created or edited).
- A user who was assigned, was watching or had reported any issues in JIRA before their account is deactivated, will still appear as the respective assignee, watcher or reporter of those issues. This situation remains until another user is specified as the assignee or reporter of these issues, or the deactivated user is removed as a watcher from them.
- A user who voted on any issues in JIRA before their account is deactivated, will continue to appear as a voter on these issues.
- Will continue to appear on the JIRA user interface with '(Inactive)' displayed after their name, where applicable.
- Can still be used to filter issues in a JIRA search query.
- Will not receive any email notifications from JIRA, even if they continue to remain the assignee, reporter, or watchers of issues.
- Will not count towards your JIRA user license limit. Refer to the JIRA Users global permission explanation on Managing Global Permissions for more information.

Please Note:

- Users who are project or component leads cannot be deactivated. To deactivate these users, assign other users as the relevant project or component leads first.
- Any JIRA site's users who are configured in an external Atlassian Crowd user directory and deactivated in Crowd, will be deactivated in JIRA.
- With the exception of JIRA users configured with 'delegated LDAP authentication', JIRA does not deactivate users who are configured and deactivated/disabled in an external Microsoft Active Directory or LDAP-based user directory.

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 history of activity on JIRA.

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.
   
   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.
2. Take any actions required to disassociate the user with JIRA. These may include:
   - Reassigning any issues assigned to the user.
   - Bulk-editing the issues created by the user and change the 'Reporter' to someone else. You'll need the 'Modify Reporter' permission to do this. 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.
   - Changing the owner of shared dashboards owned by the user. See Managing Shared Dashboards.
   - Changing the project lead for any projects that the user is a lead of.
3. If there are no issues assigned to, or reported by the user, and the user has not commented on any issues, the confirmation screen will display a Delete button. Click this to proceed with the deletion.

Please Note:

- You cannot delete a user from JIRA if they have performed any of the following actions:
  - reported or been assigned to any issues
  - commented on any issues
1. 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.
2. 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 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 at the top of the screen to open the Create Issue dialog box.
   - Keyboard shortcut: c
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 that are not shown on this dialog box or you want to hide existing fields:
     a. Click the Configure Fields button at the top right of the screen.
     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: To create a series of similar issues — with the same Project and Issue Type — select the Create another check box at the bottom of the dialog.
5. When you are satisfied with the content of your issue, click the Create button.
   - If you selected the Create another check box (above), a new Create Issue dialog appears. This issue is 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.
- In certain text fields for an issue, you can link to other issues, insert macros, insert images and more. For more information, see Editing Rich-Text Fields.
- 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**.

- You may automatically become a **watcher** of the issues that you create, depending on the **Autowatch** setting in your user profile. Note, if you have not changed this setting, you will inherit the global Autowatch settings set by your JIRA administrator (in **Administration > User Preferences**).
- With appropriate configuration by your JIRA administrator, it is also possible to **create issues via email**.

**Screenshot: Example 'Create Issue' dialog box**

![Create Issue Dialog](image)

**Related topics**

- Sharing a Search Result
- Connecting JIRA to a Database

**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:

<table>
<thead>
<tr>
<th>'Recommended' distributions:</th>
<th>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.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAR distribution:</td>
<td>If you have set up a JIRA WAR installation, you need to manually configure your database connection.</td>
</tr>
</tbody>
</table>

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 database. A version of these instructions specific to Linux and JIRA is available.

**Before you begin**

- Check whether your version of PostgreSQL is supported. See Supported Platforms.
- 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 your old database to your new database, as described in Switching databases.
- Shut down JIRA before you begin, unless you are running the Setup Wizard.

**On this page:**

- Before you begin
- 1. Create and configure the PostgreSQL database
- 2. Copy the PostgreSQL JDBC driver to your application server (JIRA WAR only)
- 3. Configure your JIRA server to connect to your PostgreSQL database
- 4. Start JIRA
- Installation notes

1. Create and configure the PostgreSQL database

1. Create a database user (login role) 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.

   **Remember this database name,** as it will be used to configure JIRA's connection to this database in subsequent steps.

   ```sql
   CREATE DATABASE jiradb WITH ENCODING 'UNICODE';
   ```

   Or from the command-line:
3. Ensure that the user has permissions to connect to the database, and to create and write to tables in the database.

2. Copy the PostgreSQL JDBC driver to your application server (JIRA WAR only)

⚠️ **Skip this step** if you installed a *Recommended* distribution of JIRA, which includes the PostgreSQL JDBC driver. The **JIRA WAR** distribution does not include this driver.

1. Download the PostgreSQL JDBC driver from [http://jdbc.postgresql.org/download.html](http://jdbc.postgresql.org/download.html). (Obtain version 9.0 of the **JDBC 4** driver.)
2. Add the PostgreSQL JDBC driver jar to the `lib/` directory of your application server.

3. Configure your JIRA server to connect to your PostgreSQL database

There are three ways to configure your JIRA server to connect to your PostgreSQL database:

- **Using the JIRA setup wizard** (not applicable to **JIRA WAR**) — Use this method, if you have just installed JIRA and are setting it up for the first time. Your settings will be saved to the `dbconfig.xml` file in your **JIRA Home Directory**.
- **Using the JIRA configuration tool** (not applicable to **JIRA WAR**) — Use this method, if you have an existing JIRA instance. Your settings will be saved to the `dbconfig.xml` file in your **JIRA Home Directory**.
- **Manually** — Only use this method if you have a **JIRA WAR** instance or you have a console-only connection to your JIRA server. You will be required to manually update the `dbconfig.xml` file in your **JIRA Home Directory**.

**Instructions for each configuration method**

**JIRA setup wizard**

The **JIRA setup wizard** will display when you access JIRA for the first time in your browser.

1. In the first screen, 'Configure Language and Database', set **Database Connection** to **My own database**.
2. Set **Database Type** to **PostgreSQL**.
3. Fill out the fields, as described in the **Database connection fields** section below.
4. Test your connection and save.

**JIRA configuration tool**

1. Run the JIRA configuration tool as follows:
   - **Windows**: Open a command prompt and run `config.bat` in the `bin` sub-directory of the **JIRA Installation Directory**.
   - **Linux/Unix**: Open a

```bash
$ createdb -E UNICODE jiradb
```
console and execute `config.sh` in the `bin` sub-directory of the JIRA Installation Directory. This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.

2. Navigate to the **Database** tab and set **Database type** to PostgreSQL.

3. Fill out the fields, as described in the **Database connection fields** section below.

4. Test your connection and save.

5. Restart JIRA.

**Manually**

1. Locate 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 (shown below) into this file.
   - Note, the `<database-type/>` element must specify your type of database, e.g. `postgres`. If you forget to do this and you start JIRA, your database tables may be created incorrectly. See **Incorrect database type specified** if this happens to you.

2. Update the file, as described in the **Database connection fields** section below. Escape any `&` characters by adding `&` to the end of each one.

3. Save the file and restart JIRA.

**Database connection fields**

<table>
<thead>
<tr>
<th>Setup Wizard / Configuration Tool</th>
<th>dbconfig.xml</th>
<th>Description</th>
</tr>
</thead>
</table>

Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](https://creativecommons.org/licenses/by/2.5/au/).
| **Hostname** | Located in the `<url>` tag (bold text in example below): `<url>jdbc:postgresql://db server:5432/jiradb</url>` | The name or IP address of the machine that the PostgreSQL server is installed on. |
| **Port** | Located in the `<url>` tag (bold text in example below): `<url>jdbc:postgresql://db server:5432/jiradb</url>` | The TCP/IP port that the PostgreSQL server is listening on. You can leave this blank to use the default port. |
| **Database** | Located in the `<url>` tag (bold text in example below): `<url>jdbc:postgresql://db server:5432/jiradb</url>` | The name of your PostgreSQL database (into which JIRA will save its data). You should have created this in Step 1 above. |
| **Username** | Located in the `<username>` tag (see bold text in example below): `<username>jiradbuser</username>` | The user that JIRA uses to connect to the PostgreSQL server. You should have created this in Step 1 above. |
| **Password** | Located in the `<password>` tag (see bold text in example below): `<password>jiradbuser</password>` | The user's password — used to authenticate with the PostgreSQL server. |
| **Schema** | Located in the `<schema-name>` tag (see bold text in example below): `<schema-name>public</schema-name>` | The name of the schema that your PostgreSQL database uses. PostgreSQL 7.2 and later require 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. Ensure that your database schema name is lower-case, as JIRA cannot work with PostgreSQL databases whose schema names contain upper-case characters. |

Sample dbconfig.xml file

For more information about the child elements of `<jdbc-datasource/>` beginning with `pool` in the dbconfig.xml file above, see Tuning Database Connections.
4. Start JIRA

You should now have JIRA configured to connect to your PostgreSQL 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.

✅ Congratulations, you now have JIRA connected to your PostgreSQL database.

Installation notes

Please see JIRA and PostgreSQL for topics related to JIRA and PostgreSQL.

Connecting JIRA to MySQL

These instructions will help you connect JIRA to a MySQL database. A version of these instructions specific to Linux and JIRA is available.

Before you begin

- Check whether your version of MySQL is supported. See Supported Platforms.
- 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 your old database to your new database, as described in Switching databases.
- If you plan to set up Confluence and JIRA on the same MySQL server, please read the Confluence MySQL setup guide and configure your MySQL server to suit Confluence as well as JIRA. Note that the Confluence requirements are more strict than JIRA's, so you should configure MySQL to suit Confluence. This configuration will work for JIRA too.
- Shut down JIRA before you begin, unless you are running the Setup Wizard.
1. 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.
   **Remember this database name**, as it will be used to configure JIRA's connection to this database in subsequent steps.

   ```
   CREATE DATABASE jiradb CHARACTER SET utf8 COLLATE utf8_bin;
   ```

   (if you want your database to be named `jiradb`).

3. Ensure that the user has permission to connect to the database, and permission to create and populate tables. These can be provided with the following:

   ```
   GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, INDEX on <JIRADB>.* TO '<USERNAME>'@'<JIRA_SERVER_HOSTNAME>' IDENTIFIED BY '<PASSWORD>';
   ```

   flush privileges;

2. Copy the MySQL JDBC driver to your application server

If you are **upgrading JIRA** and you are using the **recommended MySQL driver** (Connector/J JDBC driver v5.1), you can skip the instructions in this section. The JIRA upgrade task will automatically copy over your existing driver to the upgraded installation.

To copy the MySQL JDBC driver to your application server:

1. Get the MySQL driver:
   - If you are **installing JIRA**, download the recommended MySQL **Connector/J JDBC driver v5.1**. You can download either the `.tar.gz` or the `.zip` file. Extract the jar for the driver (e.g. `mysql-connector-java-5.x.x-bin.jar`) from the archive.
   - If you are **upgrading JIRA** and you are not using the **recommended MySQL driver** (JDBC Connector/J 5.1), back up the driver from your JIRA installation before you upgrade. The driver will be in the `<JIRA installation directory>/lib/` directory.

2. Copy the MySQL JDBC driver jar to the `<JIRA installation directory>/lib/` directory for your new/upgraded installation. If you are installing JIRA using the Windows installer, you will need to do this step after running the Windows installer, but **before running the Setup Wizard**.

3. Restart JIRA / JIRA service.

4. If you are installing JIRA, skip the rest of the instructions on this page and access JIRA in your browser to run the Setup Wizard instead.

**Please note:**

- We recommend the Connector/J driver from MySQL (linked above). A user has reported experiencing problems with the Resin JDBC driver for MySQL.
3. Configure your JIRA server to connect to your MySQL database

There are three ways to configure your JIRA server to connect to your MySQL database:

- **Using the JIRA setup wizard** (not applicable to JIRA WAR) — Use this method, if you have just installed JIRA and are setting it up for the first time. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.
- **Using the JIRA configuration tool** (not applicable to JIRA WAR) — Use this method, if you have an existing JIRA instance. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.
- **Manually** — Only use this method if you have a JIRA WAR instance or you have a console-only connection to your JIRA server. You will be required to manually update the `dbconfig.xml` file in your JIRA Home Directory.

**Instructions for each configuration method**

**JIRA setup wizard**

The JIRA setup wizard will display when you access JIRA for the first time in your browser.

1. In the first screen, 'Configure Language and Database', set **Database Connection** to **My own database**.
2. Set **Database Type** to **MySQL**.
3. Fill out the fields, as described in the **Database connection fields** section below.
4. Test your connection and save.

**JIRA configuration tool**

1. Run the JIRA configuration tool as follows:
   - **Windows**: Open a command prompt and run `config.bat` in the `bin` sub-directory of the JIRA Installation Directory.
   - **Linux/Unix**: Open a console and execute `config.sh` in the `bin` sub-directory of the JIRA Installation Directory. This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.
2. Navigate to the **Database** tab and set **Database type** to **MySQL**.
3. Fill out the fields, as described in the Database connection fields section below.
4. Test your connection and save.
5. Restart JIRA.

Manually

1. Locate 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 (shown below) into this file.
2. Update the file, as described in the Database connection fields section below. Escape any ' & ' characters by adding ' & ' to the end of each one.
   - Note, the <database-type/> element must specify your type of database, e.g. mysql. If you forget to do this and you start JIRA, your database tables may be created incorrectly. See Incorrect database type specified if this happens to you.
3. Save the file and restart JIRA.

**Database connection fields**

<table>
<thead>
<tr>
<th>Setup Wizard / Configuration Tool</th>
<th>dbconfig.xml</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname</td>
<td>Located in the &lt;url&gt; tag (bold text in example below): ![jdbc:mysql://dbserver:3306/jiradb?useUnicode=true&amp;characterEncoding=UTF8&amp;sessionVariables=storage_engine=InnoDB&lt;/url&gt;]</td>
<td>The name or IP address of the machine that the MySQL server is installed on.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below):&lt;url&gt;jdbc:mysql://dbserver:3306/jiradb?useUnicode=true&amp;characterEncoding=UTF8&amp;sessionVariables=storage_engine=InnoDB&lt;/url&gt;</td>
<td>The TCP/IP port that the MySQL server is listening on. You can leave this blank to use the default port.</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Database</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below):&lt;url&gt;jdbc:mysql://dbserver:3306/jiradb?useUnicode=true&amp;characterEncoding=UTF8&amp;sessionVariables=storage_engine=InnoDB&lt;/url&gt;</td>
<td>The name of your MySQL database (into which JIRA will save its data). You should have created this in Step 1 above.</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>Located in the <code>&lt;username&gt;</code> tag (see bold text in example below):&lt;username&gt;jiradbuser&lt;/username&gt;</td>
<td>The user that JIRA uses to connect to the MySQL server. You should have created this in Step 1 above.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>Located in the <code>&lt;password&gt;</code> tag (see bold text in example below):&lt;password&gt;jiradbuser&lt;/password&gt;</td>
<td>The user's password — used to authenticate with the MySQL server.</td>
</tr>
</tbody>
</table>

### Sample dbconfig.xml file

- For more information about the child elements of `<jdbc-datasource/>` beginning with `pool` in the `dbconfig.xml` file above, see Tuning Database Connections.
- Both the JIRA setup wizard and database configuration tool also add the element `<validation-query>` to this file, which is usually required when running JIRA with default MySQL installations. See Surviving Connection Closures for more information.
- The database URL in the example below 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 database URL in the example below contains the `sessionVariables=storage_engine=InnoDB` parameter. We strongly recommend adding this parameter to avoid data corruption. See the Installation Notes section below for more information.
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>mysql</database-type>
  <jdbc-datasource>
    <url>jdbc:mysql://dbserver:3306/jiradb?useUnicode=true&amp;characterEncoding=UTF8&amp;sessionVariables=storage_engine=InnoDB</url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <username>jiradbuser</username>
    <password>password</password>
    <pool-min-size>20</pool-min-size>
    <pool-max-size>20</pool-max-size>
    <pool-max-wait>30000</pool-max-wait>
    <pool-max-idle>20</pool-max-idle>
    <pool-remove-abandoned>true</pool-remove-abandoned>
    <pool-remove-abandoned-timeout>300</pool-remove-abandoned-timeout>
    <validation-query>select 1</validation-query>
    <min-evictable-idle-time-millis>60000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>300000</time-between-eviction-runs-millis>
    <pool-test-while-idle>true</pool-test-while-idle>
    <validation-query-timeout>3</validation-query-timeout>
  </jdbc-datasource>
</jira-database-config>

4. Start JIRA

You should now have JIRA configured to connect to your MySQL 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.

✔ Congratulations, you now have JIRA connected to your MySQL database.

Installation notes

See JIRA and MySQL. Please also take note of the following:

Known issues and troubleshooting

- **Hostnames in permissions are compared as strings** - If you grant permission in MySQL to a hostname such as localhost then you must use the same string for the connecting to the database from JIRA. So using 127.0.0.1 won’t work even though it resolves to the same place. This mistake produces warnings about not finding tables because the JDBC connection did not have permission to create the new tables when JIRA was set up.

- **Connection closures** — If you are using a MySQL database with any of the following, you may experience problems with your connections dropping out (see [JIRA-15731](https://jira.atlassian.com/browse/JIRA-15731) for details). Please read **Surviving Connection Closures** for information on how to address this.
  - JIRA 3.13 or above,
  - version 5.5.25 or higher of Tomcat 5,
  - version 6.0.13 or higher of Tomcat 6.

- **Special characters for database password** — JIRA is not able to interpret special characters for database password. Please refer to additional hints about setting password for database [here](https).

- **Using the InnoDB storage engine** — 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 ([JIRA-24124](https://jira.atlassian.com/browse/JIRA-24124)), which may result in data corruption in JIRA. We strongly recommend specifying the sessionVariables=storage_engine=In
noDB parameter in your database URL (as stated above). Doing so ensures that tables written to JIRA's MySQL database will use the InnoDB storage engine, which supports ‘database transactions’ required by JIRA.

- **Binary logging** — 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](#).

## Connecting JIRA to Oracle

These instructions will help you connect JIRA to an Oracle database.

### Before you begin

- Check whether your version of Oracle is supported. See [Supported Platforms](#).
- 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 your old database to your new database, as described in [Switching databases](#).
- Shut down JIRA before you begin, unless you are running the Setup Wizard.

### On this page:

- Before you begin
- 1. Configure Oracle
- 2. Copy the Oracle JDBC Driver to Your Application Server (JIRA WAR Only)
- 3. Configure Your JIRA Server to Connect to Your Oracle Database
- 4. Start JIRA
- Installation notes

### 1. 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.

   ```
   create user <user> identified by <user_pass> default tablespace <tablespace_name> quota unlimited on <tablespace_name>;
   ```

   **Note:**

   - 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.

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>;
   ```

### 2. Copy the Oracle JDBC Driver to Your Application Server (JIRA WAR Only)

⚠️ **Skip this step** if you installed a ‘Recommended’ distribution of JIRA, which includes the Oracle JDBC driver. The JIRA WAR distribution does not include this driver.

1. Download the Oracle JDBC driver (from Oracle's site).
2. Add the appropriate Oracle JDBC driver jar (`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 version** of 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 `ArrayIndexOutOfBoundsException`.
- Note, that JDK 1.6 is not supported for JIRA from JIRA 6.0 and later. However, you should be able to use the Oracle JDBC driver for JDK 1.6 with JDK 1.7.

3. **Configure Your JIRA Server to Connect to Your Oracle Database**

There are three ways to configure your JIRA server to connect to your Oracle database:

- **Using the JIRA setup wizard** (not applicable to JIRA WAR) — Use this method, if you have just installed JIRA and are setting it up for the first time. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.
- **Using the JIRA configuration tool** (not applicable to JIRA WAR) — Use this method, if you have an existing JIRA instance. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.
- **Manually** — Only use this method if you have a JIRA WAR instance or you have a console-only connection to your JIRA server. You will be required to manually update the `dbconfig.xml` file in your JIRA Home Directory.

**Instructions for each configuration method**

**JIRA setup wizard**

The JIRA setup wizard will display when you access JIRA for the first time in your browser.

1. In the first screen, 'Configure Language and Database', set Database Connection to My own database.
2. Set Database Type to Oracle.
3. Fill out the fields, as described in the Database connection fields section below.
4. Test your connection and save.

**JIRA configuration tool**

1. Run the JIRA configuration tool as follows:
   - **Windows**: Open a command prompt and run `config.bat` in the `bin` sub-directory of the JIRA Installation Directory.
   - **Linux/Unix**: Open a console and execute `config.sh` in the `bin` sub-directory of the JIRA installation directory.
A Installation Directory. This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.

2. Navigate to the **Database** tab and set **Database type to Oracle**.

3. Fill out the fields, as described in the **Database connection fields** section below.

4. Test your connection and save. Any custom settings specified while manually configuring JIRA with Oracle (e.g., adding the `<connection-properties>SetBigStringTryClob=true</connection-properties>`) will be deleted. You will need to reinstate them manually. 

5. Restart JIRA.

**Manually**

1. Locate 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 (shown below) into this file.
   - Note, the `<database-type/>` element must specify your type of database, e.g. `oracle 11g`. If you forget to do this and you start JIRA, your database tables may be created incorrectly. See **Incorrect database type specified** if this happens to you.

2. Update the file, as described in the **Database connection fields** section below. Escape any ‘’ characters by adding ‘&’ to the end of each one.

3. Save the file and restart JIRA.

**Database connection fields**
<table>
<thead>
<tr>
<th>Setup Wizard / Configuration Tool</th>
<th>dbconfig.xml</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hostname</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below): <code>&lt;url&gt;jdbc:oracle:thin:@dbserver:1521:ORCL&lt;/url&gt;</code></td>
<td>The name or IP address of the machine that the Oracle server is installed on.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below): <code>&lt;url&gt;jdbc:oracle:thin:@dbserver:1521:ORCL&lt;/url&gt;</code></td>
<td>The TCP/IP port that the Oracle server is listening on. The default port number for Oracle is ‘1521’.</td>
</tr>
<tr>
<td><strong>SID</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below): <code>&lt;url&gt;jdbc:oracle:thin:@dbserver:1521:ORCL&lt;/url&gt;</code></td>
<td>The Oracle &quot;System Identifier&quot;. The default value for most Oracle servers is ‘ORCL’. If you are using the Oracle Express Edition, this will be ‘XE’.</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>Located in the <code>&lt;username&gt;</code> tag (see bold text in example below): <code>&lt;username&gt;jiradbuser&lt;/username&gt;</code></td>
<td>The user that JIRA uses to connect to the Oracle server. You should have created this in Step 1 above.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>Located in the <code>&lt;password&gt;</code> tag (see bold text in example below): <code>&lt;password&gt;jiradbuser&lt;/password&gt;</code></td>
<td>The user's password — used to authenticate with the Oracle server.</td>
</tr>
</tbody>
</table>

Sample dbconfig.xml file

For more information about the child elements of `<jdbc-datasource/>` beginning with pool in the dbconfig.xml file above, see Tuning Database Connections.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>oracle10g</database-type>
  <jdbc-datasource>
    <url>jdbc:oracle:thin:@dbserver:1521:ORCL</url>
    <driver-class>oracle.jdbc.OracleDriver</driver-class>
    <username>jiradbuser</username>
    <password>password</password>
    <pool-min-size>20</pool-min-size>
    <pool-max-size>20</pool-max-size>
    <pool-max-wait>30000</pool-max-wait>
    <pool-max-idle>20</pool-max-idle>
    <pool-remove-abandoned>true</pool-remove-abandoned>
    <pool-remove-abandoned-timeout>300</pool-remove-abandoned-timeout>
    <validation-query>select 1 from dual</validation-query>
    <min-evictable-idle-time-millis>60000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>300000</time-between-eviction-runs-millis>
    <pool-test-while-idle>true</pool-test-while-idle>
  </jdbc-datasource>
</jira-database-config>
```
4. Start JIRA

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.

**Congratulations, you now have JIRA connected to your Oracle database.**

Installation notes

Please see JIRA and Oracle.

Known issues and troubleshooting

- 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.

**Connecting JIRA to SQL Server 2005**

These instructions will help you connect JIRA to a Microsoft SQL Server 2005 database.

Before you begin

- Check whether your version of SQL Server is supported. See Supported Platforms. Note, SQL Server Express is not supported, however, it is possible to set up JIRA to work with this database. As it's not supported, we have no docs available for this configuration.
- 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 your old database to your new database, as described in Switching databases.
- Shut down JIRA before you begin, unless you are running the Setup Wizard.

1. 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.
      - 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. Remember this database schema name, as it will be used to configure JIRA’s connection to this database in subsequent steps.

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 (see the Error caused by SET NOCOUNT in MS SQL Server for an explanation). 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:

2. 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 includes the SQL Server JDBC driver. The JIRA WAR distribution does not include this 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.
3. Configure Your JIRA Server to Connect to Your SQL Server 2005 Database

There are three ways to configure your JIRA server to connect to your SQL Server database:

- **Using the JIRA setup wizard** (not applicable to JIRA WAR) — Use this method, if you have just installed JIRA and are setting it up for the first time. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.

- **Using the JIRA configuration tool** (not applicable to JIRA WAR) — Use this method, if you have an existing JIRA instance. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.

- **Manually** — Only use this method if you have a JIRA WAR instance or you have a console-only connection to your JIRA server. You will be required to manually update the `dbconfig.xml` file in your JIRA Home Directory.

**Instructions for each configuration method**

**JIRA setup wizard**

The **JIRA setup wizard** will display when you access JIRA for the first time in your browser.

1. In the first screen, 'Configure Language and Database', set **Database Connection** to **My own database**.
2. Set **Database Type** to **SQL Server**.
3. Fill out the fields, as described in the **Database connection fields** section below.
4. Test your connection and save.

**JIRA configuration tool**

1. Run the JIRA configuration tool as follows:
   - **Windows**: Open a command prompt and run `config.bat` in the `bin` sub-directory of the JIRA Installation Directory.
   - **Linux/Unix**: Open a console and execute `config.sh` in the `bin` sub-directory of the JIRA Installation Directory. This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.
2. Navigate to the **Database** tab and set **Database type** to **SQL Server**.
3. Fill out the fields, as described
in the Database connection fields section below.
4. Test your connection and save.
5. Restart JIRA.

Manually
1. Locate 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 (shown below) into this file.
2. Update the file, as described in the Database connection fields section below. Escape any ’&’ characters by adding ’&amp;’ to the end of each one.
   - Note, the <database-type/> element must specify your type of database, e.g. mssql. If you forget to do this and you start JIRA, your database tables may be created incorrectly. See Incorrect database type specified if this happens to you.
3. Save the file and restart JIRA.

**Database connection fields**

<table>
<thead>
<tr>
<th>Setup Wizard / Configuration Tool</th>
<th>dbconfig.xml</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname</td>
<td>Located in the &lt;url&gt; tag (bold text in example below):&lt;url&gt;jdbc:jtds:sqlserver://dbserver:1433/jiradb&lt;/url&gt;</td>
<td>The name or IP address of the machine that the SQL Server server is installed on.</td>
</tr>
<tr>
<td>Port</td>
<td>Located in the &lt;url&gt; tag (bold text in example below):&lt;url&gt;jdbc:jtds:sqlserver://dbserver:1433/jiradb&lt;/url&gt;</td>
<td>The TCP/IP port that the SQL Server server is listening on. You can leave this blank to use the default port.</td>
</tr>
<tr>
<td>Database</td>
<td>Located in the &lt;url&gt; tag (bold text in example below):&lt;url&gt;jdbc:jtds:sqlserver://dbserver:1433/jiradb&lt;/url&gt;</td>
<td>The name of your SQL Server database (into which JIRA will save its data). You should have created this in Step 1 above.</td>
</tr>
</tbody>
</table>
### JIRA 6.1 Documentation

#### Username
Located in the `<username>` tag (see bold text in example below):

```xml
<username> <username> </username>
```

The user that JIRA uses to connect to the SQL Server server. You should have created this in Step 1 above.

#### Password
Located in the `<password>` tag (see bold text in example below):

```xml
<password> <password> </password>
```

The user's password — used to authenticate with the SQL Server server.

#### Schema
Located in the `<schema-name>` tag (see bold text in example below):

```xml
<schema-name> <schema-name> </schema-name>
```

The name of the schema that your SQL Server database uses. You should have created this in Step 1 above.

---

**Sample dbconfig.xml file**

For more information about the child elements of `<jdbc-datasource/>` beginning with `pool` in the `dbconfig.xml` file above, see [Tuning Database Connections](https://confluence.atlassian.com/jiralandscapetopics/tuning-database-connections-97031).  

```xml
<?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>dbo</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>password</password>
    <pool-min-size>20</pool-min-size>
    <pool-max-size>20</pool-max-size>
    <pool-max-wait>30000</pool-max-wait>
    <pool-max-idle>20</pool-max-idle>
    <pool-remove-abandoned>true</pool-remove-abandoned>
    <pool-remove-abandoned-timeout>300</pool-remove-abandoned-timeout>
    <validation-query>select 1</validation-query>
    <min-evictable-idle-time-millis>60000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>300000</time-between-eviction-runs-millis>
    <pool-test-while-idle>true</pool-test-while-idle>
  </jdbc-datasource>
</jira-database-config>
```

4. Start JIRA

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.

✔️ Congratulations, you now have JIRA connected to your SQL Server database.

**Installation notes**

Please see [JIRA and MS SQL Server 2005](https://confluence.atlassian.com/jiralandscapetopics/INSTALLING-JIRA-2005-ON-MS-SQL-SERVER-97033).  

**Connecting JIRA to SQL Server 2008**
These instructions will help you connect JIRA to a Microsoft SQL Server 2008 database.

Before you begin

- Check whether your version of SQL Server is supported. See Supported Platforms.
  Note, SQL Server Express is not supported, however, it is possible to set up JIRA to work with this
database. This external blog post may be helpful, if you want to try this.
- 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 your old database to your new database, as described in Switching
databases.
- Shut down JIRA before you begin, unless you are running the Setup Wizard.

**On this page:**
- Before you begin
- 1. Create and Configure the SQL Server Database
- 2. Copy the SQL Server JDBC Driver to Your Application Server (JIRA WAR Only)
- 3. Configure Your JIRA Server to Connect to Your SQL Server 2008 Database
- 4. Start JIRA
- Installation notes

### 1. 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.
   - 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 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**:

![Configuration Panel](image)

- You will also need to access the **Server > Properties > Connections > Default Connections** properties box and clear the **no count** option.
2. 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 includes the SQL Server JDBC driver. The JIRA WAR distribution does not include this driver.

1. Download the SQL Server JDBC driver (v1.2.4) from [jTDS](https://jtds.kohsuke.org/).  
   ⚠️ 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](https://jira.atlassian.com/browse/JRA-5760) and [JRA-6872](https://jira.atlassian.com/browse/JRA-6872)), workflow problems ([JRA-8443](https://jira.atlassian.com/browse/JRA-8443) and Chinese character problems ([JRA-5054](https://jira.atlassian.com/browse/JRA-5054)).

2. Add the SQL Server JDBC driver jar (`jtds-1.2.4.jar`) to the `lib/` directory.

3. Configure Your JIRA Server to Connect to Your SQL Server 2008 Database

There are three ways to configure your JIRA server to connect to your SQL Server database:

- **Using the JIRA setup wizard** (not applicable to JIRA WAR) — Use this method, if you have just installed JIRA and are setting it up for the first time. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.

- **Using the JIRA configuration tool** (not applicable to JIRA WAR) — Use this method, if you have an existing JIRA instance. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.

- **Manually** — Only use this method if you have a JIRA WAR instance or you have a console-only connection to your JIRA server. You will be required to manually update the `dbconfig.xml` file in your JIRA Home Directory.
RA Home Directory.

Instructions for each configuration method

JIRA setup wizard

The JIRA setup wizard will display when you access JIRA for the first time in your browser.

1. In the first screen, 'Configure Language and Database', set Database Connection to My own database.
2. Set Database Type to SQL Server.
3. Fill out the fields, as described in the Database connection fields section below.
4. Test your connection and save.

JIRA configuration tool

1. Run the JIRA configuration tool as follows:
   - Windows: Open a command prompt and run \config.bat in the bin sub-directory of the JIRA Installation Directory.
   - Linux/Unix: Open a console and execute \config.sh in the bin sub-directory of the JIRA Installation Directory. This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.
2. Navigate to the Database tab and set Database type to SQL Server.
3. Fill out the fields, as described in the Database connection fields section below.
4. Test your connection and save.
5. Restart JIRA.

Manually

1. Locate 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 (shown below) into this file.

2. Update the file, as described in the Database connection fields section below. Escape any ’&’ characters by adding ’amp;’ to the end of each one.

   • Note, the <database-type/> element must specify your type of database, e.g. mssql. If you forget to do this and you start JIRA, your database tables may be created incorrectly. See Incorrect database type specified if this happens to you.

3. Save the file and restart JIRA.

Database connection fields

<table>
<thead>
<tr>
<th>Setup Wizard / Configuration Tool</th>
<th>dbconfig.xml</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname</td>
<td>Located in the &lt;url&gt; tag (bold text in example below):&lt;url&gt;jdbc:jtds:sqlserver://dbserver:1433/jiradb&lt;/url&gt;</td>
<td>The name or IP address of the machine that the SQL Server server is installed on.</td>
</tr>
<tr>
<td>Port</td>
<td>Located in the &lt;url&gt; tag (bold text in example below):&lt;url&gt;jdbc:jtds:sqlserver://dbserver:1433/jiradb&lt;/url&gt;</td>
<td>The TCP/IP port that the SQL Server server is listening on. You can leave this blank to use the default port.</td>
</tr>
<tr>
<td>Database</td>
<td>Located in the &lt;url&gt; tag (bold text in example below):&lt;url&gt;jdbc:jtds:sqlserver://dbserver:1433/jiradb&lt;/url&gt;</td>
<td>The name of your SQL Server database (into which JIRA will save its data). You should have created this in Step 1 above.</td>
</tr>
<tr>
<td>Username</td>
<td>Located in the &lt;username&gt; tag (see bold text in example below):&lt;username&gt;jiradbuser&lt;/username&gt;</td>
<td>The user that JIRA uses to connect to the SQL Server server. You should have created this in Step 1 above.</td>
</tr>
<tr>
<td>Password</td>
<td>Located in the &lt;password&gt; tag (see bold text in example below):&lt;password&gt;jiradbuser&lt;/password&gt;</td>
<td>The user’s password — used to authenticate with the SQL Server server.</td>
</tr>
<tr>
<td>Schema</td>
<td>Located in the &lt;schema-name&gt; tag (see bold text in example below):&lt;schema-name&gt;dbo&lt;/schema-name&gt;</td>
<td>The name of the schema that your SQL Server database uses. You should have created this in Step 1 above.</td>
</tr>
</tbody>
</table>
Sample dbconfig.xml file

For more information about the child elements of `<jdbc-datasource/>` beginning with `pool` in the `dbconfig.xml` file above, see Tuning Database Connections.

```xml
<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>mssql</database-type>
  <schema-name>dbo</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>password</password>
    <pool-min-size>20</pool-min-size>
    <pool-max-size>20</pool-max-size>
    <pool-max-wait>30000</pool-max-wait>
    <pool-max-idle>20</pool-max-idle>
    <pool-remove-abandoned>true</pool-remove-abandoned>
    <pool-remove-abandoned-timeout>300</pool-remove-abandoned-timeout>
    <validation-query>select 1</validation-query>
    <min-evictable-idle-time-millis>60000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>300000</time-between-eviction-runs-millis>
    <pool-test-while-idle>true</pool-test-while-idle>
  </jdbc-datasource>
</jira-database-config>
```

4. Start JIRA

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.

✅ Congratulations, you now have JIRA connected to your SQL Server database.

Installation notes

Please see JIRA and MS SQL Server 2008. Connecting JIRA to HSQLDB

These instructions will help you connect JIRA to a PostgreSQL database. A version of these instructions specific to Linux and JIRA is available.

Before you begin

⚠️ Do not use HSQLDB in production

JIRA ships with a built-in HSQLDB (HyperSQL DataBase) database. 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. See our list of supported databases on Supported Platforms.

- 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 your old database to your new database, as described in Switching databases.
On this page:

- Before you begin
- 1. Copy the HSQLDB Driver to Your Application Server (JIRA WAR Only)
- 4. Configure Your JIRA Server to Connect to Your HSQL Database
- 4. Start JIRA
- Installation notes

1. Copy the HSQLDB Driver to Your Application Server (JIRA WAR Only)

⚠️ **Skip this step** if you installed a ‘Recommended’ distribution of JIRA, which includes the PostgreSQL JDBC driver. The JIRA WAR distribution does not include this 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 three ways to configure your JIRA server to connect to your PostgreSQL database:

- **Using the JIRA setup wizard** (not applicable to JIRA WAR) — Use this method, if you have just installed JIRA and are setting it up for the first time. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.
- **Using the JIRA configuration tool** (not applicable to JIRA WAR) — Use this method, if you have an existing JIRA instance. Your settings will be saved to the `dbconfig.xml` file in your JIRA Home Directory.
- **Manually** — Only use this method if you have a JIRA WAR instance or you have a console-only connection to your JIRA server. You will be required to manually update the `dbconfig.xml` file in your JIRA Home Directory.

**Instructions for each configuration method**

**JIRA setup wizard**

The JIRA setup wizard will display when you access JIRA for the first time in your browser.

1. In the first screen, ‘Configure Language and Database’, set Database Connection to Database bundled with JIRA.
2. Go to the next step in the setup wizard. The setup wizard will set up the bundled HSQL database automatically.

**JIRA configuration tool**

1. Run the JIRA configuration tool as follows:
   - **Windows**: Open a command prompt and run `config.bat` in the
bin sub-directory of the JIRA Installation Directory.

- **Linux/Unix:** Open a console and execute `config.sh` in the `bin` sub-directory of the JIRA Installation Directory. This may fail with the error as described in our Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error KB article. Please refer to it for the workaround.

2. Navigate to the **Database** tab and set **Database type** to **HSQL**.

3. Test your connection and save. The JIRA configuration tool will configure your bundled HSQL database automatically. This tool adds the following elements to the `dbconfig.xml`, which are normally required when running JIRA with HSQLDB:

   ```xml
   <min-evictable-idle-time-millis>4000</min-evictable-idle-time-millis>
   <time-between-eviction-runs-millis>5000</time-between-eviction-runs-millis>
   ``

4. Restart JIRA.

**Manually**

1. Locate 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 (shown below) into this file.

2. Update the file, as described in the **Database connection fields** section below. Escape any `'` characters by adding `'` to the end of each one.
   - **Note:** the `<database-type/>` element must specify your type of database, e.g. `hsqldb`. If
you forget to do this and you start JIRA, your database tables may be created incorrectly. See Incorrect database type specified if this happens to you.

3. Save the file and restart JIRA.

### Database connection fields

<table>
<thead>
<tr>
<th>Field name</th>
<th>dbconfig.xml</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hostname</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below): &lt;url&gt;jdbc:hsqldb:C:\Data\JIRA 5.2.7/database/jiradb&lt;/url&gt;</td>
<td>The name or IP address of the machine that the HSQL server is installed on.</td>
</tr>
<tr>
<td><strong>Database</strong></td>
<td>Located in the <code>&lt;url&gt;</code> tag (bold text in example below): &lt;url&gt;jdbc:hsqldb:C:\Data\JIRA 5.2.7/database/jiradb&lt;/url&gt;</td>
<td>The name of your HSQL database (into which JIRA will save its data).</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>Located in the <code>&lt;username&gt;</code> tag (see bold text in example below): &lt;username&gt;sa&lt;/username&gt;</td>
<td>The user that JIRA uses to connect to the HSQL server. You should have created this in Step 1 above.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>Located in the <code>&lt;password&gt;</code> tag (see bold text in example below): &lt;password&gt;&lt;/password&gt;</td>
<td>The user's password — used to authenticate with the HSQL server.</td>
</tr>
<tr>
<td><strong>Schema</strong></td>
<td>Located in the <code>&lt;schema-name&gt;</code> tag (see bold text in example below): <code>&lt;schema-name&gt;PUBLIC&lt;/schema-name&gt;</code></td>
<td>The name of the schema that your HSQL database uses.</td>
</tr>
</tbody>
</table>

### Sample dbconfig.xml file

For more information about the child elements of `<jdbc-datasource/>` beginning with pool in the dbconfig.xml file above, see Tuning Database Connections.
<?xml version="1.0" encoding="UTF-8"?>

<jira-database-config>
  <name>defaultDS</name>
  <delegator-name>default</delegator-name>
  <database-type>hsq1</database-type>
  <schema-name>PUBLIC</schema-name>
  <jdbc-datasource>
    <url>jdbc:hsqldb:C:\Data\JIRA 5.2.7/database/jiradb</url>
    <driver-class>org.hsqldb.jdbcDriver</driver-class>
    <username>sa</username>
    <password></password>
    <pool-min-size>20</pool-min-size>
    <pool-max-size>20</pool-max-size>
    <pool-max-wait>30000</pool-max-wait>
    <min-evictable-idle-time-millis>4000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>5000</time-between-eviction-runs-millis>
    <pool-max-idle>20</pool-max-idle>
    <pool-remove-abandoned>true</pool-remove-abandoned>
    <pool-remove-abandoned-timeout>300</pool-remove-abandoned-timeout>
  </jdbc-datasource>
</jira-database-config>

4. Start JIRA

You should now have JIRA configured to connect to your HSQL 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 HSQL.

Tuning Database Connections

JIRA uses a database connection pool, based on Apache Commons DBCP (DataBase Connection Pool), to manage JIRA's access to its underlying database.

In earlier JIRA versions, the database connection pool was handled purely through the Apache Tomcat application server running JIRA. However, from JIRA version 4.4, JIRA's dbconfig.xml file provides a set of database connection pool settings to Tomcat, which in turn are used by Tomcat to manage JIRA's database connection pool. From JIRA version 5.1, the number database connection pool settings defined in JIRA's dbconfig.xml file substantially increased.

The information on this page can help you tweak JIRA's database connection pool settings. You can do this by using the JIRA Configuration Tool or by directly editing JIRA's dbconfig.xml file, as described below.

The Advanced tab of the JIRA Configuration Tool makes it easier to both configure and control JIRA's database connection pool. The Database Monitoring page (accessible to JIRA system administrators) provides a visual tool for monitoring JIRA's database connection usage.

On this page:
- Connection pool architecture
- Tuning JIRA's database connections
  - Connection pool settings
  - Monitoring the connection pool

Connection pool architecture

Whenever JIRA needs to access (i.e. read from or write to) its database, a database connection is required.
A database connection is a large and complex object that handles all communication between JIRA and its database. As such, database connections are time consuming to establish and consume a significant amount of memory on both the client (the JIRA application) and database server.

To avoid the impact of creating a new database connection for each database access request made by JIRA, a pool of pre-established database connections is maintained. Each new database access request made by JIRA uses a connection from this pool of pre-established connections, as required. Hence:

1. When JIRA starts up, a minimum number of database connections are established in the pool between JIRA and its database.
2. When JIRA needs to access its database, JIRA:
   a. requests a database connection from the pool
   b. uses this database connection to read from and/or write to its database
   c. returns the database connection to the pool when finished.

If the frequency of JIRA's database access requests begin to exceed the number of available database connections in the pool, extra connections are automatically created to handle the load.

Conversely, if the frequency of JIRA's database access requests begin to drop below the number of available database connections in the pool, connections can be automatically closed to release resources back to the system.

Modern databases can handle large numbers of connections relatively easily and with sufficient memory, many hundred. On the client side, however, these connections can consume a significant amount memory. Hence, it is generally best to limit the number of connections to a much smaller number while having a sufficient number for the application to rarely need to wait for a connection when it needs one.

**Tuning JIRA’s database connections**

**To tune JIRA’s database connections:**

1. Shut down your JIRA installation.
2. Do either of the following:
   * If you are using a 'Recommended' distribution of JIRA, use the JIRA Configuration Tool to tune JIRA’s database connections.
     a. Start the JIRA Configuration Tool:
        * **Windows:** Open a command prompt and run `config.bat` in the `bin` sub-directory of the JIRA Installation Directory.
        * **Linux/Unix:** Open a console and execute `config.sh` in the `bin` sub-directory of the JIRA Installation Directory.

       This may fail with the error as described in our [Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error](https://kb.atlassian.com/ent/188) KB article. Please refer to it for the workaround.

       **Please Note:** You may need to set the `JAVA_HOME` environment variable to run the JIRA Configuration Tool. See [Installing Java](https://www.atlassian.com/software/jira/docs/atlassian-jira/installing) for details.

     b. Once the JIRA Configuration Tool is running, click its **Advanced** tab.
c. Refer to **Connection pool settings** below for more information about the options on this tab. To specify a value for one of these options, ensure its leftmost check box has been selected first.

Some options above are simple check boxes (i.e. in the centre of the JIRA Configuration Tool). Selecting these check boxes sets the values of their associated options to 'true'. Conversely, clearing these check boxes sets the values of their associated options to 'false'.

d. Click the **Save** button to save your changes, which will be stored as elements in your `dbconfig.xml` file.

- Alternatively, edit the `dbconfig.xml` file at the root of your JIRA **Home Directory**.

  a. Refer to **Connection pool settings** below for more information about the elements you can add to your `dbconfig.xml` file to fine tune JIRA's database connection.

  b. Save your edited `dbconfig.xml` file.

3. Restart your JIRA installation.

### Connection pool settings

<table>
<thead>
<tr>
<th>JIRA Configuration Tool 'Advanced' tab option</th>
<th>Element in <code>dbconfig.xml</code></th>
<th>Explanation</th>
<th>Recommendations / Notes</th>
<th>Default value*</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum Size</strong></td>
<td><strong>pool-max-size</strong></td>
<td>The maximum number of database connections that can be opened at any time.</td>
<td>This value should be sufficiently large enough that JIRA rarely needs to wait for a database connection to become available when JIRA requires one. See Monitoring below for suggestions on how to set this parameter.</td>
<td>20</td>
</tr>
<tr>
<td><strong>Maximum Idle</strong></td>
<td><strong>pool-max-idle</strong></td>
<td>The maximum number of database connections that are allowed to remain idle in the pool.</td>
<td>Specifying a negative number sets no limit on the number of database connections that can remain idle. If the value of <strong>Minimum Idle/Size</strong> (below) is the same as that of <strong>Maximum Size</strong> (above), which is the case by default, then this setting has no effect.</td>
<td>Value of <strong>Maximum Size</strong></td>
</tr>
<tr>
<td><strong>Minimum Idle/Size</strong></td>
<td><strong>pool-min-size</strong> (&lt;em&gt;min-idle&lt;/em&gt;)</td>
<td>The minimum number of idle database connections that are kept open at any time.</td>
<td>Having this value set to that of <strong>Maximum Size</strong> (above), which is the case by default, means the pool will always have a fixed number of connections and idle connections will never be closed. On very large JIRA installations, there may be some benefit in specifying a lower value for this setting than that of <strong>Maximum Size</strong>, to conserve resources.</td>
<td>Value of <strong>Maximum Size</strong></td>
</tr>
<tr>
<td><strong>Initial Size</strong></td>
<td><strong>pool-initia-s</strong></td>
<td>The initial number of database connections opened in the pool.</td>
<td>This setting is not usually configured (other than the default value of 0), since a number of database connections are quickly created when JIRA starts up.</td>
<td>0 (when not specified in dbconfig.xml)</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
<td>---------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td><strong>Maximum Wait Time</strong></td>
<td><strong>pool-max-wait</strong></td>
<td>The length of time (in milliseconds) that JIRA is allowed to wait for a database connection to become available (whilst there are no free ones available in the pool), before returning an error.</td>
<td>Specifying a value of <code>-1</code> means that Tomcat will wait indefinitely. You should specify a time here which is long enough to allow for any contention spikes, but short enough that users will receive a meaningful error rather than just getting no response or a browser time out.</td>
<td>30000</td>
</tr>
</tbody>
</table>

**Advanced settings**

Generally, changing the settings below are not usually required. Refer to the [Apache DBCP documentation](http://dbcp.sourceforge.net/) if required.

<table>
<thead>
<tr>
<th><strong>Pool Statements</strong></th>
<th><strong>pool-prepared-statements</strong></th>
<th>Enable the pooling of prepared statements for the database connection pool.</th>
<th>Prepared statements allow the pre-compilation of commonly used SQL statements, which can dramatically improve performance if the statements are executed multiple times.</th>
<th>false (when not specified in dbconfig.xml)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum Open Statements</strong></td>
<td><strong>max-open-prepared-statements</strong></td>
<td>The maximum number of open statements that can be allocated from the statement pool at the same time.</td>
<td>Specify zero for no limit.</td>
<td>0 (when not specified in dbconfig.xml)</td>
</tr>
</tbody>
</table>
### Validation Query

**validation-query**

The SQL query that will be used to validate connections from this pool. If specified, this query MUST be an SQL SELECT statement that returns at least one row.

See [Surviving Connection Closures](dbconfig.xml) for more information.

**select 1** (for MySQL)

(otherwise, not specified in `dbconfig.xml`)

### Validation Query Timeout

**validation-query-timeout**

The length of time (in seconds) that the system should wait for a validation query to succeed before it considers the database connection broken.

The length of time should be quite short as the validation query should be designed to do a minimum amount of work.

If you specify a **Validation Query** above, then you should specify a value for the **Validation Query Timeout** too. If not, a value of '-1' is assumed, which results in the system waiting indefinitely until a validation query succeeds against a broken database connection, which it never will.

3 (for MySQL)

(otherwise, not specified in `dbconfig.xml`)

### Test On Borrow

**pool-test-on-borrow**

Tests if the database connection is valid when it is borrowed from the database connection pool by JIRA.

If the database connection is broken, it is removed from the pool.

This value should always be 'false' as JIRA borrows a connection for each database operation.

✅ If you continue to have problems with database connections closing, try setting this option to 'true'. However, this should only be used as a last resort and only in the event that decreasing the value of **Time Between Eviction Runs** has not reduced or prevented problems with database connections closing.

false (when not specified in `dbconfig.xml`)
<table>
<thead>
<tr>
<th>Feature</th>
<th>Configuration Name</th>
<th>Description</th>
<th>Default Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test On Return</td>
<td>pool-test-on-return</td>
<td>Tests if the database connection is valid when it is returned to the database connection pool by JIRA. If the database connection is broken, it is removed from the pool.</td>
<td>false (when not specified in dbconfig.xml)</td>
<td></td>
</tr>
<tr>
<td>Test While Idle</td>
<td>pool-test-while-idle</td>
<td>Periodically tests if the database connection is valid when it is idle. If the database connection is broken, it is removed from the pool.</td>
<td>true (for MySQL) false (when not specified in dbconfig.xml)</td>
<td>The Test While Idle only needs to be specified if you have specified a Validation Query above.</td>
</tr>
<tr>
<td>Time Between Eviction Runs</td>
<td>time-between-eviction-runs-millis</td>
<td>The number of milliseconds to sleep between runs of the idle object eviction thread. When non-positive, no idle object eviction thread will be run. The eviction thread will remove idle database connections when the number of idle connections exceeds Minimum Idle/Size (above).</td>
<td>300000 (for MySQL) 5000 (for HSQLDB) (otherwise, not specified in dbconfig.xml)</td>
<td>If you continue to have problems with database connections closing, try setting this option to a lower value.</td>
</tr>
</tbody>
</table>
Minimum Evictable Idle Time

| Minimum Evictable Idle Time | min-evictable-idle-time-millis | The minimum amount of time an object may sit idle in the database connection pool before it is eligible for eviction by the idle object eviction (if any). | 60000 (for MySQL)
4000 (for HSQLDB)
(otherwise, not specified in dbconfig.xml) |

Remove Abandoned

| Remove Abandoned | pool-remove-abandoned | Flag to remove abandoned database connections if they exceed the Removed Abandoned Timeout (below). If an internal failure occurs, it is possible that JIRA may borrow a connection and never return it. If this happens too often, then the pool may run short of database connections, causing JIRA’s performance to degrade or JIRA to fail altogether. | This value should be set to ‘true’. This will allow the pool to recover any abandoned connections and prevent this effecting system performance. | true |

Remove Abandoned Timeout

| Remove Abandoned Timeout | pool-remove-abandoned-timeout | The length of time (in seconds) that a database connection can be idle before it is considered abandoned. | 300 |

*Please Note:*

- JIRA writes elements with their default values (in the right-hand column of the table above) to the dbconfig.xml file after:
  - You have run through the JIRA Setup Wizard or
  - You use the Advanced tab of the JIRA Configuration Tool to configure/tune your database connection — even when the leftmost check boxes of options associated with these elements have not been selected.
  - The exception to this are elements whose values have ‘when not specified in dbconfig.xml’ indicated below them. These elements are:
    - Not written to the dbconfig.xml file after running through the JIRA Setup Wizard.
    - Only written to the dbconfig.xml file by:
      - Manually writing them into this file.
      - Using the Advanced tab of the JIRA Configuration Tool, selecting the leftmost check boxes of the options associated with these elements and specifying values for these options.
    - When 'when not specified in dbconfig.xml' is indicated below a default value in the right-hand column of the table above, then this default value is assumed, even when it is not present in the dbconfig.xml file.

*Monitoring the connection pool*
JIRA provides a view of its database connection usage via the 'Database Monitoring' page. See Monitoring Database Connection Usage for more information.

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, database connections in the database connection pool can be validated by running a simple SQL query. If a broken database connection is detected in the pool, 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 database connection pool. See the following procedure for details.

**Ensuring JIRA validates connections to its database**

To ensure JIRA validates database connections in the database 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 or use the Advanced tab of the JIRA Configuration Tool to configure the relevant settings.
3. Configure the validation query for your type of database:
   - If editing the dbconfig.xml file, add the <validation-query/> element with the appropriate validation query for your type of database, as shown in the example below for MySQL. (See Detailing 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>
  <jdbc-datasource>
    <url>jdbc:mysql://dbserver:3306/jiradb?useUnicode=true&amp;characterEncoding=UTF8&amp;sessionVariables=storage_engine=InnoDB</url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <username>jiradbuser</username>
    <password>password</password>
    <pool-min-size>20</pool-min-size>
    <pool-max-size>20</pool-max-size>
    <pool-max-wait>30000</pool-max-wait>
    <validation-query>select 1</validation-query>
    <min-evictable-idle-time-millis>60000</min-evictable-idle-time-millis>
    <time-between-eviction-runs-millis>300000</time-between-eviction-runs-millis>
    <pool-max-idle>20</pool-max-idle>
    <pool-remove-abandoned>true</pool-remove-abandoned>
    <pool-remove-abandoned-timeout>300</pool-remove-abandoned-timeout>
    <pool-test-while-idle>true</pool-test-while-idle>
    <validation-query-timeout>3</validation-query-timeout>
  </jdbc-datasource>
</jira-database-config>
```
3. If using the JIRA Configuration Tool, on the Advanced tab, select the Validation Query check box and enter the appropriate validation query for your type of database. (See Determining the Validation Query below for details.)

4. Specify a validation query timeout for your validation query, whose value is the appropriate length of time (in seconds) that the system should wait for a validation query to succeed before the system considers the database connection broken:
   - If editing the dbconfig.xml file, add the `<validation-query-timeout/>` element with the appropriate length of time (in seconds). This should only be done for MySQL.
   - If using the JIRA Configuration Tool, on the Advanced tab, select the Validation Query Timeout check box and enter the appropriate length of time (in seconds).

5. You may wish to specify the following options, which relate to the above validation query options (see Tuning Database Connections - Connection pool settings section for details):

<table>
<thead>
<tr>
<th>JIRA Configuration Tool 'Advanced' tab option</th>
<th>Element in dbconfig.xml</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test While Idle</td>
<td>pool-test-while-idle</td>
</tr>
<tr>
<td>Time Between Eviction Runs</td>
<td>time-between-eviction-runs-millis</td>
</tr>
<tr>
<td>Minimum Evictable Idle Time</td>
<td>min-evictable-idle-time-millis</td>
</tr>
</tbody>
</table>

6. Save your edited dbconfig.xml file (or click the Save button if using the JIRA Configuration Tool).
7. Restart JIRA (or the Tomcat installation running JIRA).

**Please Note:** If you continue to have problems with connections closing, you may need to set the `time-between-eviction-runs-millis` parameter to a lower value or as a last resort, set `test-on-borrow` to `true`. For more information about `test-on-borrow`, see Tuning Database Connections - Connection pool settings section.

**Determining the Validation Query and Timeout**

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 validation query timeout should only be set for MySQL.

The following validation queries are recommended for the following types of databases:

<table>
<thead>
<tr>
<th>Database Type</th>
<th>Validation Query</th>
<th>Validation Query Timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL</td>
<td><code>select 1</code></td>
<td>3</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td><code>select 1</code></td>
<td>N/A</td>
</tr>
<tr>
<td>Oracle</td>
<td><code>select 1 from dual</code></td>
<td>N/A</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td><code>select version();</code></td>
<td>N/A</td>
</tr>
</tbody>
</table>

⚠️ If the Validation Query Timeout is used on any database other than MySQL it will cause significant problems with the JIRA instance.

**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.

**Switching Databases**

JIRA’s data can be migrated from one database to:
1. A different database on the same database server,
2. The same database type on a different server (e.g. from one PostgreSQL server to another PostgreSQL server) or
3. 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** (covers scenarios 1 and 2 above)
- **Migrating JIRA's data to a different type of database server** (covers scenario 3 above)

### 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 data 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 Database** section 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).

You can also use this procedure if your JIRA installation is currently using the internal HSQL database (which is only supported for evaluating JIRA) and you need to switch your JIRA installation across to using a **supported database** (which are supported for JIRA installations used in a production environment).

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.
6. Configure JIRA’s connection to your new database (created in step 2 above) and click the ‘Next’ button.
7. 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.

Upgrading JIRA

There are several different ways to upgrade JIRA, and the method you choose to use depends on which version of JIRA you use and the type of environment you use it in. Use this table to determine which steps to follow to complete your JIRA upgrade:

<table>
<thead>
<tr>
<th>Required uptime (SLA)</th>
<th>Hardware/Software Change</th>
<th>Operating system</th>
<th>JIRA package</th>
<th>Current JIRA version</th>
<th>Upgrade process</th>
</tr>
</thead>
<tbody>
<tr>
<td>High / Mission Critical</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Upgrading JIRA with a Fallback Method</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low – Medium</td>
<td>Neither operation system, database or home directory will be changed.</td>
<td>MS Windows / Linux</td>
<td>Standalone</td>
<td>4.3.0 or later</td>
<td>Upgrading JIRA Using a Rapid Upgrade Method</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.0.0 – 4.2.x</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Upgrading JIRA Manually</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>earlier than 4.0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Upgrading JIRA with a Fallback Method</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WAR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.0.0 or later</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Upgrading JIRA Manually</td>
</tr>
</tbody>
</table>
If you are upgrading from JIRA 4.3.0 or earlier, you also have the option to use the upgrade capabilities built into the installer to perform a rapid upgrade of your existing JIRA installation.

<table>
<thead>
<tr>
<th>Solaris</th>
<th>any</th>
<th>earlier than 4.0.0</th>
<th>Upgrading JIRA with a Fallback Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4.0.0 or later</td>
<td>Upgrading JIRA Manually</td>
<td></td>
</tr>
</tbody>
</table>

If you plan to skip multiple major versions of JIRA when you upgrade, please review the Skipping Major Versions when Upgrading JIRA for important information on the recommended way to skip versions.

Upgrading JIRA Manually

This page describes how to upgrade JIRA installations that don't support the rapid upgrade method or fallback method. You should use this method to upgrade JIRA if you meet any of the following criteria:

- You use a WAR distribution of JIRA version 4.0.0 or later.
- You use JIRA 4.0.0 or later on Solaris.
- You use JIRA 4.0.0 – 4.2.x on Windows or Linux.

See Upgrading JIRA for more information on the methods you can use to upgrade JIRA.

On this page:
- 1. Before you start
- 2. Backing up
- 3. Setting up your new JIRA installation
- 4. Post upgrade checks and tasks

1. Before you start

- **Read about the new version** - Review the release notes and upgrade notes for the version of JIRA that you are upgrading to. See Production Releases. If you plan to skip a few JIRA versions during your 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.
- **Check your license** - Verify that your license support period is still valid.
- **Check for known issues** - Use the JIRA Knowledge Base to search for any issues in the new version that will affect you.
- **Check for compatibility**:
  - Confirm that your operating system, database, other applicable platforms and hardware still comply with the requirements for JIRA 6.1. The End of Support Announcements for JIRA page also has important information regarding platform support for future versions of JIRA.
  - If you have installed JIRA plugins (i.e. not included with JIRA), 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 plugin's home page on the Atlassian Plugin Exchange.
  - 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 before proceeding with the JIRA upgrade.

**Warning**: 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:
2. Backing up

Before you begin the JIRA upgrade, we strongly recommend that you back up your existing JIRA installation.

2.1 Stop users from updating JIRA data

During the upgrade process, you’ll export JIRA’s database from your existing JIRA installation (via an XML backup) and then restore this backup into a new JIRA installation. To ensure that the data in the XML backup is consistent with the latest data in the system, you must temporarily restrict access to JIRA so users can’t update the data. Refer to the Preventing users from accessing JIRA during backups page for more information.

⚠️ 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 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 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

1. Shut down JIRA.
2. Locate the JIRA Home directory. You can find information about the location of the directory by navigating to the `<jira-application-dir>/WEB-INF/classes/jira-application.properties` file in your JIRA Installation Directory. Alternatively, you can open the JIRA Configuration Tool to see the directory that is set as your JIRA Home.
3. Navigate to the directory specified in the configuration file and create a backup of it in another directory.

2.4 Back up your attachments and index directories if located outside your JIRA Home directory

If the attachments and index directories are located outside of your JIRA Home Directory, you must back them up separately. These pages describe how to find out where these directories are located in your implementation:

- 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.

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 in production.

3.1 Install the new version of JIRA

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** (just Step one), or
- **Installing JIRA WAR** (Steps one through seven)

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 6.1 installation is on a new server, **copy the backup of your existing JIRA Home Directory** from the old server to the new server before proceeding.

**To set up a "recommended" (not WAR) distribution:**

1. Open the **JIRA Configuration Tool**.
2. Click the **JIRA Home** tab.
3. Update the **JIRA Home Directory** field:
   - If your JIRA 6.1 installation is on a new server, update the **JIRA Home Directory** field to the path of your **copied** JIRA Home directory.
   - If your JIRA 6.1 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**.

**To set up a WAR distribution:**

1. Edit the **jira-application.properties** file located within the `<jira-application-dir>/WEB-INF/classes` subdirectory of your new JIRA 6.1 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 6.1 installation is on a new server, update the **jira.home** property to the path of your **copied** JIRA Home directory.
   - If your JIRA 6.1 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 the **jira.home** 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 precedence 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

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.

ℹ️ 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

If you have modified properties in configuration files of your existing JIRA installation, make the same modifications in your new JIRA installation. However, because the properties in the configuration files may have changed between versions, you cannot simply copy the configuration files from your existing installation and replace the equivalent files in the new installation.

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:

<table>
<thead>
<tr>
<th>File</th>
<th>Location in 'recommended' (formerly 'Standalone') JIRA distributions</th>
<th>Location in JIRA WAR</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira-application.properties</td>
<td>atlassian-jira/WEB-INF/classes</td>
<td>webapp/WEB-INF/classes</td>
<td>Location of the JIRA Home Directory and Advanced JIRA Configuration in JIRA 4.3.x and earlier. Any custom property values defined in the jira-application.properties file of your existing JIRA 4.3.x (or earlier) installation must be migrated across to the jira-application.properties file of your new JIRA 6.1 installation before you start your new JIRA installation. Upon starting your new JIRA installation, any custom property values in the jira-application.properties file will automatically be migrated across to either the JIRA database or jira-config.properties file. jira.home is the only property of the jira-application.properties file subsequently used by JIRA.</td>
</tr>
<tr>
<td>setenv.bat (Windows) or setenv.sh (Linux)</td>
<td>bin</td>
<td>Application server's bin directory</td>
<td>Increasing JIRA Memory</td>
</tr>
</tbody>
</table>
### The version-specific upgrade notes

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.
  - Remember to configure Crowd to grant JIRA’s new hostname/IP access: Specifying an Application’s Address or Hostname
- **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](http://plugins.atlassian.com) site.
- **Character encoding** — Ensure that character encoding (i.e. locale) is the same on the new and old locations. Your new version of JIRA may not function correctly if attachments are moved between two systems with incompatible encoding.
- **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) 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.
3.6 Import your old JIRA data into your new JIRA

After you have started your new JIRA installation, 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. 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, specify the XML backup file you created previously during the export process (above). That zipped file should contain two xml files: activeobjects.xml and entities.xml. Both of these files must be included in the zipped file for the import process to work.
4. Restore the attachments directory that you backed up previously, into the attachments directory of your new JIRA. (See Restoring Data.)
   🔴 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).
6. Once you have confirmed that the new server is working correctly, ensure that the production license is updated for the new server ID, as follows:
   b. Locate the appropriate license.
   c. Edit the Server ID, as per the new production Server ID, and save it.
   d. Update the production license in the new server.

Congratulations! You have completed your JIRA migration/upgrade.

See Also

Disabling Auto-Export
Restoring Data
Upgrading JIRA with a Fallback Method

This page describes how to upgrade JIRA 4.4.x or later in a way that allows you to safely roll back to your previous system if the upgrade process takes longer than expected or if you encounter issues. This method is especially useful for enterprise environments and for organisations where JIRA is mission-critical for the business. You can also use this method so you have a fallback option if you are performing a complex system change, such as changing the operating system that will run JIRA, the database that will store JIRA's data or the location of JIRA's index and/or attachments paths.

Because this process is designed to provide the safest possible upgrade method, it requires advanced knowledge of database administration tasks. We recommend you have the following resources and/or skill sets available for your upgrade:

- **Database Administrator** - for general production-level database administration *(i.e. run backups, create, remove, restore, etc.)*
- **JIRA Application Administrator** - for general application administration and upgrade management *(i.e. JIRA SME, user with System Administrator privileges and deep understanding of application and associated dependencies within your organization.)*
- **Systems/Network Administrator** - for managing systems and networks *(i.e. proxy servers, DNS changes, monitoring, VM's, hardware, etc.)*

This upgrade process also requires you to make backups of your database, which can be time-consuming. Customers with large JIRA environments should plan for four hours of downtime. If you know your system takes several hours to re-index, you might want to allocate more than four hours for the upgrade.

See [Upgrading JIRA](#) for more information on the methods you can use to upgrade JIRA.
Before You Start

- **Read about the new version** - Review the release notes and upgrade notes for the version of JIRA that you are upgrading to. See Production Releases. If you plan to skip a few JIRA versions during your 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.
- **Check your license** - Verify that your license support period is still valid.
- **Check for known issues** - Use the JIRA Knowledge Base to search for any issues in the new version that will affect you.
- **Check for compatibility**:  
  - Confirm that your operating system, database, other applicable platforms and hardware still comply with the requirements for JIRA 6.1. The End of Support Announcements for JIRA page also has important information regarding platform support for future versions of JIRA.
  - If you have installed JIRA plugins (i.e. not included with JIRA), 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 plugin's home page on the Atlassian Plugin Exchange.
  - 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 before proceeding with the JIRA upgrade.

If you have any problems during the upgrade process, create an issue at our support site so that we can help you resolve the problems with your upgrade. We strongly recommend that you perform the below procedure first as a test only. This will allow you to note any complications (e.g. with customized settings or add-ons) ahead of time so that you can minimize the downtime of the system.

1. Prepare Your Production Instance for Upgrade

*Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](https://creativecommons.org/licenses/by/2.5/au/).*
When you begin preparing to upgrade, it’s best practice to halt any major changes to your production system (such as plugin upgrades, customisations, etc.). Keeping your production system as stable as possible will make testing the upgrade version simpler.

It’s also a good idea to let your users know about planned downtime, either through email or by using JIRA’s announcement banner.

2. Set Up a Proxy Server

Before beginning the upgrade process set up a reverse proxy, such as a load balancer. The proxy server allows you to redirect users to a different JIRA server without having to wait for a DNS change - this change will be invisible to the end-user. If, at any point during the upgrade process, you encounter issues you can’t resolve and you need to rollback to your existing JIRA instance, simply restart your existing JIRA instance and reconfigure the proxy server to point to the old server.

If you use monitoring, API calls (such as SOAP, REST, or CLI), or scripts associated with your production server, update them with the new proxy information.

Please see the following documentation for further information on configuring Apache:

- Integrating JIRA with Apache
- Integrating JIRA with Apache using SSL

3. Pre-Stage and Test the New Version of JIRA

1. If you want to use a copy of your production data when you test the upgraded JIRA system, make a copy of your production database using your native database backup tools. See Backing Up Data. You can alternatively skip this step and use a new database for testing.
2. Install the version of JIRA you want to upgrade to onto a system you can use for testing (use either a test server or a separate directory on an existing system). This will become your new production system after you complete the upgrade process. Follow the instructions here to install a new version of JIRA: Installing JIRA.
3. Migrate any customisations you use in your production system. Follow the instructions in step 3.4 (Migrate your existing JIRA configurations over to your new JIRA installation) in the "Migrating JIRA to Another Server" page.
4. Connect the new version of JIRA to the copy of the production database (not the existing production database) or a new testing database. See Connecting JIRA to a Database.
5. Start the new version of JIRA. See this Knowledge Base article about how to test mail settings without accidentally sending notifications to users from the test system: How to Prepare a Development Server's Mail Configuration.
6. Install any plugins that you use in your existing production version of JIRA. Some plugins have different compatibility for different JIRA versions, so this step will ensure that your plugins are updated for this new JIRA version.
7. Re-index JIRA so the new plugin information is captured in the index.
8. Check out the features and functionality you use in the new version to understand how they behave and how any changes will impact your team. It can be very helpful to have a group of users look at the new system and carry out their usual tasks to make sure they won’t run into any issues when the new version is in production.

When you are ready to begin the process of migrating your production data to this new version, 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).

4. Disable the Old JIRA Production Instance and Start the New Instance

Before disabling your old JIRA production instance, identify the location of your attachments and index directories. If they are located outside of your JIRA Home Directory, you will back them up manually later during the upgrade process. These pages describe how to find out where these directories are located in your environment:

- Your attachments directory — Refer to the Configuring File Attachments page for your version of JIRA.
- Your index directory — Refer to the Search Indexing page for your version of JIRA.

If your attachments and index directories are located outside of the JIRA Home directory, note their location so you can easily find them later.
After you've located the attachments and index directories, disable the old JIRA production instance so you can perform a database backup:

1. Shut down your old production JIRA instance (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. Using your database's native backup tools, perform a backup of the data in your old production JIRA instance. See Backing Up Data.
3. Set the newest copy of the production database as the new database for production.

Make sure that the database set up for the new production version of JIRA is clearly distinguishable from the database backup of your old production JIRA, and that the new production instance is not configured to connect to the old production database.

4. Synchronise the JIRA attachment directories:
   a. Locate the JIRA Home directory. You can find information about the location of the directory by navigating to the `<jira-application-dir>/WEB-INF/classes/jira-application.properties` file in your JIRA Installation Directory. Alternatively, you can open the JIRA Configuration Tool to see the directory that is set as your JIRA Home.
   b. Navigate to the directory specified in the configuration file and create a backup of it in another directory.
   c. If the attachments and index directories are located outside of your JIRA Home Directory, you must back them up separately. (See the beginning of this task for information on how to find these files.)
   d. Replace the JIRA Home directory (and the attachment and index directories, if separate from the JIRA Home directory) in the new JIRA production environment with the backups you made of the old production directories.

5. Start the new version of JIRA in your new production environment. When you start this version, JIRA will upgrade your data and may perform a re-index. When the re-indexing is complete, verify that your data is present and that there are no issues with the system.

The re-indexing may take up to several hours, depending on the size of your instance. If you know that your instance takes a long time to index, make sure to plan your scheduled downtime accordingly.

6. Reconfigure the proxy server you set up in step 2 so that the new version of JIRA becomes your production instance. Make sure to let your users know about the new instance (including the new domain name) and any changes that might affect them.

If you experience any issues in the new production environment, you can simply revert the proxy server settings and re-instate your old production instance until you can resolve the issue.

9. 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).
6. Once you have confirmed that the new server is working correctly, ensure that the production license is
updated for the new server ID, as follows:

b. Locate the appropriate license.
c. Edit the Server ID, as per the new production Server ID, and save it.
d. Update the production license in the new server.

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. It is recommended that you re-index JIRA after upgrading your plugins.

Congratulations! You have completed your JIRA migration/upgrade.

Upgrading JIRA Using a Rapid Upgrade Method

This page describes how to upgrade JIRA 4.3.0 or later in the quickest way possible. This method can save you time since it does not require you to set up a separate test instance before you upgrade (that is, you upgrade JIRA "in-place"). However, it does assume that your JIRA instance is not mission critical and that users or the business won’t be negatively affected when JIRA is unavailable during the upgrade.

You should use this method to upgrade JIRA if you are upgrading from the recommended (not WAR) JIRA distribution 4.3.0 or later on Windows or Linux. See Upgrading JIRA for more information on the methods you can use to upgrade JIRA.

Before You Start

- Read about the new version - Review the release notes and upgrade notes for the version of JIRA that you are upgrading to. See Production Releases. If you plan to skip a few JIRA versions during your 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.
- Check your license - Verify that your license support period is still valid.
- Check for known issues - Use the JIRA Knowledge Base to search for any issues in the new version that will affect you.
- Check for compatibility:
  - Confirm that your operating system, database, other applicable platforms and hardware still comply with the requirements for JIRA 6.1. The End of Support Announcements for JIRA page also has important information regarding platform support for future versions of JIRA.
  - If you have installed JIRA plugins (i.e. not included with JIRA), 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 plugin's home page on the Atlassian Plugin Exchange.
  - 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 before proceeding with the JIRA upgrade.

1. Checking for Customizations

Using the rapid upgrade method allows the installer to automatically perform many of the upgrade tasks for you. However, if you have made customizations to your JIRA installation, you must migrate customized files manually to the upgraded installation. The installer checks for and migrates automatically:

- 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. Other configurations or
customizations in this file are not migrated.

- Custom values in your existing JIRA installation's `jira-application.properties` and `setenv.sh`/`setenv.bat` files.

  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`

During the upgrade process, the installer 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. If you have made customizations to your `seraph-config.xml` file or any other file customizations in your JIRA installation directory which are not handled by the upgrade wizard, you must migrate these to the upgraded JIRA installation manually.

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.
- The installer automatically backs up the Installation and Home directories of the existing JIRA installation. If your attachments and index files are located outside your JIRA Home Directory, you must manually back up these files. These pages describe how to find out where these directories are located in your environment:
  - Your attachments directory — Refer to the Configuring File Attachments page for your version of JIRA.
  - Your index directory — Refer to the Search Indexing page for your version of JIRA.

### 2. Backing Up Your External Database

After you launch the upgrade wizard, but before it begins the upgrade, it asks you to back up your external database. You can back up the database using your database's native backup tools, however, note the following:

- If your database's native backup tools support online backups (i.e. backups that 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 when it prompts you to back up the database.
  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.
  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-start 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 when it prompts you about the backup 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.

### 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](https://www.atlassian.com/software/jira/downloads).
2. Run the '.exe' file to start the upgrade wizard.
   - If a Windows 7 (or Vista) 'User Account Control' dialog box asks you 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 customizations 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 customizations, please make a note of these files as you will need to manually migrate their customizations (which are not mentioned in the view above) to your upgraded JIRA installation. One relatively common customization 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 backed up your database (in step 2: Backing Up Your External Database).
   c. After the 'Upgrade Check List' step, the 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. Start your new (upgraded) JIRA installation.
      - **Please Note:**
        - If you noted any files that contain customizations which must be migrated manually to your upgraded JIRA installation (above), then:
          1. Stop the upgraded JIRA installation.
          2. Migrate the customizations 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](https://www.atlassian.com/software/jira/downloads).
2. Open a Linux console and change directory (`cd`) to the '.bin' file's directory.
   - **Please Note:**
     - 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.
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 customizations 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 customizations, please make a note of these files as you will need to manually migrate their customizations (which are not mentioned in the over view above) to your upgraded JIRA installation. One relatively common customization 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. You may have already backed up your database (in step 2 **Backing Up Your External Database**).

   c. After the 'Upgrade Check List' step, the 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.

      ![Alert Symbol] If you noted any files that contain customizations which must be migrated manually to your upgraded JIRA installation (above), then:

      1. Stop the upgraded JIRA installation.
      2. Migrate the customizations 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!

### 4. Post Upgrade Checks and Tasks

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.

**Congratulations! You have completed your JIRA upgrade.**

**Skipping Major Versions When Upgrading JIRA**

To upgrade from early versions of JIRA (3.x or earlier) to newer versions, you must upgrade to JIRA 4.4.5 before upgrading to a later version.

Follow these steps to skip major versions as you upgrade JIRA:

1. **Prepare**: Read the upgrade guides for all the major versions between your current version and the version to which you are upgrading. You can read about important changes between versions in the **Important Version-Specific Upgrade Notes**.

2. **Upgrade to an interim version** (**JIRA 3.x or earlier only**): Upgrade to JIRA 4.4.5 following these **Upgrading JIRA with a Fallback Method** instructions.

3. **Upgrade to the new version**: Upgrade to the new version of JIRA following these appropriate instructions. Use the table on **Upgrading JIRA** page to determine which method is appropriate for your environment.
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. Note that any data changed since the last backup will not be present after rolling back.

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.
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).

Migrating JIRA to Another Server

This document describes how to migrate/upgrade to JIRA 6.1 on different server hardware or in a different server environment that entails one or more of the following:

- 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.

If you are upgrading to a newer version of JIRA during the migration, please see Upgrading JIRA for information on the pre-requisite tasks you need to complete before upgrading.

On this page:

- 1. Before You Start
- 2. Backing Up
  - 2.1 Stop users from updating 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

1. Before You Start
We strongly recommend performing your migration in a test environment first. Do not migration 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 which you cannot resolve, create an issue at our support site so that we can assist you.
- If you have any problems during the migration 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 your migration.

Some anti-virus or other Internet security tools may interfere with the migration 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 migration.

2. Backing Up

2.1 Stop users from updating JIRA data

During the upgrade process, you'll export JIRA's database from your existing JIRA installation (via an XML backup) and then restore this backup into a new JIRA installation. To ensure that the data in the XML backup is consistent with the latest data in the system, you must temporarily restrict access to JIRA so users can't update the data. Refer to the Preventing users from accessing JIRA during backups page for more information.

⚠️ 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 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 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

1. Shut down JIRA.
2. Locate the JIRA Home directory. You can find information about the location of the directory by navigating to the `<jira-application-dir>/WEB-INF/classes/jira-application.properties` file in your JIRA Installation Directory. Alternatively, you can open the JIRA Configuration Tool to see the directory that is set as your JIRA Home.
3. Navigate to the directory specified in the configuration file and create a backup of it in another directory.

2.4 Back up your attachments and index directories if located outside your JIRA Home directory

If the attachments and index directories are located outside of your JIRA Home Directory, you must back them up separately. These pages describe how to find out where these directories are located in your implementation:
1. Your attachments directory — Refer to Configuring File Attachments page in the documentation for your version of JIRA.
2. Your index directory — Refer to Search Indexing page in the documentation for your version of JIRA.

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 JIRA, either the current version or a newer one. If you are upgrading JIRA during this process, please see Upgrading JIRA for information on the pre-requisite tasks you need to complete before upgrading.

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 section.

3.2 Point your new JIRA to (a copy of) your existing JIRA Home directory

If your new JIRA 6.1 installation is on a new server, copy the backup of your existing JIRA Home Directory from the old server to the new server before proceeding.

To set up a "recommended" (not WAR) distribution:

1. Open the JIRA Configuration Tool.
2. Click the JIRA Home tab.
3. Update the JIRA Home Directory field:
   - If your JIRA 6.1 installation is on a new server, update the JIRA Home Directory field to the path of your copied JIRA Home directory.
   - If your JIRA 6.1 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.

To set up a WAR distribution:

1. Edit the jira-application.properties file located within the <jira-application-dir>/WEB-INF/classes subdirectory of your new JIRA 6.1 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 6.1 installation is on a new server, update the jira.home property to the path of your copied JIRA Home directory.
   - If your JIRA 6.1 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 the jira.home 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 precedence over the value of the jira.home property in the jira-home.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

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

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

If you have modified properties in configuration files of your existing JIRA installation, make the same modifications in your new JIRA installation. However, because the properties in the configuration files may have changed between versions, you cannot simply copy the configuration files from your existing installation and replace the equivalent files in the new installation.

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:

<table>
<thead>
<tr>
<th>File</th>
<th>Location in 'recommended' (formerly 'Standalone') JIRA distributions</th>
<th>Location in JIRA WAR</th>
<th>Description</th>
</tr>
</thead>
</table>

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.
<table>
<thead>
<tr>
<th>File Name</th>
<th>Location</th>
<th>Detailed Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira-application.properties</td>
<td><code>atlassian-jira/WEB-INF/classes</code> <code>webapp/WEB-INF/classes</code></td>
<td>Location of the JIRA Home Directory and Advanced JIRA Configuration in JIRA 4.3.x and earlier. Any custom property values defined in the <code>jira-application.properties</code> file of your existing JIRA 4.3.x (or earlier) installation must be migrated across to the <code>jira-application.properties</code> file of your new JIRA 6.1 installation before you start your new JIRA installation. Upon starting your new JIRA installation, any custom property values in the <code>jira-application.properties</code> file will automatically be migrated across to either the JIRA database or <code>jira-config.properties</code> file. <code>jira.home</code> is the only property of the <code>jira-application.properties</code> file subsequently used by JIRA.</td>
</tr>
<tr>
<td>setenv.bat (Windows)</td>
<td><code>bin</code></td>
<td>Application server’s bin directory</td>
</tr>
<tr>
<td>or setenv.sh (Linux)</td>
<td></td>
<td>Increasing JIRA Memory</td>
</tr>
<tr>
<td>osuser.xml</td>
<td><code>atlassian-jira/WEB-INF/classes</code> <code>webapp/WEB-INF/classes</code></td>
<td>Modified if you have integrated LDAP with JIRA, integrated Crowd with JIRA, or if you are using a custom form of external user management or user authentication.</td>
</tr>
<tr>
<td>(not required if upgrading from JIRA 4.3.0 or later)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seraph-config.xml</td>
<td><code>atlassian-jira/WEB-INF/classes</code> <code>webapp/WEB-INF/classes</code></td>
<td>Modified if you have integrated Crowd with JIRA.</td>
</tr>
<tr>
<td>server.xml</td>
<td><code>conf</code></td>
<td>Application server’s conf directory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modified in the following situations:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If you had previously configured JIRA’s TCP ports differently from their defaults.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If you had implemented SSL.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When connecting JIRA to a database in JIRA 4.3.x and earlier.</td>
</tr>
</tbody>
</table>
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 6.1 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 started your new JIRA installation, 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. 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, specify the XML backup file you created previously during the export process (above). That zipped file should contain two xml files: activeobjects.xml and entities.xml. Both of these files must be included in the zipped file for the import process to work.
4. Restore the attachments directory that you backed up previously, into the attachments directory of your new JIRA. (See Restoring Data.)
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).

6. Once you have confirmed that the new server is working correctly, ensure that the production license is updated for the new server ID, as follows:
   b. Locate the appropriate license.
   c. Edit the Server ID, as per the new production Server ID, and save it.
   d. Update the production license in the new server.

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

Migrating from JIRA OnDemand to a JIRA Installed Site

This page is for people who are currently using a JIRA OnDemand site and wish to move to a JIRA site that is hosted on their own servers.

Summary

You will need to download and install the latest production release of JIRA (for example, 'JIRA 6.0.2') and then move your data from your hosted JIRA OnDemand site into your newly installed site.

On this page:
- Summary
- Before you begin
- Instructions
- Version matrix for imports

Before you begin

JIRA OnDemand is regularly updated with the absolute latest features and improvements — it is essentially running on a later version of JIRA than the latest downloadable version of JIRA. If you want to migrate from JIRA OnDemand to a site installed from the standard JIRA download, please be aware of the following information before you begin:
Feature loss

If you migrate from JIRA OnDemand to a site installed from the standard JIRA download, you will likely find a few features missing. This is because we have introduced features from the upcoming JIRA downloadable version into JIRA OnDemand. For example, the latest JIRA production release for download is JIRA 6.0.x. JIRA 6.1 is currently under development. Some of the JIRA 6.1 features have been made available in JIRA OnDemand, but will not become available for downloadable JIRA until the final JIRA 6.1 version is released.

JIRA license

Your Atlassian OnDemand license cannot be used in a site installed from the standard JIRA download. You will need to generate a new "JIRA" license from https://my.atlassian.com for your site installed from the standard JIRA download.

You can reuse your licenses for plugins in your site installed from the standard JIRA download. The licenses for Atlassian plugins and Gliffy for JIRA can be viewed on https://my.atlassian.com. You will need to contact your vendor for the licenses for all other third-party plugins.

Migrating other OnDemand applications

The instructions on this page only apply to JIRA. If you are migrating other OnDemand applications (e.g. Confluence OnDemand to a site installed from the standard Confluence download), please see this page: Backing up and exporting data.

Note, if you are migrating JIRA OnDemand and other applications (e.g. Confluence OnDemand) to a site hosted on your own servers, you will also lose a number of integration features that are native to OnDemand (e.g. Creating Links). These can be re-enabled by configuring application links between your applications. See Configuring Application Links and Quick Start: Application Links for instructions. Contact support if you need assistance.

Instructions

1. Generate a backup of your JIRA OnDemand data
2. Install JIRA from a standard download
3. Import your JIRA OnDemand data into your JIRA installation
4. Change the system administrator password
5. Check which plugins are installed on your JIRA OnDemand site
6. Install plugins (add-ons)

1. Generate a backup of your JIRA OnDemand data

1. Log in to your JIRA OnDemand site as an administrator.
2. Generate an XML export from your JIRA OnDemand data by following the instructions in Exporting issues. This includes instructions on how to back up your attachments. Note, the export process will strip your OnDemand license and plugin licenses out of the XML, they will not be available when importing into the installed site, but they will remain available in JIRA OnDemand.
3. Download the backup file from your OnDemand WebDAV directory (also described in Exporting issues).

2. Install JIRA from a standard download

You must use the latest version of downloadable JIRA.

2. Follow the instructions in Installing JIRA until you are instructed to run the setup wizard, then see step 4 below.

3. Import your JIRA OnDemand data into your JIRA installation

Complete the instructions in Running the Setup Wizard until you have configured a database (described in step 1 of Running the Setup Wizard). We recommend that you use PostgreSQL for your database, although you can use any of the supported databases. If you already have some data in your JIRA installation, this step will overwrite it.

In step 2 of the Setup Wizard (Application Properties), you will be asked whether you have existing data. Click import your existing data and follow the instructions to import the JIRA OnDemand backup that you generated earlier. Note, you may see the following message. This is a warning only, your import will still work.
1. Log in to your new JIRA site, using the following credentials:
   - Username: sysadmin
   - Password: sysadmin
2. Change the password immediately after logging in.

5. Check which plugins are installed on your JIRA OnDemand site

Any plugins that you are currently using with JIRA OnDemand will need to be installed in your JIRA installation. For example, JIRA Agile, Tempo, etc.

Choose 🕵️‍♂️ > Add-ons. The 'Find add-ons' screen shows add-ons available via the Atlassian Marketplace. Choose Manage Add-ons to view the plugins currently installed on your JIRA site. Choose Manage Add-ons and note the plugins listed under the User-installed Plugins section. You will need to note the plugin names and versions.

6. Install plugins (add-ons)

For each plugin that you noted in the previous step, install it in your JIRA installation. You must install a version of the plugin that is equal to or later than the plugin version that was installed JIRA OnDemand. Atlassian does not provide support for data that is downgraded as a result of installing an older version of a plugin.

See Managing Add-ons for instructions on how to install a plugin. You will need to manually add the plugin license keys.

Version matrix for imports

The following table tells you which version of downloadable JIRA to use, when migrating from JIRA OnDemand. The version number is dependent on when you exported your data from JIRA OnDemand.

We recommend that you use the latest JIRA version unless otherwise specified below. Only use the versions listed below if you cannot use the latest JIRA version.

<table>
<thead>
<tr>
<th>Date when export was made</th>
<th>Version of downloadable JIRA to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prior to 3 Dec 2012</td>
<td>Contact support for assistance</td>
</tr>
<tr>
<td>3 Dec 2012 — 16 Dec 2012</td>
<td>5.2.1</td>
</tr>
<tr>
<td>17 Dec 2012 — 20 Jan 2013</td>
<td>5.2.2</td>
</tr>
</tbody>
</table>

See Restoring Data for more information.

For instances with large backups (2Gigabyte and up), we recommend importing the attachments separate from the Issue and user data.

(Optional) If you exported your attachments you will need to add the saved directory into your new JIRA home path after the import process has completed.
Establishing Staging Server Environments for JIRA

This document describes best practices for an enterprise environment setup for JIRA:

- Best-practice recommendations for procedural governance around rolling out changes
- Recommendations for development / staging / production architecture
- Technical steps for how to deploy non-production servers

Assumptions:

- For this document we are assuming that as an administrator, you would rather script changes. Therefore we have omitted UI-based changes or separate tools such as the database configuration tool in favour of specifying file system locations.

On this page:

1. Architecture Strategy
2. Governance Strategy
3. How to Refresh a Staging Server
   - 3.1 Create a complete production backup
   - 3.2 Copy your complete production backup to a staging environment
   - 3.3 Modify your staging environment for the unique configurations
   - 3.4 Restart your Staging Server
   - 3.5 Post-Startup Modifications

Please Note:

- The procedures described in this document will work with JIRA version 4.0 and later.
- Please read the entire document before bringing a staging server live. There are risks associated with connecting to production instances that require attention, which are called out in the document.

1. Architecture Strategy

Often systems administration teams will have an established architecture for enterprise applications, including staging environments and failover setups. We offer these recommendations in this section not to supplant or change those company-wide strategies, but rather to help illustrate what some of the considerations will be with Atlassian products in staging environments.

Definitions

For the purpose of this document, we'll assume the following definitions:

- **Production**: your live instance, expecting minimal downtime and well tested changes.
- **Staging**: a pre-production environment, where the systems administration team can establish exact procedures prior to rollout.
- **Development**: a free-for-all environment where users can play with cutting-edge or risky changes.

Recommendation

If Atlassian products are critical systems, we recommend this 3-tier strategy for development, staging, and production.

- The staging environment is primarily for system administrators to test changes and upgrades before going into production.
- The development environment is for different business units to test changes on their own, before requesting a production rollout.

2. Governance Strategy
In addition to an architecture, we also recommend establishing a governance strategy for changes. This could include:

- Create a strategy for deploying and testing plugin installation requests. Note that some plugins that are extremely useful in some environments are not appropriate for high-volume critical systems.
- Publish a timeline for refreshing the development environment, so users know when to remove their changes.
- Set up a source control repository to house any file system changes, so you can track when changes were made and by whom, historically. If you don't have one already established, Bitbucket is an option. In addition to file system customizations, record your procedures for upgrades, staging refresh (see below) and any other scripted changesets in your source control.
  
  **Tip:** JIRA has a tool to manage any changes in your installation. Check the System Information page in the UI for "modified files." This will tell you which files have been customized in your installation directory.
- For changes such as creating new workflows (that require administrative access), you have two options:
  1. Create an administrative user which has temporary access to administrative functions, on a per-request basis. Add this user to the appropriate groups so they can perform the necessary administrative functions. When the user has completed their administrative functions, remove the user from these groups.
  2. Keep your development server devoid of production data and give more administrative privilege on this server. Require end-users to document specific workflow or scheme setups, then repeat these steps in production.

3. How to Refresh a Staging Server

We're assuming that you have an existing staging installation. If not, you can use these instructions to set up your staging environment now.

! Take care to make sure your staging server setup does not interfere with your production environment. Read the tips below before launching your staging (or development) server.

### 3.1 Create a complete production backup

1. **Back up your home directory.** See Setting your JIRA Home Directory for the location of your production home directory.  
   - Back up your production attachments and index directories if located outside your JIRA Home directory. If you're unsure where these are stored, refer to Configuring File Attachments and Search Indexing to determine these locations.  
   - Refer to Backing Up Data for more information about backing up attachments in JIRA.
2. **Back up your installation directory.** The 'JIRA Installation Directory' is the directory into which the JIRA application files and libraries were extracted when JIRA was installed.
3. **Back up your production database.** Use your native backup tools to take a snapshot of your production database.

### 3.2 Copy your complete production backup to a staging environment

1. **Shut down your staging server.**
2. **Restore your installation and home directories on the staging server.**
3. **Point the newly restored installation directory to the newly restored JIRA Home directory.**
   - Edit the jira-application.properties file located within the <jira-application-dir>/WEB-INF/classes subdirectory of your new Installation Directory JIRA Installation Directory.
   - Update the jira.home property in this file to the path of the new JIRA Home Directory to the path of your copied JIRA Home directory.
   - 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 precedence 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.
4. **Restore your database to a staging database.**
   - 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). Oracle does not support schema names with periods or underscores. Ensure the new database has identical access permissions to the old JIRA database.

### 3.3 Modify your staging environment for the unique configurations

1. Configure your database connection to point to your staging database. Edit the `dbconfig.xml` file at the root of your **JIRA Home Directory**, or the datasource in `<jira-install>/conf/server.xml` for older versions. **This is extremely important!** Make sure your staging environment is not pointing to your production database.

2. There are two options to handling email:
   a. **Disable mail on your staging server.** 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. You can leave emails on, if you're wanting to test email functionality. If you choose to do keep emails enabled, watch particularly for:
      i. Create or comment handlers, which can pull mail from your production mail servers. You can disable these from **Administration > Advanced > Services**, or delete them from the 'serviceconfig' table in the database.
      ii. Filter subscriptions, as your users will receive notifications for filters they’re subscribed to. Delete filter subscriptions from the 'filtersubscription' table in the database.
      iii. Notifications on tickets that are updated. For these, dissociate any notification schemes to projects you wish to test without email notifications.
   b. **Keep email enabled** and configure your staging instance to test email:
      i. See the guide here: [How to Prepare a Development Server's Mail Configuration](#)

### 3.4 Restart your Staging Server

You are now ready to restart your server. Once you've restarted, perform the following checks to verify you've done the above steps safely:

1. **Ensure the database is not pointing to production.** To check this, see **Viewing your System Information**. Check the 'Database URL' to ensure it's pointing to the right place.

2. **Ensure emails are disabled or configured for dev server.** Also when **Viewing your System Information**, check the 'JVM Input Arguments' for the line 'atlassian.mail.senddisabled'. If you configured the email for a dev server as described above, this line will not be there.

### 3.5 Post-Startup Modifications

1. **Modify the Site Colors.** See **Customizing the Look and Feel**. This is a good practice for users to identify that they're on the staging server.

2. **Modify the Site Base URL.** See **Configuring JIRA Options** and change the Site URL to the staging URL.

3. **Consider the URL Whitelist.** You may wish to change some of the approved URLs. See **Configuring the Whitelist.**

4. **Apply a Development License.** See our licensing FAQ to generate a license for the staging server. Refer to **Updating your JIRA License Details** to apply it.

5. **Reconfigure applinks.** If you are connecting to other servers via applinks, you'll need to change the server ID for those instances.
   a. Confluence: [How to Change the Server ID of Confluence](#)
   b. JIRA: [Changing Server ID for Test Installations](#)

### Important Directories and Files

On this page:

- **JIRA Installation Directory**
  - **Important Files and Directories**
    - `<jira-application-dir>/atlassian-jira/WEB-INF/classes/jira-application.properties`
    - `<jira-application-dir>/atlassian-jira/WEB-INF/classes/jpm.xml`
    - `<jira-application-dir>/atlassian-jira/WEB-INF/lib/`
    - `<jira-application-dir>/atlassian-jira/WEB-INF/classes/log4j.properties`
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 (used throughout the rest of this section), 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.

For example:

<jira-application-dir>/atlassian-jira/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>/atlassian-jira/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 customized (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>/atlassian-jira/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>/atlassian-jira/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>/atlassian-jira/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](https://confluence.atlassian.com/jira/advanced-configuration-settings-225067109.html) 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 customized (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 until 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.

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 (used throughout the rest of this section), 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>/atlassian-jira/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>/atlassian-jira/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 customized (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>/atlassian-jira/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>/atlassian-jira/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>/atlassian-jira/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.
JIRA 6.1 Documentation

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.xm1 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 customized (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 until 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.

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 you 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 (this location should be something different than the application directory, or you may run into problems later).
  
  - If you are specifying this location's path on Windows, use double back-slashes ('\') between subdirectories. For example, `X:\path\to\JIRA\home`.
  
  - If you define an UNC path in Microsoft Windows, be sure to double escape the leading backslash: `\\machinename\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 (`=`)
          - Please set 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
        ```
      - Please set the command above in a script used to start JIRA.
- 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.

**Notes**

- 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.

*Please Note:* If you specify 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.

Alternatively, you can specify the location of your JIRA Home Directory as property within your application server:
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.

What is stored in the JIRA Home Directory?

The following page describes the data stored in the JIRA Home Directory: JIRA Home Directory.

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 Administrator 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:
  ```bash
  sudo adduser jira-tomcat
  ```
- Run Tomcat as a specific user:
  ```bash
  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:

![Apache Tomcat & Properties](image)

**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:
  ```bash
  sudo tar xzvf apache-tomcat-6.0.20.tar.gz
  ```
- Remove the default webapps:
  ```bash
  sudo rm -rf apache-tomcat-6.0.20/webapps/*
  ```
- Remove write permissions:
  ```bash
  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:
  
  ```bash
  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](https://owasp.org/www-community/virtual-server)
- [Critical Steps to Secure Tomcat on Windows NT/2K/XP](http://www.tomcat.apache.org/security/v7/critical.html)
- [Tomcat Security FAQ](http://tomcat.apache.org/security.html)

**Customizing Your JIRA Installation**

- [Changing JIRA's TCP Ports](http://confluence.atlassian.com/display/JIRA/Changing+JIRA%27s+TCP+Ports)
- [Running JIRA over SSL or HTTPS](http://confluence.atlassian.com/display/JIRA/Running+JIRA+over+SSL+or+HTTPS)
- [Installing Confluence and JIRA Together](http://confluence.atlassian.com/display/JIRA/Installing+Confluence+and+JIRA+Together)
- [Integrating JIRA with a Web Server](http://confluence.atlassian.com/display/JIRA/Integrating+JIRA+with+a+Web+Server)
- [Securing JIRA with Apache HTTP Server](http://confluence.atlassian.com/display/JIRA/Securing+JIRA+with+Apache+HTTP+Server)

**Changing JIRA's TCP Ports**

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).

Changing JIRA’s TCP Ports

Before you change JIRA’s TCP ports, read the following:

- **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.

- **If you are using 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.

You can change JIRA’s TCP ports by using the JIRA configuration tool or by manually editing the server.xml file. If you installed JIRA using the ‘Windows Installer’, ‘Linux Installer’ or from an ‘Archive File’, you can use the JIRA configuration tool. The JIRA WAR distribution does not include this tool.

**Changing JIRA’s TCP Ports using the JIRA Configuration tool**

1. Start the JIRA configuration tool, see Using the JIRA Configuration Tool for instructions on where to find the tool.
2. Click the Web Server tab.
3. In the HTTP Port field, enter the new TCP listening port number.
4. In the Control Port field, enter the new TCP shutdown port number.
5. Click the Save button. Your changes are saved to the server.xml file located in the conf subdirectory of your JIRA Installation Directory.

**Changing JIRA’s TCP Ports by editing the server.xml file**

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). 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" />

  ...
```

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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.

**RELATED TOPICS**

Changing Confluence's listening ports

Running JIRA over SSL or HTTPS

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian cannot 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.

The instructions on this page describe how to run JIRA over SSL or HTTPS by configuring Apache Tomcat with HTTPS. This procedure 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 setup.

**Why should you run 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.

**Before you begin**

Please note the following before you begin:

- Atlassian Support will refer SSL support to the Certificate Authority (CA) that issues the Certificate. The SSL-related instructions on this page are provided as a reference only.
- 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:**

- Before you begin
- Generate the Java KeyStore
- Configuring your web server using the JIRA configuration tool
- Advanced configuration
- Troubleshooting

**Related topics:**

Using the JIRA Configuration Tool

Configuring JIRA Options

Connecting to SSL services

Integrating JIRA with Apache using SSL

If hosting JIRA behind a reverse-proxy such as Apache, please follow our [Integrating JIRA with Apache using SSL documentation](#).
In this section, you will create a Java Key Store (JKS) which will hold your SSL certificates. The SSL certificates are required in order for SSL to work in JIRA. In the SSL world, certificates fall into two major categories:

<table>
<thead>
<tr>
<th>Certificate</th>
<th>Description</th>
<th>When to Use</th>
<th>Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-signed</td>
<td>These are certificates that have not been digitally signed by a CA, which is a method of confirming the identity of the certificate that is being served by the web server. They are signed by themselves, hence the name self-signed.</td>
<td>Test, dev or internal servers only.</td>
<td>1 - 13</td>
</tr>
<tr>
<td>CA-signed</td>
<td>A certificate that has had its identity digitally signed by a Certificate Authority (CA). This will allow browsers and clients to trust the certificate.</td>
<td>Production servers.</td>
<td>1 - 21</td>
</tr>
</tbody>
</table>

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)

We recommend using a CA-signed certificate.

If you’re unable to install Portecle on the server due to security constraints, please see our [Command Line Installation](#) section below. Otherwise, continue with the below steps for generating both self and CA-signed certificates.

1. Download and install the Portecle app onto the server that runs JIRA.
   - This is a third-party application and is not supported by Atlassian.
2. Run the App as an Administrator, so it will have the appropriate permissions. Also, ensure the `<JAVA_HOME>` variable is pointing to the same version of Java that JIRA uses. See our [Setting JAVA_HOME](#) docs for further information on this.
   - If running on a Linux/UNIX server, X11 will need to be forwarded when connecting to the server (so you can use the GUI), as below:
   ```
   ssh -X user@server
   ```
3. Select the **Create a new Keystore** option:
4. Select the type **JKS** and OK:

Select the type of the new Keystore:
- JKS
- PKCS #12
- JCEKS
- JKS (case sensitive)
- BKS
- JBER
- GKR

5. Select the **Generate Key Pair** button:
6. Select the RSA algorithm and a Key Size of 2048:

   Key Algorithm:  
   - DSA
   - RSA

   Key Size: 2048

   [OK] [Cancel]

7. Make sure the **Signature Algorithm** is "SHA1withRSA" and then edit the certificate details, as per the below example and select OK:

   Signature Algorithm: SHA1withRSA

   Validity (days): 365

   Common Name (CN): jira.atlassian.com

   Organisation Unit (OU): IT

   Organisation Name (O): ATLASSIAN PTY LTD

   Locality Name (L): Sydney

   State Name (ST): NSW

   Country (C): US

   Email (E): admin@example.com

   [OK] [Cancel]

⚠️ The **Common Name** **MUST** match the server’s URL, otherwise errors will be displayed in the browser.

8. Choose an alias for the certificate - for example jira.
9. Enter a password for the KeyStore (the default password used is typically changeit).
10. The Key Pair Generation will report as successful, as per the below example:

<table>
<thead>
<tr>
<th>Alias Name</th>
<th>Last Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira</td>
<td>04/05/2013 9:47:03 AM</td>
</tr>
</tbody>
</table>

![Key Pair Generation Successful](image)

11. Save the KeyStore in `<JIRA_INSTALL>/jira.jks`, ensuring the use the same password in step 11. This can be done by **File > Save Keystore**.

   ![Keystore type: JKS, provider: SUN, size: 1 entry](image)

   **Warning:** If using a self-signed certificate, proceed to Configuring your web server using the JIRA configuration tool, otherwise continue on.

12. We need to generate a Certificate Signing Request for the CA to sign and confirm the identity of the certificate. To do so, right click on the certificate and choose **Generate CSR**. Save it in `<JIRA_INSTALL>/jira.csr`. 

**Configuring your web server using the JIRA configuration tool**
13. Submit the CSR to a Certificate Authority for signing. They will provide a signed certificate (CA reply) and a set of root/intermediate CA certificates.

14. Import the root and/or intermediate CA certificates with **Import Trusted Certificate**, repeating this step for each certificate.
15. Import the signed certificate by right clicking on the jira certificate and selecting **Import CA Reply**:

![Certificate Details]

<table>
<thead>
<tr>
<th>Alias Name</th>
<th>Last Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira</td>
<td>04/06/2013 9:47:03 AM</td>
</tr>
</tbody>
</table>

16. Select the certificate provided by the CA, which should be jira.crt. This will respond with CA Reply Import successful.

17. Verify this by checking **Tools > Keystore Report**. It should display the certificate as a child of the root certificates.

18. Save the KeyStore and proceed to the next section.

### Configuring your web server using the JIRA configuration tool

In this section, you will finish setting up SSL encryption for JIRA, by configuring your web server using the JIRA configuration tool. For more information on the JIRA configuration tool, see **Using the JIRA Configuration Tool**.

**To configure your web server using the JIRA configuration tool:**

1. Run the JIRA configuration tool, as follows:
   - **Windows**: Open a command prompt and run `config.bat` in the `bin` sub-directory of the JIRA Installation Directory.
   - **Linux/Unix**: Open a console and execute `config.sh` in the `bin` sub-directory of the JIRA Installation Directory.

   ![Important note]
   This may fail with the error as described in our **Unable to Start JIRA Config Tool due to No X11 DISPLAY variable was set error** KB article. Please refer to it for the workaround.

2. Click the **Web Server** tab.

   ![Screenshot: JIRA configuration tool — 'Web Server' tab]
3. Fill out the fields as follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Port</td>
<td>Leave as default. You can change the port number if you wish. See Changing JIRA's TCP Ports.</td>
</tr>
</tbody>
</table>
| Profile       | A profile is a preset web server configuration. You can pick from the four following values:  
|               | • Disabled  
|               | • HTTP only  
|               | • HTTP & HTTPS (redirect HTTP to HTTPS)  
|               | • HTTPS only  
|               | To run JIRA over HTTPS, you must pick either 'HTTP & HTTPS' or 'HTTPS'.  
|               | Pick 'HTTP & HTTPS' if you want to run JIRA over HTTPS but you have users that access JIRA via HTTP. If you pick 'HTTP & HTTPS', users who try to access JIRA via HTTP will be redirected to the HTTPS address. |
| HTTP port     | Leave as default (8080). You can change the port number if you wish. See Changing JIRA's TCP Ports.  
|               | This will be disabled if you set the Profile to 'HTTPS only'. |
| HTTPS port    | Leave as default (8443). You can change the port number if you wish. See Changing JIRA's TCP Ports. |
### Keystore path
Specify the location of the keystore of your certificate. This will have been chosen when the keystore was saved in step 13 and should be `<JIRA_INSTALL>/jira.jks`.

### Keystore password
Specify the password for your keystore. If you generated a self-signed certificate, this is the password you specified for the key and keystore when generating the certificate in step 13.

### Keystore alias
Each entry in the keystore is identified by an alias. We recommend using `jira` for this certificate as in step 10.

4. Click the **Check Certificate in Key Store** button to validate the following:
   - Test whether the certificate can be found in the key store.
   - Test whether keystore password works.
   - Test whether key can be found using key alias.
5. Click the **Save** button to save your changes.

### Advanced configuration

#### Running more than one instance on the same host

When running more than one instance on the same host, it is important to specify the `address` attribute in the `<JIRA_INSTALL>/conf/server.xml` file because by default the connector will listen on all available network interfaces, so specifying the address will prevent conflicts with connectors running on the same default port. See the Tomcat Connector documentation for more about setting the address attribute in the [Apache Tomcat 7 docs](https://tomcat.apache.org/docs/7.0/config/http.html).

### Command Line Installation

**Expand for instructions...**

1. Generate the Java KeyStore (JKS):
   ```bash
   <JAVA_HOME>/keytool -genkey -alias jira -keyalg RSA -keystore <JIRA_INSTALL>/jira.jks
   ```
   
   *Instead of first and last name, enter the server URL, excluding "https://" (e.g.: jira.atlassian.com).*

2. Enter an appropriate password (e.g.: `changeit`).
3. Create the CSR for signing, using the password from step 2:
   ```bash
   <JAVA_HOME>/keytool -certreq -keyalg RSA -alias jira -keystore <JIRA_INSTALL>/jira.jks -file jira.csr
   ```

4. Submit the CSR to the CA for signing. They will provide a signed certificate and a root and/or intermediate CA.
   *If the certificate will not be signed, skip to step 7.*

5. Import the root and/or intermediate CA:
   ```bash
   <JAVA_HOME>/keytool -import -alias rootCA -keystore <JIRA_INSTALL>/jira.jks -trustcacerts -file root.crt
   ```

6. Import the signed certificate (this is provided by the CA):
Follow the Configuring your web server using the JIRA configuration tool instructions.

Redirecting certain pages to HTTPS

You can redirect users from HTTP URLs to HTTPS URLs by choosing the 'HTTP & HTTPS' profile in the JIRA configuration tool. However, if you want to only redirect certain pages to HTTPS, you can do this manually. To do this, select the 'HTTPS only' profile in the JIRA configuration tool and save the configuration. Then edit the WEB-INF/web.xml file and add 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>*.jspx</url-pattern>
    <url-pattern>/browse/*</url-pattern>
  </web-resource-collection>
  <user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

In the example above, all URLs except attachments are redirected from HTTP to HTTPS.

Restart JIRA after you have saved your changes.

Troubleshooting

Here are some troubleshooting tips if you are using a self-signed key created by Portecle, as described above.

When you enter "https://localhost:<port number>" 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" \n  nokeepalive ssl-unclean-shutdown \ndowngrade-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:

```xml
keystoreFile="<location of keystore file>"
```

This can also happen ("Cannot find /root/.keystore") if you add a `keystoreFile` attribute to the `http` connector in `server.xml` instead of the `https` connector.

- **Certificate reply and certificate in keystore are identical:**

  ```java
  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
  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
  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:**

  ```java
  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:
Using Apache Portable Runtime:

APR uses a different SSL engine, and you will see an exception like this in your logs:

```java
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.

```xml
<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.

```xml
<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 Configuration HOW-TO Tomcat 7 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` 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 `http://localhost:8080/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`:

   ```xml
   <Connector port="8009" enableLookups="false" redirectPort="8443" protocol="AJP/1.3" />
   
   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:

   ```xml
   <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.properties` 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 u
riworkermap.properties tells IIS what requests to proxy to Tomcat.
To edit these files:
a. Edit the uriworkermap.properties and 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 jir
a. 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 'IIS 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 (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:
```bash
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
This page describes how to integrate Apache HTTP Server (also referred to as `httpd`) with JIRA, utilising `mod_proxy` so that Apache operates as a reverse-proxy over HTTP. If HTTPS configuration is required, please see our Integrating JIRA with Apache using SSL documentation. Configuring Apache allows for running JIRA on non-standard HTTP port (such as 8080) and users will be able to access JIRA over standard HTTP as their traffic will be routed through the proxy.

Apache can be configured to allow access to JIRA in any of the following methods:

- Directly on its own domain: http://jira.com
- As a subdomain of another domain: http://jira.atlassian.com
- It can also be accessed on a context path on either a domain or subdomain: http://atlassian.com/jira

This documentation will cover a straightforward implementation of `mod_proxy` using the above three configurations. If a more complication solution is required, refer to the Apache HTTP Server Version Documentation, consult with the Apache SME within your organisation and if need be raise a question on Atlassian Answers or look at getting in touch with one of our Atlassian Experts.

Expand for an example of a common Apache configuration

1. JIRA is running on port 8080 on a server within the LAN that cannot be accessed externally (the router/firewall is not forwarding port 8080 to it).
2. Apache is set up on another server (or the same server as JIRA) that can be accessed externally on HTTP (80).
3. Apache is then accessed over HTTP on the appropriate URL (`VirtualHost`), routing the traffic to and from the JIRA server.

---

### Step 1: Configure Tomcat

This step is only required if JIRA will be accessed on a context path, for example http://atlassian.com/jira. If this is not required, this step can be skipped.

1. Stop JIRA.
2. Edit Tomcat's `server.xml` to include the required JIRA context path. The below example uses `path="jira"` - this means JIRA is accessible on `http://jiraserver:8080/jira` given the default JIRA port used.
<Engine defaultHost="localhost" name="Catalina">
  <Host appBase="webapps" autoDeploy="true" name="localhost"
  unpackWARs="true">
    <Context docBase="${catalina.home}/atlassian-jira"
    path="/jira" reloadable="false" useHttpOnly="true">
      <!--
      Note, you no longer configure your database driver or connection parameters here.
      These are configured through the UI during application setup.
      -->
    </Context>
  </Host>
</Engine>

Ensure the path value is set with a prepending forward slash (/). For example, path="/jira" rather than path="jira".

3. Start JIRA.
4. Test that JIRA is accessible on the new context path, for example http://jiraserver:8080/jira.

Step 2: Configure Apache HTTP Server

The installation of Apache and configuration of a DNS is not covered in this documentation. Additionally, it is assumed that Apache 2.2 has been installed and DNS entries have been configured for the JIRA domain. As Apache's configuration is specific to the operation system that is used, only some distributions and their configurations are currently documented.

2.1 Enable the Proxy Modules

Debian/Ubuntu
 Expand to see Debian/Ubuntu instructions
1. Enable the module with the following:

$ sudo a2enmod proxy_http
Considering dependency proxy for proxy_http:
Enabling module proxy.
Enabling module proxy_http.
To activate the new configuration, you need to run:
  service apache2 restart

2. Restart Apache.

Windows/Other OS
 Expand to see Windows/Other OS instructions
1. Locate and edit the httpd.conf file, adding the below lines:
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_http_module modules/mod_proxy_http.so

2. Restart Apache.

2.2. Configure Apache to use those Modules

Debian/Ubuntu

Expand to see Debian/Ubuntu instructions

1. Switch into user root.
2. Backup the existing site or create a new one. Creating a new site is not covered within this documentation (copying the default should be sufficient).
3. Modify the existing site within $APACHE_INSTALL/sites-available, for example default.
4. Add the following inside the VirtualHost, replacing jiraserver with the hostname of the JIRA server and also modifying the port if required.

On its own domain or subdomain:

```
# JIRA Proxy Configuration:
<Proxy *>
  Order deny,allow
  Allow from all
</Proxy>

ProxyRequests Off
ProxyPreserveHost On
ProxyPass / http://jiraserver:8080/
ProxyPassReverse / http://jiraserver:8080/
```

Missing a forward slash at the end of the URL will cause proxy errors - ensure this is in place!

Using a context path:

```
# JIRA Proxy Configuration:
<Proxy *>
  Order deny,allow
  Allow from all
</Proxy>

ProxyRequests Off
ProxyPreserveHost On
ProxyPass /jira http://jiraserver:8080/jira
ProxyPassReverse /jira http://jiraserver:8080/jira
```

The path used must be identical to the Tomcat context path. For example, forwarding /jira to /jira520 cannot be done without considerable rewrite rules that are not always reliable.

5. (Optional): Enable the site with the following:

```
# a2ensite jira
Enabling site jira.
To activate the new configuration, you need to run:
service apache2 reload
```

This is only required if a new site has been created in favour of using the default.
6. Reload the Apache configuration.
7. Test by accessing JIRA through Apache, for example http://jira.com or http://atlassian.com/jira.

Windows/Other OS

Expand to see Windows/Other OS instructions

1. Locate and edit the httpd.conf file.
2. Add the following inside the VirtualHost, replacing jiraserver with the hostname of the JIRA server and also modifying the port if required.

On its own domain or subdomain:

```
# JIRA Proxy Configuration:
<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>

ProxyRequests Off
ProxyPreserveHost On
ProxyPass / http://jiraserver:8080/
ProxyPassReverse / http://jiraserver:8080/
```

- Missing a forward slash at the end of the URL will cause proxy errors - ensure this is in place!

Using a context path:

```
# JIRA Proxy Configuration:
<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>

ProxyRequests Off
ProxyPreserveHost On
ProxyPass /jira http://jiraserver:8080/jira
ProxyPassReverse /jira http://jiraserver:8080/jira
```

- The path used must be identical to the Tomcat context path. For example, forwarding /jira to /jira520 cannot be done without considerable rewrite rules that are not always reliable.

3. Restart Apache.
4. Test by accessing JIRA through Apache, for example http://jira.com or http://atlassian.com/jira.

---

**Step 3: Configure JIRA**

1. Set Use gzip compression to OFF as in Configuring JIRA Options. GZIP compression is known to cause performance issues using a reverse-proxy, especially if the proxy is also compressing the traffic.
2. Set the Base URL to be the FQDN that JIRA will be accessed on, for example http://jira.atlassian.com. This is also located in Configuring JIRA Options. **WARNING** JIRA can only be configured to respond to a single URL and the Base URL (as in Configuring JIRA Options) must match the URL end-users are accessing. Misconfiguration of this may cause significant problems within JIRA such as the Activity Stream and Dashboard Gadgets failing to function correctly.
3. Test by accessing JIRA on the FQDN (e.g.: http://jira.atlassian.com), ensuring that JIRA is accessible and all dashboard gadgets correctly display.

---

**Troubleshooting**

- **Hijacked Sessions:** Some users have reported problems with user sessions being hijacked when the mod_cache module is enabled. If these problems are encountered, try disabling the mod_cache module.
This module is enabled by default in some Apache HTTP Server version 2 distributions.

- **Permission Denied Errors enabling mod_proxy (and mod_jk) on Linux distros that use SELinux:** Users have reported ‘permission denied’ errors when trying to get mod_proxy (and mod_jk) working. Disabling SELinux (/etc/selinux/config) apparently fixes this.

- **Running Mac OS X:** 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, as below.

  > Additionally we do not recommend using Max OS X as it is not supported, as in our Supported Platforms.

  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 :-


  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.

- **Too many redirects:** Both Tomcat & Apache are redirecting, when only one should be. Disable redirection in Tomcat (revert any changes as in Running JIRA over SSL or HTTPS) and check that there is only one redirection in Apache.

- **General Problems:**
  1. Clear the browser cache and try again.
  2. Ensure that JIRA works as expected when running directly from Tomcat and bypassing Apache. For example, accessing http://jiraserver:8080 instead of http://jira.atlassian.com.
  3. Increase the LogLevel for Apache to debug and restart it.
  4. Attempt to access JIRA and check the Apache Log Files for any errors.
  5. Raise a question on Atlassian Answers for assistance.

- **403 Forbidden error:**
  - Add the RequestHeader unset Authorization line to the apache configuration page to disable authorization headers.

```<Location /jira>
  RequestHeader unset Authorization
  ProxyPreserveHost On
  ProxyPass http://jiraserver/jira
  ProxyPassReverse http://jiraserver/jira
</Location>```

**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.
- Apache Virtual Host documentation

Configuring Apache Reverse Proxy Using the AJP Protocol

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian cannot 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 Apache HTTP Server (also referred to as httpd) with JIRA, utilising mod_proxy_aip so that Apache operates as a reverse-proxy. AJP is a wire protocol and is an optimized version of the HTTP protocol to allow a standalone web server such as Apache to talk to Tomcat.

This protocol can be used in favour of HTTP/1.1 as in either of the following Apache configurations:

- Integrating JIRA with Apache
- Integrating JIRA with Apache using SSL
Step 1: Configure Tomcat

1. Stop JIRA.
2. Enable the AJP Connector on the Tomcat container hosting JIRA by uncommenting the following element in $JIRA_INSTALL/conf/server.xml:

```xml
<Connector port="8009" URIEncoding="UTF-8" enableLookups="false" redirectPort="8443" protocol="AJP/1.3" />
```
3. Start JIRA.
4. Test that JIRA is accessible on the standard HTTP connector, for example http://jiraserver:8080. This is to ensure that Tomcat has successfully restarted.

Step 2: Configure Apache HTTP Server

The installation of Apache and configuration of a DNS is not covered in this documentation. Additionally, it is assumed that Apache 2.2 has been installed and DNS entries have been configured for the JIRA domain. As Apache’s configuration is specific to the operation system that is used, only some distributions and their configurations are currently documented.

2.1 Enable the Proxy Modules

**Debian/Ubuntu**

```
$ sudo a2enmod proxy_ajp
```
Considering dependency proxy for proxy_ajp:
Module proxy already enabled
Enabling module proxy_ajp.
To activate the new configuration, you need to run:
```
service apache2 restart
```

**Windows/Other OS**

```
1. Locate and edit the httpd.conf file, adding the below lines:

LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
```
2. Restart Apache.

2.2 Configure Apache to use those Modules

**Debian/Ubuntu**

```
1. Switch into user root.
```
2. Backup the existing site or create a new one. Creating a new site is not covered within this documentation (copying the default should be sufficient).
3. Modify the existing site within $APACHE_INSTALL/sites-available, for example default (HTTP) or default-ssl (HTTPS).
4. Add the following inside the VirtualHost, replacing jiraserver with the hostname of the JIRA server and also modifying the port if required.

```
# JIRA AJP Proxy Configuration:
<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>

ProxyRequests           Off
ProxyPass               /       ajp://jiraserver:8009/
ProxyPassReverse        /       ajp://jiraserver:8009/
```

* Missing a forward slash at the end of the URL will cause proxy errors - ensure this is in place!*

5. *(Optional):* Enable the site with the following:

```
# a2ensite jira
Enabling site jira.
To activate the new configuration, you need to run:
    service apache2 reload
```

* This is only required if a new site has been created in favour of using the default.*

6. **If using HTTP, skip to step 8.** For HTTPS, the certificates need to be installed by copying the certificate and private key to the appropriate directories and the following will also need to be added to the site:

```
SSLProxyEngine          On
```

7. Include them in the Apache configuration, within the VirtualHost as below:

```
SSLCertificateFile    /etc/ssl/certs/jira.crt
SSLCertificateKeyFile /etc/ssl/private/jira.key
```

8. Reload the Apache configuration.

9. Test by accessing JIRA through Apache, for example http://jira.com or http://atlassian.com/jira.

Windows/Other OS

Expand to see Windows/Other OS instructions

1. Locate and edit the httpd.conf file.
2. Add the following inside the VirtualHost, replacing jiraserver with the hostname of the JIRA server and also modifying the port if required.
# JIRA AJP Proxy Configuration:

```xml
<Proxy *>
  Order deny,allow
  Allow from all
</Proxy>
```

ProxyRequests Off
ProxyPass / ajp://jiraserver:8009/
ProxyPassReverse / ajp://jiraserver:8009/

Missing a forward slash at the end of the URL will cause proxy errors - ensure this is in place!

3. **If using HTTP, skip to step 5.** For HTTPS, the certificates need to be installed by copying the certificate and private key to the appropriate directories and the following will also need to be added to the site:

```plaintext
SSLProxyEngine On
```

4. Include them in the Apache configuration, within the VirtualHost as below:

```plaintext
SSLCertificateFile /etc/ssl/certs/jira.crt
SSLCertificateKeyFile /etc/ssl/private/jira.key
```

5. Restart Apache.


## 2.3 Redirect HTTP to HTTPS

This is an optional step and is only required if using HTTPS. It can be done by using `mod_rewrite` (this module may require enabling), add the following to the HTTP VirtualHost:

```plaintext
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
```

## Step 3: Configure JIRA

1. Set **Use gzip compression** to OFF as in Configuring JIRA Options. GZIP compression is known to cause performance issues using a reverse-proxy, especially if the proxy is also compressing the traffic.

2. Set the **Base URL** to be the FQDN that JIRA will be accessed on, for example [http://jira.atlassian.com](http://jira.atlassian.com). This is also located in Configuring JIRA Options.

   **Warning:** JIRA can only be configured to respond to a single URL and the Base URL (as in Configuring JIRA Options) must match the URL end-users are accessing. Misconfiguration of this may cause significant problems within JIRA such as the Activity Stream and Dashboard Gadgets failing to function correctly.

3. Test by accessing JIRA on the FQDN (e.g.: [http://jira.atlassian.com](http://jira.atlassian.com)), ensuring that JIRA is accessible and all dashboard gadgets correctly display.

## Troubleshooting

- **Hijacked Sessions:** Some users have reported problems with user sessions being hijacked when the `mod_cache` module is enabled. If these problems are encountered, try disabling the `mod_cache` module.

  This module is enabled by default in some Apache HTTP Server version 2 distributions.

- **Permission Denied Errors enabling mod_proxy (and mod_jk) on Linux distros that use SELinux:** Users have reported 'permission denied' errors when trying to get `mod_proxy` (and `mod_jk`) working. Disabling SELinux (/etc/selinux/config) apparently fixes this.
Running Mac OS X: 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, as below. Additionally we do not recommend using Max OS X as it is not supported, as in our Supported Platforms.

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.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.

**Too many redirects:** Both Tomcat & Apache are redirecting, when only one should be. Disable redirection in Tomcat (revert any changes as in Running JIRA over SSL or HTTPS) and check that there is only one redirection in Apache.

**General Problems:**
1. Clear the browser cache and try again.
2. Ensure that JIRA works as expected when running directly from Tomcat and bypassing Apache.
   For example, accessing `http://jiraserver:8080` instead of `http://jira.atlassian.com`
3. Increase the LogLevel for Apache to debug and restart it.
4. Attempt to access JIRA and check the Apache Log Files for any errors.
5. Raise a question on Atlassian Answers for assistance.

**403 Forbidden error:**
   - Add the `RequestHeader unset Authorization` line to the apache configuration page to disable authorization headers.

   ```
   <Location /jira>
      RequestHeader unset Authorization
      ProxyPreserveHost On
      ProxyPass http://jiraserver/jira
      ProxyPassReverse http://jiraserver/jira
   </Location>
   ```

**See Also**

- Integrating JIRA with Apache
- Integrating JIRA with Apache using SSL
- Apache Virtual Host documentation

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 how to integrate Apache HTTP Server (also referred to as `httpd`) with JIRA, utilising `mod_proxy` & `mod_ssl` so that Apache operates as a reverse-proxy over HTTPS. If a HTTP configuration is required, please see our Integrating JIRA with Apache documentation. Configuring Apache allows for running JIRA on non-standard HTTP port (such as 8080) and users will be able to access JIRA over standard HTTPS as their traffic will be routed through the proxy and encrypted outside of the network.

Apache can be configured to allow access to JIRA in any of the following methods:
- Directly on its own domain: `https://atlassian.com`
- As a subdomain of another domain: `https://jira.atlassian.com`
- It can also be accessed on a context path on either a domain or subdomain: `https://atlassian.com/jira`

This means the SSL certificate will be managed within Apache and not Tomcat, additionally the connection between Apache and Tomcat will not be encrypted. However, the connection between the browser and the outside network will be encrypted. This is suitable for configurations where the JIRA server is within the same network as the Apache server and is illustrated below:
Client Browser -> HTTPS -> Apache Proxy -> HTTP -> Tomcat (JIRA)

This is a common configuration for networks with multiple SSL certificates and/or web applications as they are all managed in one location (Apache).

If a more complicated solution is required, refer to the Apache HTTP Server Version Documentation, consult with the Apache SME within your organisation and if need be raise a question on Atlassian Answers or look at getting in touch with one of our Atlassian Experts.

Expand for an example of a common Apache configuration

1. JIRA is running on port 8080 on a server within the LAN that cannot be accessed externally (the router/firewall is not forwarding port 8080 to it).
2. Apache is set up on another server (or the same server as JIRA) that can be accessed externally on HTTPS (443).
3. Apache is then accessed over HTTPS on the appropriate URL (VirtualHost), routing the traffic to and from the JIRA server.

### On this page:
- Before you begin
- Step 1: Configure Tomcat
- Step 2: Configure Apache HTTP Server
  - 2.1 Enable the Proxy Modules
  - 2.2 Configure Apache to use those Modules
  - 2.3 Redirect HTTP to HTTPS
- Step 3: Configure JIRA
- Troubleshooting
- See Also

**Before you begin**

⚠️ It is expected that the SSL certificate has been signed by a CA and is in the PEM format prior to configuring Apache. For assistance preparing and generating SSL certificates, please consult with a SSL Vendor (for example, GoDaddy, Verisign, RapidSSL).

Identifying whether to use a domain, subdomain or context path largely depends on the type of SSL certificate provided and also any business rules around website configurations. For SSL to function without error, the domain must match the Common Name (CN) of the certificate.

Expand for further information on configuring the FQDN to match the certificate’s CN

This table indicates which URLs will work with the certificate CN and also makes a recommendation on the URL to use.

<table>
<thead>
<tr>
<th>JIRA FQDN</th>
<th>Common Name</th>
<th>Valid</th>
<th>Recommend JIRA FQDN</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://jira.atlassian.com">https://jira.atlassian.com</a></td>
<td>jira.atlassian.com</td>
<td>✔️</td>
<td><a href="https://jira.atlassian.com">https://jira.atlassian.com</a></td>
</tr>
<tr>
<td><a href="https://atlassian.com">https://atlassian.com</a></td>
<td>jira.atlassian.com</td>
<td>✗</td>
<td><a href="https://jira.atlassian.com">https://jira.atlassian.com</a></td>
</tr>
</tbody>
</table>

A certificate that has a CN with an asterisk (*) in it is a wildcard certificate and can support any subdomain of that domain. If you are uncertain about the URL to use, please consult with your System Administrator and the
SSL vendor that provided the certificate.

**Step 1: Configure Tomcat**

1. Stop JIRA.
2. *(Optional: If JIRA does not require a context path, skip this step.)*

Edit Tomcat's `server.xml` to include the required JIRA context path. The below example uses `path="jira"` - this means JIRA is accessible on `http://jiraserver:8080/jira` given the default JIRA port is used.

```xml
<Host appBase="webapps" autoDeploy="true" name="localhost" unpackWARs="true">
    <Context docBase="${catalina.home}/atlassian-jira" path="/jira" reloadable="false" useHttpOnly="true">
        <!--
        Note, you no longer configure your database driver or connection parameters here.
        These are configured through the UI during application setup.
        -->
        <Resource auth="Container" factory="org.objectweb.jotm.UserTransactionFactory" jotm.timeout="60" name="UserTransaction" type="javax.transaction.UserTransaction"/>
        <Manager pathname=""/>
    </Context>
</Host>
```

Ensure the `path` value is set with a prepending forward slash (`/`). For example, `path="/jira"` rather than `path="jira"`.

3. Edit Tomcat's `server.xml` to include a separate connector to proxy the requests. This requires the `scheme`, `proxyName` & `proxyPort` attributes. Replace them with the appropriate domain and port of the proxy, as in the below example:

```xml
<Service name="Catalina">
    <!-- Apache Proxy Connector -->
    <Connector acceptCount="100" connectionTimeout="20000" disableUploadTimeout="true" enableLookups="false" maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25" port="8080" protocol="HTTP/1.1" redirectPort="8443" useBodyEncodingForURI="true" scheme="https" proxyName="jira.atlassian.com" proxyPort="443"/>

    <!-- Standard HTTP Connector -->
    <Connector acceptCount="100" connectionTimeout="20000" disableUploadTimeout="true" enableLookups="false" maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25" port="8081" protocol="HTTP/1.1" redirectPort="8443" useBodyEncodingForURI="true"/>
</Service>
```

4. Disable any redirections within Tomcat to HTTPS if they have been enabled - for example the changes to `WEB-INF/web.xml` in **Running JIRA over SSL or HTTPS** will cause errors when using Apache.

5. Start JIRA.
6. Test that JIRA is accessible on the normal connector, using a context path if applicable - for example http://jiraserver:8081/jira.
7. Test that the new connector is working by accessing JIRA on the appropriate proxy connector, for example http://jiraserver:8080/. This should redirect to the proxy FQDN (in this example, https://jira.atlassian.com), which will fail as the proxy is not yet configured. The test is to ensure Tomcat is set up to correctly redirect to the proxy.

We use two different Tomcat connectors so that testing can be done on JIRA, bypassing the proxy when needed as this is a useful step when troubleshooting. It is expected that the standard connector will not be allowed external access from outside the network (the firewall will not forward any ports to it).

Step 2: Configure Apache HTTP Server

The installation of Apache and configuration of a DNS is not covered in this documentation. Additionally, it is assumed that Apache 2.2 has been installed and DNS entries have been configured for the JIRA domain. As Apache's configuration is specific to the operation system that is used, only some distributions and their configurations are currently documented.

2.1 Enable the Proxy Modules

Debian/Ubuntu

Enable the module with the following:

```
$ sudo a2enmod proxy_http ssl
Considering dependency proxy for proxy_http:
Enabling module proxy.
Enabling module proxy_http.
Enabling module ssl.
See /usr/share/doc/apache2.2-common/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
    service apache2 restart
```

Windows/Other OS

Locate and edit the httpd.conf file, adding the below lines if they do not already exist:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule ssl_module modules/mod_ssl.so
```

2. Restart Apache.

2.2. Configure Apache to use those Modules

Debian/Ubuntu

Switch into user root.

1. Backup the existing site or create a new one. Creating a new site is not covered within this documentation (copying the default should be sufficient).
2. Modify the existing site within $APACHE_INSTALL/sites-available, for example default-ssl.
3. Add the following inside the VirtualHost, replacing jiraserver with the hostname of the JIRA server and also modifying the port if required.
4. On its own domain or subdomain:
# JIRA Proxy Configuration:

```
<Proxy *
  Order deny,allow
  Allow from all
</Proxy>
```

SSLProxyEngine On
ProxyRequests Off
ProxyPreserveHost On
ProxyPass / http://jiraserver:8080/
ProxyPassReverse / http://jiraserver:8080/

## Missing a forward slash at the end of the URL will cause proxy errors - ensure this is in place!

Using a context path:

```
# JIRA Proxy Configuration:
<Proxy */jira
  Order deny,allow
  Allow from all
</Proxy>
```

SSLProxyEngine On
ProxyRequests Off
ProxyPreserveHost On
ProxyPass /jira http://jiraserver:8080/jira
ProxyPassReverse /jira http://jiraserver:8080/jira

## The path used must be identical to the Tomcat context path. For example, forwarding /jira to /jira520 cannot be done without considerable rewrite rules that are not always reliable.

5. Enable the site with the following:

```
# a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
service apache2 reload
```

6. Copy the certificate and private key to the appropriate directories.
7. Include them in the Apache configuration, within the VirtualHost as below:

```
SSLCertificateFile /etc/ssl/certs/jira.crt
SSLCertificateKeyFile /etc/ssl/private/jira.key
```

8. Reload the Apache configuration.
9. Test by accessing JIRA through Apache, for example http://jira.com or http://atlassian.com/jira.

**Windows/Other OS**

- Expand to see Windows/Other OS instructions

1. Locate and edit the `httpd.conf` file.
2. Add the following inside the VirtualHost, replacing `jiraserver` with the hostname of the JIRA server and also modifying the port if required.

   **On its own domain or subdomain:**

   ```
   SSLEngine On
   SSLEngineOptions CipherSuite
   SSLEngineOptions Protocols TLSv1.2
   ```
# JIRA Proxy Configuration:

```xml
<Proxy *>
  Order deny,allow
  Allow from all
</Proxy>

SSLProxyEngine  On
ProxyRequests    Off
ProxyPreserveHost On
ProxyPass        /       http://jiraserver:8080/
ProxyPassReverse /       http://jiraserver:8080/
```

- Missing a forward slash at the end of the URL will cause proxy errors - ensure this is in place!

### Using a context path:

```xml
<Proxy *>
  Order deny,allow
  Allow from all
</Proxy>

SSLProxyEngine  On
ProxyRequests    Off
ProxyPreserveHost On
ProxyPass        /jira       http://jiraserver:8080/jira
ProxyPassReverse /jira       http://jiraserver:8080/jira
```

- The path used must be identical to the Tomcat context path. For example, forwarding `/jira` to `/jira` cannot be done without considerable rewrite rules that are not always reliable.

3. Copy the certificate and private key to the appropriate directories.
4. Include them in the Apache configuration, within the VirtualHost as below:

```xml
SSLCertificateFile    /etc/ssl/certs/jira.crt
SSLCertificateKeyFile /etc/ssl/private/jira.key
```

5. Restart Apache.
6. Test by accessing JIRA through Apache, for example `http://jira.com` or `http://atlassian.com/jira`.

## 2.3 Redirect HTTP to HTTPS

This can be done with either of the following:

- Set up the HTTP VirtualHost to forward to the same Tomcat Connector. Tomcat will redirect to HTTPS using the `scheme`, `proxyName` & `proxyPort` parameters. This can be done as in our Integrating JIRA with Apache documentation.
- Using `mod_rewrite` (this module may require enabling), add the following to the HTTP VirtualHost:

```apache
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
```

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
1. Set **Use gzip compression** to **OFF** as in Configuring JIRA Options. GZIP compression is known to cause performance issues using a reverse-proxy, especially if the proxy is also compressing the traffic.
2. Set the **Base URL** to be the FQDN that JIRA will be accessed on, for example https://jira.atlassian.com. This is also located in Configuring JIRA Options.
   - JIRA can only be configured to respond to a single URL and the Base URL (as in Configuring JIRA Options) must match the URL end-users are accessing. Misconfiguration of this may cause significant problems within JIRA such as the Activity Stream and Dashboard Gadgets failing to function correctly.
3. Test by accessing JIRA on the FQDN (e.g.: https://jira.atlassian.com), ensuring that JIRA is accessible and all dashboard gadgets correctly display.

### Troubleshooting

- **Hijacked Sessions**: Some users have reported problems with user sessions being hijacked when the `mod_cache` module is enabled. If these problems are encountered, try disabling the `mod_cache` module.
  - This module is enabled by default in some Apache HTTP Server version 2 distributions.
- **Permission Denied Errors enabling mod_proxy (and mod_jk) on Linux distros that use SELinux**: Users have reported 'permission denied' errors when trying to get `mod_proxy (and mod_jk)` working. Disabling SELinux (/etc/selinux/config) apparently fixes this.
- **Running Mac OS X**: 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, as below.
  - Additionally we do not recommend using Max OS X as it is not supported, as in our Supported Platforms.

> 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 :-

> 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.

- **Too many redirects**: Both Tomcat & Apache are redirecting, when only one should be. Disable redirection in Tomcat (revert any changes as in Running JIRA over SSL or HTTPS) and check that there is only one redirection in Apache.
- **General Problems**:
  1. Clear the browser cache and try again.
  2. Ensure that JIRA works as expected when running directly from Tomcat and bypassing Apache. For example, accessing http://jiraserver:8080 instead of http://jira.atlassian.com.
  3. Increase the **LogLevel** for Apache to debug and restart it.
  4. Attempt to access JIRA and check the **Apache Log Files** for any errors.
  5. Raise a question on **Atlassian Answers** for assistance.
- **403 Forbidden error**:
  - Add the `RequestHeader unset Authorization` line to the apache configuration page to disable authorization headers.

```nginx
<Location /jira>
  RequestHeader unset Authorization
  ProxyPreserveHost On
  ProxyPass http://jiraserver/jira
  ProxyPassReverse http://jiraserver/jira
</Location>
```

### See Also

- Integrating JIRA with Apache
- Configuring Apache Reverse Proxy Using the AJP Protocol
- For more advanced `mod_webapp` configurations (eg. SSL), see this `mod_proxy` guide.
- Apache Virtual Host documentation

**Troubleshooting Apache**

- **Hijacked Sessions**: Some users have reported problems with user sessions being hijacked when the **mo
The d_cache module is enabled. If these problems are encountered, try disabling the mod_cache module.

- **Permission Denied Errors enabling mod_proxy (and mod_jk) on Linux distros that use SELinux:** Users have reported 'permission denied' errors when trying to get mod_proxy (and mod_jk) working. Disabling SELinux (/etcselinux/config) apparently fixes this.

- **Running Mac OS X:** 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, as below. Additionally we do not recommend using Max OS X as it is not supported, as in our Supported Platforms.

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 :-

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.

- **Too many redirects:** Both Tomcat & Apache are redirecting, when only one should be. Disable redirection in Tomcat (revert any changes as in Running JIRA over SSL or HTTPS) and check that there is only one redirection in Apache.

- **General Problems:**
  1. Clear the browser cache and try again.
  2. Ensure that JIRA works as expected when running directly from Tomcat and bypassing Apache. For example, accessing http://jiraserver:8080 instead of http://jira.atlassian.com.
  3. Increase the LogLevel for Apache to debug and restart it.
  4. Attempt to access JIRA and check the Apache Log Files for any errors.
  5. Raise a question on Atlassian Answers for assistance.

- **403 Forbidden error:**
  - Add the RequestHeader unset Authorization line to the apache configuration page to disable authorization headers.

    ```
    <Location /jira>
    RequestHeader unset Authorization
    ProxyPreserveHost On
    ProxyPass http://jiraserver/jira
    ProxyPassReverse http://jiraserver/jira
    </Location>
    ```

**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>
  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>  
  <LocationMatch ConfigureFieldLayout>  
    Include sysadmin_ips_only.conf  
  </LocationMatch>  
</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>
<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>
<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>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</LocationMatch>
</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 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>
  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>
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 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.

Report a Bug

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 [Atlassian Support Offerings](http://jira.atlassian.com) for more support-related information.

**How to Report a Security Issue**

Finding and Reporting a Security Issue

If you find a security issue in the product, open an issue on https://jira.atlassian.com in the relevant project.

- Set the security level of the bug to ‘Reporters and Developers’.
- Set the priority of the bug to ‘Blocker’.
- Provide as much information on reproducing the bug as possible.

All communication about the security issue 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 cannot find the right project to file your issue in, email the details to security@atlassian.com.

When reporting a security vulnerability, please keep in mind the following:

We need a technical description that allows us to assess exploitability and impact of the issue.

- Provide steps to reproduce the issue, including any URLs or code involved.
- If you are reporting an XSS, your exploit should at least pop up an alert in the browser. It is much better if the XSS exploit shows user's authentication cookie.
- If you are reporting an SQL injection, we want to see the exploit extracting database data, not just

---

*Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
We are not looking for the reports listing generic "best practice" issues such as:

- Specific cookies being not marked as Secure or HTTPOnly
- Presence or absence of HTTP headers (X-Frame-Options, HSTS, CSP, nosniff and so on)
- Clickjacking
- Mixed HTTP and HTTPS content
- Auto-complete enabled or disabled
- SSL-related issues

Further reading

See Atlassian Support Offerings for more support-related information.

**New Features Policy**

**Summary**

- We encourage and display customer comments and votes openly in our issue tracking system, http://jira.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:

- **Customer contact**: We get the chance to meet customers and hear their successes and challenges at Atlassian Summit, Atlassian Unite, developer conferences, and road shows.
- **Customer interviews**: All product managers at Atlassian do customer interviews. Our interviews are not simply to capture a list of features, but to understand our customers' goals and plans.
- **Community forums**: There are large volumes of posts on answers, of votes and comments on jira.atlassian.com, and of conversations on community forums like groups on LinkedIn.
- **Customer Support**: Our support team provides clear insights into the issues that are challenging for customers, and which are generating the most calls to support
- **Atlassian Experts**: Our Experts provide insights into real-world customer deployments, especially for customers at scale.
- **Evaluator Feedback**: When someone new tries our products, we want to know what they liked and disliked and often reach out to them for more detail.
- **In product feedback**: The JIRA Issue Collectors that we embed our products for evaluators and our Early Access Program give us a constant pulse on how users are experiencing our product.
- **Usage data**: Are customers using the features we have developed?
- **Product strategy**: 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 system, http://jira.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 Atlassian Support Offerings 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 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)
- 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 Atlassian Support Offerings for more support-related information.

Security Advisory Publishing Policy

Publication of Security Advisories
When a critical severity 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.
- We will send a copy of all posted 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 'Communications Centre' 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.

If you want to track non-critical severity security vulnerabilities, you need to monitor the issue trackers for the relevant products on http://jira.atlassian.com. For example, https://jira.atlassian.com/browse/JRA for JIRA and https://jira.atlassian.com/browse/CONF for Confluence. Security issues in trackers will be marked with a “security”
label. All security issues will be listed in the release notes of the release where they have been fixed, similar to other bugs.

One of the ways to monitor updates to security issues is subscribing to the results of a sample search via email or RSS.

Further reading

See Atlassian Support Offerings 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 OnDemand, since OnDemand 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 Atlassian Support Offerings 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:

<table>
<thead>
<tr>
<th>CVSS score range</th>
<th>Severity in advisory</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 – 2.9</td>
<td>Low</td>
</tr>
<tr>
<td>3 – 5.9</td>
<td>Medium</td>
</tr>
<tr>
<td>6.0 – 7.9</td>
<td>High</td>
</tr>
<tr>
<td>8.0 – 10.0</td>
<td>Critical</td>
</tr>
</tbody>
</table>

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 most of 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 some of 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 some of 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 Atlassian Support Offerings 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 customized 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 customized 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 customizations we have made to the header, footer and left-hand navigation bar. These customizations 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:
   - Customize 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.:
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 favorites). 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. 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 documentation 'branching' process 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 on 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 officially released, we begin the 'branching' steps (above) earlier during development of the next major version of JIRA. For the release of:

- JIRA 4.3.0 — we performed the branching steps just prior to the release of JIRA 4.3 Beta 1 and at sub-step 2a, the 'JIRA' space was renamed 'JIRA 4.3 Beta'.
- JIRA 4.4.0 — we performed the branching even earlier (just prior to the release of JIRA 4.4 EAP 2) and at sub-step 2a, the 'JIRA' space was renamed 'JIRA 4.4 EAP'.
  - When JIRA 4.4 Beta 1 was released, we performed sub-step 2a again, by renaming the 'JIRA' space '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 JIRA-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 JIRA-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)
- 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
How to disable or enable the GOT FEEDBACK button
Reduce the number of users synchronised from LDAP to JIRA
How do I reduce my user count in JIRA
How to Bulk Edit Groups in JIRA
How to enable "Attach Screenshot" on Linux Machine — When Attaching a Screenshot in JIRA, a user must go to More Actions and select the the Attach Screenshot link. However, on a Linux machine, this feature is not available.
How to Configure Workflow to allow Certain Group to Perform Certain Workflow Transition
How to Create a FishEye's Changeset or Crucible's Review Link on JIRA's Comment or Description —
How to make the 'Assignee' field required when 'Allow unassigned issues' is turn ON in General Configuration?
How to allow users to view sub-task only but not its parent issue
Why user does not get notification emails when he/she included in Notification Scheme?
How to change the Date Time Range Picker format to 24h format
What if you wish your Custom Field Appearing in Issue Navigator but not willing to set the context to All issue types
Why Fix/Affected Version sort differently in issue screen after picking more than one version?
Modify the gadget limit on a dashboard
How to import attachment using CSV
Log work optional view — You can have different view for Log Work, where there is the optional checkbox log work and the other showing the log work fields.
Receiving Daily Due Date Subscription
How to add comments to Excel export — A guide on how to have comment field in Excel export
How to change the Original Step in Workflow Transition — How to change the Original Step in Workflow Transition
How to Hide the "Configure Fields" Button on the Create Issue Screen
How to revert from New Issue Navigator layout to Old layout in JIRA 5.2.x
How to disable the Invite button in JIRA
How to display the issue type name in issue navigator column beside the image icon
How to Edit Bundled or System Plugins
How to auto disable compatibility mode in Internet Explorer
How to attach an attachment in a JIRA issue using REST API
How to restore a single attachment to an issue
Why not able to edit assignee in issue screen?
How do I turn off notification when new user is created using SOAP API
JIRA does not pick up the data after running an UPDATE or DELETE SQL statement
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 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
• 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 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 comment field to become mandatory in workflow transition
• 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 deactivate comments for closed issues
• 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 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
• How to update custom field values during workflow transition.
• 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

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
- 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
- Receiving 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
- 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 internal database which is used for evaluation purposes. 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

- How to Set Up SMTP Relay in Exchange 2007
- How to Use System JRE Instead of Embedded JRE
- Installation LDAP server on OpenDS and ApacheDS in JIRA (Windows Environment)
- 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 ClassNotNotFoundException
- Windows cannot find -Xms128m

How to Hide “Can’t access your account?” Link

How to display custom field of the sub-task in the parent issue screen?

How to change the font size for printing an issue

How to recover the comments viewable by a Project Role which has been deleted

How To Remove the Message 'Some issue types are unavailable due to incompatible field configuration and/or workflow associations.' from Edit Issue Screen

Also check out the JIRA Community Space and the Forums

Performance FAQ
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
- How to disable or enable the GOT FEEDBACK button
- Reduce the number of users synchronised from LDAP to JIRA
- How do I reduce my user count in JIRA
- How to Bulk Edit Groups in JIRA
- How to enable "Attach Screenshot" on Linux Machine — When Attaching a Screenshot in JIRA, a user must go to More Actions and select the the Attach Screenshot link. However, on a Linux machine, this feature is not available.
- How to Configure Workflow to allow Certain Group to Perform Certain Workflow Transition
- How to Create a FishEye's Changeset or Crucible's Review Link on JIRA's Comment or Description —
- How to make the 'Assignee' field required when 'Allow unassigned issues' is turn ON in General Configuration?
- How to allow users to view sub-task only but not its parent issue
- Why user does not get notification emails when he/she included in Notification Scheme?
- How to change the Date Time Range Picker format to 24h format
- What if you wish your Custom Field Appearing in Issue Navigator but not willing to set the context to All issue types
- Why Fix/Affected Version sort differently in issue screen after picking more than one version?
- Modify the gadget limit on a dashboard
- How to import attachment using CSV
- Log work optional view — You can have different view for Log Work, where there is the optional checkbox log work and the other showing the log work fields.
- Receiving Daily Due Date Subscription
- How to add comments to Excel export — A guide on how to have comment field in Excel export
- How to change the Original Step in Workflow Transition — How to change the Original Step in Workflow Transition
• How to Hide the "Configure Fields" Button on the Create Issue Screen
• How to revert from New Issue Navigator layout to Old layout in JIRA 5.2.x
• How to disable the Invite button in JIRA
• How to display the issue type name in issue navigator column beside the image icon
• How to Edit Bundled or System Plugins
• How to auto disable compatibility mode in Internet Explorer
• How to attach an attachment in a JIRA issue using REST API
• How to restore a single attachment to an issue
• Why not able to edit assignee in issue screen?
• How do I turn off notification when new user is created using SOAP API
• JIRA does not pick up the data after running an UPDATE or DELETE SQL statement
• 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 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
• 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 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 comment field to become mandatory in workflow transition
• 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 deactivate comments for closed issues
• 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 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
• How to update custom field values during workflow transition.
• 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
• Receiving 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
• 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
Modifying the JIRA Footer

Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

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 templates 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.

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($smallFooter) class="smallfooter" #else class="poweredbymessage" #end>
&nbsp;&nbsp;&nbsp; #if ($longFooterMessage)Powered by #end<a href="$externalLinkUtil.getProperty('external.link.jira.product.site')" class="smalltext">Atlassian JIRA</a>
&nbsp;&nbsp;&nbsp; the Professional <a href="$externalLinkUtil.getProperty('external.link.jira.product.site')">Issue Tracker</a>/a>&nbsp;&nbsp;&nbsp; - <a href="$externalLinkUtil.getProperty('external.link.atlassian.news')">Atlassian news</a>&nbsp;&nbsp;&nbsp; - <a href="${req.contextPath}/secure/Administrators.jspa">Contact Administrators</a></span></div>
```

Keep in mind the considerations about Modifying Jira Templates and JSPs. Restart JIRA after your customization in order for it to take effect.

'REDITED PAGES'

Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.
Changing the Size of the Text Area Custom Field

Removing NONE from the Issue Security Drop-Down List

Removing Commas for Values Held in Number Field Custom Field Type

Changing the Size of the Fix Versions and Affects Versions Select List

Changing the Size and Content of the Components Select List

'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. By default, Road Map, Change Log and Versions tab panels will be visible if at least a version is created on the project.

However, there are times that the Road Map, Change Log, and Versions project tab panels are still not visible despite having a version on the project. This is usually because the Fix version/s field is configured as hidden or the Road Map Panel, Change Log Panel, and Versions Panel plugin has been disabled in Project Panels Plugin.

For more information, please refer to Generating Reports and Specifying Field Behavior.

To enable the plugin modules, go to Administration > System > Plugins > Project Panels Plugin and enable the following:

- Road Map Panel
- Change Log Panel
- Versions Panel

This will enable these tab panels to appear on the Browse Project page.

Why Do Linked Issues in JIRA Appear with a Strike-Through

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.
```
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 Customizing Email Content.

How to Change the Number of Rows Allowed in the Text Type Custom Field Renderer

Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

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 templates 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

- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size and Content of the Components Select List

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 Behavior.

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

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](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.

This is not applicable for JIRA version 4.4 and above as the view_subtaskissues.jsp file no longer exist.
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.

⚠️ Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

**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`

ℹ️ 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 templates 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**

- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size and Content of the Components Select List

---

**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:


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.

Severity: Low
Article ID: JIRAKB104300816

http://support.atlassian.com/browse/JSP-15344

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, first, and last name of the users.

**MySQL**

```sql
select id, user_name, lower_first_name, lower_last_name into outfile '/tmp/jirausers.csv' FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\n' from cwd_user;
```

**PostgreSQL**

```sql
this will not include headers
```

Always back up your data before performing any modification to the database.
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.

How to disable or enable the GOT FEEDBACK button

In Early Access Program (EAP) releases, or in copies of JIRA running an evaluation license, JIRA displays 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. (However, Atlassian hopes you will pass this feedback on! Especially given that EAP releases are provided specifically so we can get early feedback on releases! And of course no one should be using Early Access Program releases in production.... Right?!)

To remove the 'GOT FEEDBACK' link:

1. Log in as a user with the JIRA System Administrators global permission.
2. Choose Add-ons. The 'Find add-ons' screen shows add-ons available via the Atlassian Marketplace. Choose Manage Add-ons to view the plugins currently installed on your JIRA site.
3. In the dropdown next to the Filter visible add-ons field, choose System.
4. Enter JIRA Feedback Plugin in the Filter visible add-ons field.
5. Click the JIRA Feedback Plugin row in the table to expand the entry and choose Disable.

Alternatively, you can remove it using the command line:

1. Shutdown JIRA.
2. Browse through JIRA Install folder.
3. Open setenv.bat inside the bin folder.
   - Back up this file, in case anything goes wrong.
   - If you are using Linux, then browse through the bin folder and open setenv.sh.
4. On set JVM_REQUIRED_ARGS= (after all the existing parameters) insert the following line:
   ```
   -Datlassian.disable.issue.collector=true
   ```
5. Save the file.
6. Restart JIRA

Note: If you are using JIRA OnDemand, you will not be able to disable this button. Please raise a support request and our support team can assist you with this (see Getting Help).

Reduce 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](https://confluence.atlassian.com/tdoa7). 
   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 fewer 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](https://confluence.atlassian.com/tdoa7). This is described in the following knowledge base article: [Unable to Create Issues Due to Exceeded License](https://confluence.atlassian.com/tdoa7).

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: [Reduce the number of users synchronised from LDAP to JIRA](https://confluence.atlassian.com/tdoa7). 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
- Reduce the number of users synchronised from LDAP to JIRA

## How to Bulk Edit Groups in JIRA

As in the [Managing Groups](https://confluence.atlassian.com/tdoa7) documentation it is possible to edit group membership directly from the group rather than through the User Browser. There is a system limitation on the number of users you can add to a group at a
time of 100, which limits the amount of users that can be edited at one time. It is possible to add more than 100 members into the group at once and they can be extracted from the database with SQL. This can be done with the below steps.

1. Identify the User Directory the users belong to:

   ```sql
   SELECT id, directory_name FROM cwd_directory
   ```

2. Select all the users from that directory:

   ```sql
   SELECT user_name FROM cwd_user where directory_id = <id from the previous query>
   ```

3. Comma separate them then copy and paste into the User Box:

   ![User Box]

   Press the button.

   Please see [JIRA-12844 - Ability to limit the number of results in a Search Request](https://jira.atlassian.com/browse/JIRA-12844) for further information on the 100 user limitation.

### How to enable "Attach Screenshot" on Linux Machine

When Attaching a Screenshot in JIRA, a user must go to More Actions and select the the Attach Screenshot link. However, on a Linux machine, this feature is not available.

**Note:** It is possible to enable this feature as per the below though this feature is not tested on Linux distributions and in hand is not supported by Atlassian Support.

1. Enable the screenshot applet on Linux

   To enable the "Attach Screenshot" feature on your JIRA instance, add the following line on the jira-config.properties file, which is located on the <JIRA_HOME> directory.

   ```properties
   jira-config.properties
   jira.screenshotapplet.linux.enabled=true
   ```

   In a new JIRA installations, jira-config.properties file may not initially exist and if so, you will need to create it manually. Create the jira-config.properties file inside <JIRA_HOME> folder.

2. Restart your JIRA instance

   Once you have modified the jira-config.properties file, restart your JIRA instance for the changes to take effect.
How to Configure Workflow to allow Certain Group to Perform Certain Workflow Transition

Steps in this page are using Scenario below, it might be vary depend on your setup
Only users from "Team Lead" group are allow to CLOSE an issue with Issue Type is Bug

Currently, JIRA is not able to set permission scheme based on issue type. It is an improvement request being filed in JIRA-5865 - Allow permission schemes to be configured per issue type (Resolved). So if we want to allow only certain users to perform a certain transition on specific issue type. It is not possible, however we can achieve this by using Issue Type Workflow Scheme.

For example, if we want to allow only Team Lead to close bug and improvement request in Project A, we can do the follow setup.

1. Create a new Workflow scheme by referring to document below:
   • [https://confluence.atlassian.com/display/JIRA051/Activating+Workflow#ActivatingWorkflow-Creatingaworkflowscheme](https://confluence.atlassian.com/display/JIRA051/Activating+Workflow#ActivatingWorkflow-Creatingaworkflowscheme)
2. Copy the current in-use project A’s workflow.
3. In the copied workflow, modify the "close issue" transition > under condition > add a new condition > only allow Team lead group to perform close issue transition. You can refer to page below to learn how to set condition to transition:
   • [https://confluence.atlassian.com/display/JIRA051/Configuring+Workflow#ConfiguringWorkflow-conditions](https://confluence.atlassian.com/display/JIRA051/Configuring+Workflow#ConfiguringWorkflow-conditions)
4. In the new workflow schemes, assign the modified workflow to the scheme by selecting the issue type - bug. Refer to page below for detail steps:
   • [https://confluence.atlassian.com/display/JIRA051/Activating+Workflow#ActivatingWorkflow-Assigningmultipleworkflowsatoaworkflowscheme](https://confluence.atlassian.com/display/JIRA051/Activating+Workflow#ActivatingWorkflow-Assigningmultipleworkflowsatoaworkflowscheme)
5. Repeat step above for improvement request and other issue type.
6. Associated the new workflow scheme to the project.

After that, only users from Team lead group/role are allow to close bug and improvement request, while other issue types are not affected.

How to Create a FishEye's Changeset or Crucible's Review Link on JIRA's Comment or Description

There are times when it is convenience to create a FishEye's changeset or Crucible's review link on a JIRA issue's field, such as on a comment or description. An example of this feature would be to write in a comment such as "I found something of interest in r234 that might explain this bug". The string r234 would then automatically be converted into a link to changeset 234 in FishEye.

In order to achieve this, you must:

1. Install the Application Links Link Rendering Plugin.
2. Create a trusted Authentication between JIRA and FishEye (Either Trusted or OAuth)
3. Configure the Comment and Description field to be Wiki Style Renderer

1. Install Application Links Link Rendering Plugin
   • Download the jar file from this link and placed it on <JIRA-HOME>\plugins\installed-plugins\ directory
   • You will need to restart your JIRA instance for the plugin to be installed

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
2. Create an Application Link between your JIRA and FishEye

- Create an Application link between JIRA and FishEye using OAuth Authentication or Trusted Application
- Make sure that you have created a project link between a Repository in FishEye and with a JIRA project

Do note that as of FishEye 2.9, JIRA does not need to create a project link to FishEye repo. However, in order for this feature to work, the project mapping must exists.

3. Make sure the renderer for the comment and description field are set to Wiki-Style Renderer

1. Go to the page "Administration >> Field Configurations"
2. Open the "Default Filed Configuration" or other field configuration associate to your project.
3. Locate the Comment field and make sure the renderer is set to Wiki Style Render
   - If not, change the render from "Default Text render" to "Wiki Style Render"

Did it work?

How to make the 'Assignee' field required when 'Allow unassigned issues' is turn ON in General Configuration?

The 'Assignee' field is required or automatically assigned by default when 'Allow unassigned issues' is turn OFF in Administration > General Configuration.

If it set as ON, user will be allow to choose the 'unassigned' option and this affect every project in JIRA. In certain use case, some project might need the Assignee field to be required and "unassign" is not an option.

In such case, there is not way to do it by default in JIRA, however this can be achieve by using a third party plugin call JIRA Suite Utilities, which has a set of conditions and validations to personalize workflow of JIRA.

To perform that:

1. Install the JIRA Suite Utilities. For install the plugin please refer to: Managing JIRA's Plugins#InstallingaJIRAPlugin
2. Use the 'Fields Required' validator function in your specified transition. Please refer to: https://studio.plugins.atlassian.com/wiki/display/JSUTIL/JIRA+Suite+Utilities+Workflow+Validators
3. Add the 'Assignee' field as required.

How to allow users to view sub-task only but not its parent issue

The subtask's security level is inherited from parents, so generally, if a users are able to view the subtasks, they are able to view the parent issue too and vise-versa.

How can we configure so that they are only able to view subtask?

This is usefull in certain use case where we want to assign subtask to users but we doesn't them to view the parent issue due to some restriction.
In this case, it is achievable by using the following steps:

1. Go to Administration > Issues > Fields > Custom fields > Add Custom field > 'User Picker' > 'set values' > Associate with your create sub-task screen > Update_. e.g. Add a custom field named "SubtaskUser".

2. Go to Administration > Issue Security Schemes > 'your current schemes' Operations > Security Levels > 'the level you wish to set' > Add > User Custom Field Value e.g. 'SubtaskUser' > Add

3. Create an sub-task issue by using the security level you just set with the custom field above.
   a. On the create sub-task page, you will see the 'SubtaskUser' custom field and you can choose the specific user who will be able to view the sub-task issue besides those already in the security level.
   b. If you wish to assign the sub-task to the user who is not include in the security level, you may also need to add the user in the 'SubtaskUser' field, otherwise, he/she will not be able to access. (Note: The assignee must have the 'Assignable' permission.)

Now the user will be able to view the sub-task, but not the parent issue.

**Why user does not get notification emails when he/she included in Notification Scheme?**

If an user is included in the Notification Scheme user groups/project roles, when he/she create an issue or comment on issue, he/she is supposed to get the notification emails.

If others in the same notification scheme all able to receive the notification emails but himself/herself, it means the SMTP email server is working properly. He/she also can get notification emails if other people comment on that issue.

In this case, you need to check on the user profile if the 'Preferences > My changes'

To view your JIRA user profile:
- Click your user name at the top-right of the browser window.

In 'My changes' you need to set 'Do not notify me' to 'Notify me'.

Note: This feature released since JIRA 5.0.3.

**How to change the Date Time Range Picker format to 24h format**

You need to make some Advanced JIRA configuration in the jira-config.properties or jira-application.properties file as following content:

<table>
<thead>
<tr>
<th>Preferred Date</th>
<th>Configuration for Date Picker</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/Dec/12 17:04</td>
<td>jira.date.picker/javascript.format=%e/%b/%y&lt;br/&gt;jira.date.time.picker/javascript.format=%e/%b/%y %H:%M&lt;br/&gt;jira.date.time.picker.java.format=dd/MM M/yy H:mm&lt;br/&gt;jira.date.picker.java.format=dd/MMM/yyyy</td>
</tr>
</tbody>
</table>
### 1.10.12 17:04

```java
jira.date.picker.java.format=d.MM.yy
jira.date.time.picker.java.format=d.MM.y %H:%M
jira.date.time.picker.javascript.format=%d.%m.%y %H:%M
jira.date.picker.javascript.format=%d.%m.%y
```

You can refer to the date picker format for different types of date and time formatting.

**Note**

- If you're using JIRA version prior to 4.4, you would need to make the changes in `$jira-install/atlassian-jira/WEB-INF/classes/jira-application.properties`
- If you're using JIRA version 4.4 and above, you just need to copy above code and create a .properites file named jira-config.properties and save it in the root directory of JIRA Home Directory. After that, restart your JIRA to reflect the changes. Otherwise, you can perform the similar settings under General Configuration without restart.

### What if you wish your Custom Field Appearing in Issue Navigator but not willing to set the context to All issue types

**‘Note’**

First of all, as mentioned in the KB: Custom Field Not Appearing in Issue Navigator Search Criteria, you HAVE to associate the custom field to ALL issue types. However this will cause other projects to have the unnecessary custom field in their project. There is workaround to hide the unnecessary custom field from these project so it won’t appear.

#### The Desire Result

- With this setup, issue navigator result will show the custom value in the right project even by searching all issue types.
- This custom field will not shown in other projects visually, ie. hide it.

#### Steps to do that

**Steps:**

1. Create a screen which does not contain the custom field which you don’t need. /Copy the current create issue screen and remove the custom field. Let's say 'TestScreen'
2. Copy/create a Screen Scheme and associate 'Create Issue' operation with the 'TestScreen'. Let's say 'Test Screen Scheme'
3. Copy/create an Issue Type Screen Scheme and associate those issue types(which you don’t want to have the custom field) with the 'Test Screen Scheme', and other issue types with the original screen scheme(which you are using currently) Let's say 'Test Issue Type Screen Scheme'
4. Go to the related projects and associate the project with the 'Test Screen Scheme'

#### Why Fix/Affected Version sort differently in issue screen after picking more than one version?

A user may notice that the versions on the issue screen are sorted "randomly". The versions are neither in ascending nor descending order.

**Reason Why the Versions Sort Differently**

The Versions are sorted according how the project versions are arranged or be scheduled.

**For Example:**

For screenshot below, you will notice there are all versions added in the project.
On the issue screen, when a user picks up all versions and it will arrange as the image below.

<table>
<thead>
<tr>
<th>Affects Version/s:</th>
<th>1.4, 1.0, 1.2, 1.1, 1.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution:</td>
<td>Unresolved</td>
</tr>
<tr>
<td>Fix Version/s:</td>
<td>1.4, 1.0, 1.2, 1.1, 1.3</td>
</tr>
<tr>
<td>Labels:</td>
<td>None</td>
</tr>
</tbody>
</table>

**Suggestion (for JIRA 5.x)**

Arrange from the Project Versions by drag and drop.

| 1.4 |
| 1.3 |
| 1.2 |
| 1.1 |
| 1.0 |

Refresh your issue screen.

<table>
<thead>
<tr>
<th>Affects Version/s:</th>
<th>1.0, 1.1, 1.2, 1.3, 1.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution:</td>
<td>Unresolved</td>
</tr>
<tr>
<td>Fix Version/s:</td>
<td>1.0, 1.1, 1.2, 1.3, 1.4</td>
</tr>
</tbody>
</table>

**Did it work?**

**Modify the gadget limit on a dashboard**

By default, the amount of gadget on a dashboard has been limit to 20 by default. If you want to change the limit, you can make it to your preferred number:

**JIRA 4.3.x and below:**

1. Proceed to $jira-install-path//atlassian-jira/WEB-INF/classes/jira-application.properties
2. Locate this line:

   ```
jira.dashboard.max.gadgets=20
   ```

3. Change the number to your prefer size
4. Restart JIRA
JIRA 4.4.x and above:

Please note that JIRA property no longer set in jira-application.properties since JIRA 4.4. Instead, you have to copy the key, paste it under jira-config.properties with the value. Please refer to this documentation for further information.

1. Edit or create a file named `jira-config.properties` if it doesn't exist at the root of Jira Home directory.
2. Insert the following (eg. set the limit to 30):

   ```
   jira.dashboard.max.gadgets=30
   ```

3. Save the file
4. Restart JIRA to reflect the changes.

How to import attachment using CSV

CSV import commonly used for importing mass of issues and some issues comes with attachments. In JIRA version 5.0 and earlier, it is not possible to use FILE protocol to include the attachment. HTTP protocol is the only choice to import attachment to JIRA through CSV.

JIRA 5.0 and earlier have to use HTTP protocol for CSV import, however, JIRA 5.1 and above can use FILE protocol for CSV import.

File Protocol

1. Proceed to your JIRA Home Directory attachments directory (eg. `<JIRA Home>/import/attachments/`)
2. Create a folder named csvimport and save your attachment within it. In the below example, a test.JPG image file was added to the csvimport directory:

   Example
   
   `<JIRA_Home>/import/attachments/csvimport/test.JPG`

3. Specify the file path within the CSV file.

   CSV file

   `Assignee, Summary, Description, Attachment, Comment`

   Admin, "Issue demonstrating the CSV attachment import", "Please check the attached image below.", file://csvimport/test.JPG, "01/01/2012 10:10;Admin; This comment works"

4. Proceed with using the CSV file import wizard.

HTTP Protocol

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
1. Save the attachment (eg. filename.png) inside the following directory `<JIRA_installation>/atlassian-jira/folder/`
   
   Example:
   ```
   <JIRA_Installation>/atlassian-jira/attach/filename.png
   ```

2. Specify the URL of the 'Attachment' column within your CSV file.
   
   Example:
   ```
   CSV file
   Assignee, Summary, Description, Attachment, Comment
   Admin, "Issue demonstrating the CSV attachment import", "Please check the attached image below.", http://jira-server:8080/attach/filename.png, "01/01/2012 10:10;Admin; This comment works"
   ```

3. Proceed with using the CSV file import wizard.

**Log work optional view**

You can have different view for Log Work, where there is the optional checkbox log work and the other showing the log work fields.

1. To have the Log Work checkbox

To show the Log work checkbox as following:

Set the transition screen to have both Log Work and Time Tracking fields.

2. To have the Log Work showing the fields

To show the Log work fields as following:
Set the transition screen to have only Log Work and not Time Tracking field.

3. If you are facing issue with Log Work checkbox is only showing after work has been logged, even after you have followed the above.

This is because Time Tracking Legacy Mode is turned ON, please turn OFF Legacy Mode by Deactivating Time Tracking in System > Issue Features > Time Tracking and activating it again with untick the Legacy Mode.

For more information relating to Log Work and Time Tracking, please refer to the Logging Work documentation.

Receiving Daily Due Date Subscription

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

1. Set up your search criteria

To find all issues that have due date before next day, use the following Advanced Search query:

```
duedate < "1d"
```

Otherwise, to find all issues that had overdue for one day, use the following Advanced Search query:

```
due <= "-1d"
```
For example, to find all issues in the "ACME" project that will be due in the next day, use the following Advanced Search query:

```
project = "ACME" and duedate < "1d"
```

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 Due Date Issues"), then click the 'Save' button.

- For further details, please see Saving Searches ('Issue Filters').

3. **Subscribe to your Filter**

Once you have 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.

**How to add comments to Excel export**

A guide on how to have comment field in Excel export

You can get comments in an issue by using the Word export or subscribing RSS feeds. However, you are not able to do so when doing an Excel export. To make this possible, you will require some plugins to get it work.

1. **Download and install plugin**

   - If you wish to have only the last comment to be included in the Excel export, you can use [Last Comment Plugin](#).

     - The latest version only compatible up to JIRA 4.4

   - If you wish to have all the comments to be included in the Excel export, you can use [All comments plugin](#).

     - The latest version only compatible up to JIRA 5.1
     - All comments plugin has been tested in JIRA 5.2 and it is working, however the comment that is being imported is not rendered correctly in the Excel view.

2. After the install, you will require to add Custom Field that comes with the plugin.

3. Include the custom field in the Issue Navigator.

4. Do an Excel export.

   A New Feature request can be found in

   [JIRA-8426 - View comments as a field (eg. in Excel view)](#).
How to change the Original Step in Workflow Transition

How to change the Original Step in Workflow Transition

When you have already set up the workflow transition for the particular step correctly and realise you have set it up on a wrong Step by mistake. You can change the Destination Step for the workflow transition but you are not able to change the Original Step through JIRA User Interface.

There are two workaround to this:

1. Change the name of the Original Step Name which linked to the particular Workflow transition
2. Do a simple hack in Workflow XML by doing the following

   - View desired workflow
   - Download workflow XML
   - Open it using text editor preferably Notepad++
   - Edit the transition from one step to another, for example:
<action id="701" name="Close Issue" view="commentassign">
<meta name="opsbar-sequence">60</meta>
<meta name="jira.il8n.submit">closeissue.close</meta>
<meta name="jira.il8n.description">closeissue.desc</meta>
<meta name="jira.il8n.title">closeissue.title</meta>
<meta name="jira.description">Closing an issue indicates there is no more work to be done on it, and it has been verified as complete.</meta>
<restrict-to>
<condition type="class">
<arg name="class.name">com.atlassian.jira.workflow.condition.PermissionCondition</arg>
</condition>
</restrict-to>
<results>
<unconditional-result old-status="Finished" status="Closed" step="6">
<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="eventTypeId">5</arg>
</function>
</post-functions>
</unconditional-result>
</results>
</action>

You will need to cut the code from `<action id=> to </action>`, and paste it to your desire Steps (step id=) inside the `<actions></actions>`

- Save the file and do Import from XML in Issues > Workflows

**How to Hide the "Configure Fields" Button on the Create Issue Screen**

⚠️ Atlassian does not support customizations to JIRA. For more information about Atlassian support, see Atlassian Support Offerings.

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
By default, during issue creation and issue edit, the “Configure Fields” button will be shown on the top right corner of the pop up (refer to the screen capture below). Some administrator would prefer to hide the button to prevent the users from changing the pre-configured fields.

⚠️ If a user has a custom view when the javascript is implemented, they will not be able to change their view until the customization is removed.

**Hiding Jira "Configure Fields" Button without restart**

1. For the javascript to run, you will need to paste the coding on the description area of the field since it supports html codings. To do this, actually you have two choices, you can either create a custom field or utilizing a default field on the screen.
2. Similar steps are applicable to both default field or custom field after you created a custom field. However do note that, if you used a custom field, the custom field will be visible to the users and hiding it from Field Configuration will disable the javascript pasted on its description.
3. Navigate to Administration > Issues > Fields > Fields Configurations > Configure to view the list of fields.
4. Choose the field that you wanted to paste the codes on (recommended to paste on Description Field or Comment Field since both of them is without pre-defined descriptions on them).
5. Click on edit to show the Field Description.
6. Paste the codes provided below to the Description field and Update the field.

```html
<script type="text/javascript">
(function($) {
  AJS.toInit(function(){
    // init on load
    AJS.$("#qf-field-picker-trigger").hide();
  });
  JIRA.bind(JIRA.Events.NEW_CONTENT_ADDED, function (e, context) {
    // init on refresh
    AJS.$("#qf-field-picker-trigger").hide();
  });
})(AJS$);
</script>
```

7. After doing this, you can verify the result by trying to create an issue, the button "Configure Fields" should not be visible anymore. (Refer screen capture below)

**How to revert from New Issue Navigator layout to Old layout in JIRA 5.2.x**

⚠️ The content on this page relates to changes which are not recommended or 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.

After upgrade to JIRA 5.2.x from version prior to 5.2, there will be the new layout of issue navigator. This is a procedure to disable this new issue navigator for the entire JIRA instance.
This will not work in versions after 5.2

To do the following, you have to disable some modules in the system plugin:
1. Go to Administration > Add-ons > Manage Add-ons
2. Under System Plugins section, Click on the Show System Plugins
3. Search for plugins named Atlassian JIRA - Plugins - Issue Navigation
4. Click on the plugin and expand the modules
5. In the module lists, disable the following 3 modules:
   - IssueNav kickass-rewrite(kickass-rewrite)
   - Quicksearch kickass-rewrite(quicksearch-rewrite)
   - Clear sorts kickass-rewrite(clearsorts-rewrite)

How to disable the Invite button in JIRA

Unable to render {include} The included page could not be found.

Please use the Documentation Style Guidelines

<INSERT INTRODUCTORY BLURB>

On this page:
- <INSERT HEADING>
- <INSERT HEADING>
- Notes

Related pages:
- <INSERT LINK>
- <INSERT LINK>

<INSERT HEADING>

<INSERT SECTION-SPECIFIC BLURB IF REQUIRED>

Before you begin:
- <prerequisite>
- <prerequisite>
- <prerequisite>

To do something:
1. Step 1 (If you are describing navigation, use the convention 'Control1' > 'Control 2' > etc)
2. Step 2
3. Step 3

Screenshot: <INSERT SCREENSHOT TITLE>
!<INSERT SCREENSHOT FILE NAME>|border=1,bordercolor=grey!

<INSERT HEADING>

<INSERT SECTION-SPECIFIC BLURB IF REQUIRED>

Before you begin:
- <prerequisite>
- <prerequisite>
- <prerequisite>

To do something:
1. Step 1
JIRA 6.1 Documentation

2. Step 2
3. Step 3
Screenshot: <INSERT SCREENSHOT TITLE>
!<INSERT SCREENSHOT FILE NAME>|border=1,bordercolor=grey!
Notes

<INSERT NOTE>
<INSERT NOTE>

How to display the issue type name in issue navigator column beside the image icon
To achieve this use case, you need to modify the following velocity file <installation
directory>\atlassian-jira\WEB-INF\classes\templates\jira\issue\field\issuetype-columnview.vm.
As an example, which tested in JIRA 5.2.11 and 4.4:
For your information
Please backup the original value vm file first before process any changes, which can allow you to easily
roll back. Atlassian does not support customisations to Velocity templates or other JIRA files. For more
information about Atlassian support, see Atlassian Support Offerings.
The original value of the file in JIRA 5.2.11

issuetype-columnview.vm
##disable_html_escaping() TODO REENABLE
#if (!($displayParams && $displayParams.get('nolink')))
<a href="${requestContext.baseUrl}/browse/${issue.getKey()}">#end #if
($displayParams.textOnly)$textutils.htmlEncode($!issuetype.nameTranslation,
false)#else #set ($iconurl = $issuetype.iconUrlHtml)#if
($iconurl.startsWith('http://') || $iconurl.startsWith('https://'))<img
src="$iconurl" height="16" width="16" border="0" align="absmiddle"
alt="$textutils.htmlEncode($issuetype.getNameTranslation(), false)"
title="$textutils.htmlEncode($issuetype.getNameTranslation(), false) $textutils.htmlEncode($!issuetype.getDescTranslation(), false)">#else<img
src="${requestContext.baseUrl}${iconurl}" height="16" width="16" border="0"
align="absmiddle" alt="$textutils.htmlEncode($issuetype.getNameTranslation(),
false)" title="$textutils.htmlEncode($issuetype.getNameTranslation(), false) $textutils.htmlEncode($!issuetype.getDescTranslation(), false)">#end#end#if
(!($displayParams && $displayParams.get('nolink')))</a>#end

The modified issuetype-columnview.vm file

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.

1087


Please restart JIRA after the changes have been made.

The above modified .vm file can demonstrate the following effect as an example:

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 templates 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.

How to Edit Bundled or System Plugins

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

This is a guide on how to edit the bundled or system plugins in JIRA. It is highly not recommended unless necessary as it may corrupt the plugins.

Modifying Contents of Bundled Plugins

⚠️ Please note that this will only work if the bundled plugin has not been upgraded/modified.

Before you begin:

- **Backup** `atlassian-bundled-plugins.zip` file in `<jira-install>/atlassian-jira/WEB-INF/classes`.

To modify its content:

1. Unzip the `atlassian-bundled-plugins.zip` file.
2. Replace customized plugin jar in place of the existing jar file. You can either build the customized jar file from its source or just extract the jar, modify the files and create it back.
3. Zip all the jars back with the same name: `atlassian-bundled-plugins.zip`
4. Restart JIRA. It should take effect now.

How to auto disable compatibility mode in Internet Explorer

⚠️ 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.

The following workarounds were provided by other users who facing the same problem. Please always generate a backup of the file before any customization to ensure a revert is possible.

JIRA 4.4.x:

1. Open the file `<JIRA_INSTALL>/atlassian-jira/include/decorators/header.jsp` and `<JIRA_INSTALL>/atlassian-jira/include/decorators/header-deprecated.jsp`
2. Make a backup of `header.jsp` and `header-deprecated.jsp`
3. In the head tag of both files, add in the meta tag code from:

```
<head>
  ...
</head>
```

   to:

```
<head>
  <meta http-equiv="X-UA-Compatible" content="IE=7,IE=8,IE=9" />
  ...
</head>
```

4. Restart JIRA
JIRA 6.x:

1. Open the file `<JIRA_INSTALL>/atlassian-jira/decorators/general.jsp`
2. Make a backup of general.jsp
3. In the head tag, change the following code from:

   ```html
   <head>
   <script type="text/javascript">
   if (!window.performance) {
       window.mpPerformance = {}; 
       window.mpPerformance.responseStart = new Date().getTime(); 
   }
   </script>
   <%@ include file="/includes/decorators/aui-layout/head-common.jsp" %>
   <%@ include file="/includes/decorators/aui-layout/head-resources.jsp" %>
   </head>
   ```

   to:

   ```html
   <head>
   <%@ include file="/includes/decorators/aui-layout/head-common.jsp" %>
   <%@ include file="/includes/decorators/aui-layout/head-resources.jsp" %>
   <script type="text/javascript">
   if (!window.performance) {
       window.mpPerformance = {}; 
       window.mpPerformance.responseStart = new Date().getTime(); 
   }
   </script>
   </head>
   ```

4. Restart JIRA

### How to attach an attachment in a JIRA issue using REST API

Upload the original_image.jpg using the following REST API code:

```bash
curl -D- -u {username}:{password} -X POST -H "X-Atlassian-Token: nocheck" -F "file=@{path/to/image}"
http://{base-url}/rest/api/2/issue/{issue-key}/attachments
```

For more information, please review our JIRA REST API code: https://docs.atlassian.com/jira/REST/latest/#idp2011456

**At the moment, JIRA do not have the essence of encoding binary data.**

### How to restore a single attachment to an issue

This guide will explain on how to restore a single attachment to an issue. Let say a user has accidentally deleted an important attachment from an issue, and you have a backup `/data/attachments/` directory elsewhere.

Since JIRA 4.2, there were Changes to Attachment Creation so that JIRA now saves the attachment using an
attachment ID instead of the attachment file name.

On this page:
- Restoring a single attachment

Related pages:
- Configuring File Attachments
- Attaching a File

Restoring a single attachment

In order to restore the single attachment you need the following:

- A backup of /data/attachments/ directory
- The file type of the attachment that is accidentally deleted

To restore the attachment:

1. Go to the backup /data/attachments/ directory
2. Open the attachments directory down to the folder of the issue key (Eg. /data/attachments/Demo/Demo-1/)
3. Compare the IDs in the directory with the existing ID from an attachment URL in the JIRA issue. For example the attachment ID below is 667983

   http://your-jira/secure/attachment/667983/EditProjectRoles.png

Since you have deleted the attachment, it won't exist in the issue. However, you can refer to this step to identify the missing attachment ID with the attachments in the backup directory.

4. (Assume the deleted attachment ID is 667980) Copy the file 667980 to a temp directory
5. Rename the copied file 667980 to 667980.png (Assuming the attachment that was deleted is a PNG file type)
6. Upload the file 667980.png to the issue referring to Attaching a File to an Issue

Why not able to edit assignee in issue screen?

If an assignee field is not associate with the screen scheme, then assignee field will not be edited. If you want to edit the assignee field, please go to project issue type screen and make sure the assignee fields is there.

If the assignee field is appear in edit screen, you may need to check the following information:

- Does the user has the Assign Issues permission?
- Does the assignee has the Assignable Issues permission?
- Try use Permission helper via System -> Add-ons -> Admin Helper. (Permission Helper looks at each condition which could determine whether the user has permission or not, and offers advice on how to change the settings.)
- If you need further help, please file a ticket to our Atlassian Support System.

How do I turn off notification when new user is created using SOAP API

Currently, JIRA do not have the control of turning off the notification being sent out after create a new user using SOAP API. Moreover, SOAP API is deprecated as of JIRA 6.0. For more information, please click on this documentation.

JIRA does not pick up the data after running an UPDATE or DELETE SQL statement

There is a lot of data not updated in JIRA when running an UPDATE or DELETE SQL statement. The reason is that the JIRA cache does not update the data in memory dynamically. After restarting JIRA the cache of the database values will be updated.
Please backup first before you proceed when running any UPDATE or DELETE statement in the JIRA database.

Adding custom content to the front page

Custom HTML content can be easily added to the dashboard by a JIRA administrator.

For example, to customize 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 customized (e.g. add your organization's logo and/or preferred color 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

Support requests are often resolved significantly faster if a data export is provided as it will allow our legendary supporters direct access to a copy of your instance. We understand that sometimes this may be a difficult option due to the sensitivity of your data and have written an anonymising tool to handle this particular scenario.

Anonymising JIRA Data:

The JIRA inbuilt backup functionality will produce a ZIP file containing either 1 or 2 XML files, depending on the version that is being used. These files are a copy of the entire contents of JIRA's database, encoded in XML, that can be used to restore an instance - we have further detail on this in our Automating JIRA Backups docu
ntation.

As of JIRA 4.4, the backup functionality will produce a ZIP file that contains 2 XML files. These files will be `activeobjects.xml` and `entities.xml`. Only `entities.xml` will need to be anonymised - please do not attempt to anonymise the `activeobjects.xml`. For versions prior to 4.4, only one XML file will be produced with the same naming convention as the ZIP it is compressed as (for example `1970-Jan-01--0001.zip` will expand to `1970-Jan-01-0001.xml`).

1. Ensure that the `JAVA_HOME` variable has been configured, as in our Setting JAVA_HOME documentation.
2. Download the JIRA Anonymiser.
3. Create a temporary directory.
4. Unzip the anonymizer in the temporary directory.
5. Unzip the JIRA backup ZIP file (for example `1970-Jan-01--0001.zip`) in the temporary directory.
6. Anonymise the backup file with the below commands:

   ```bash
   $ java -Xmx512m -jar joost.jar <JIRA BACKUP>.xml anon.stx > <NAME OF ANONYMISED BACKUP>.xml
   ```

   For example, this would be anonymising a JIRA backup with the naming convention from JIRA 4.4+:

   ```bash
   $ java -Xmx512m -jar joost.jar entities.xml anon.stx > anon-entities.xml
   ```

7. Depending on the size of the backup, additional memory may need to be allocated to the JVM. In order to do this, increase the value of the `Xmx` in increments of `128m`.
8. Compress the generated anonymised XML backup file (e.g: `anon-entities.xml`) and the `activeobjects.xml` (`JIRA 4.4.x + only`) into a ZIP or tarball.
9. Attach that ZIP or tarball onto the support issues as raised on support.atlassian.com.
10. The temporary directory can now be removed.

The screenshot below is a simple example of how it is run in the command prompt of Windows XP:

![Screenshot of command prompt](image)

Information about the Anonymiser:

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.

Please check the anonymised backup, `anon-backup.xml`, to ensure it's clean enough for the needs of your organisation before sending it to Atlassian.
Problems:

**Invalid XML Characters**

If, when the anonymiser runs, an error indicates that there are invalid XML characters in the XML backup of the database, run our utility to remove invalid XML characters first before anonymising.

**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.

1. Modify AbstractCommentHandler.java by adding 6 more lines after line 81, after the code block under `if (body != null):

   ```java
   // 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:

**No longer works starting JIRA 4.4**

This Javascript no longer works starting from JIRA 4.4. It only works up until JIRA 4.3.
Your timezone. The value is set automatically from your browser, so please only modify if it's incorrect. <br>
&lt;a href="http://www.worldtimezone.com/* target="timezone">World Timezones</a&gt;&lt;/b&gt;

<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 JIRA-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 color 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.

#### Before you begin

> Please note that editing the database or any XML backup is not supported by Atlassian.

Always **back up** your data before performing any modifications to the database. If possible, try your modifications on a test server.

#### Migrating custom field content manually

One workaround is to use bulk edit operations to migrate content:

> Please note that this workaround is only applicable when you have many issues containing the same custom field value. As a scenario let's say you have 100 issues, and each issue have different value for the custom field, it will be very difficult for you to perform the migration by using the steps below.

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

> There have been changes made to the structure of several custom fields which may have caused this article to be outdated. With that, the below *instructions may not work for JIRA versions 4.4+*

More details on the field changes found here: [Plugin Developer Notes for JIRA 4.4#Single-andMulti-SelectCustomFieldChanges](#)

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.

<table>
<thead>
<tr>
<th>Custom Field Type</th>
<th>Type Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Version</td>
<td>com.atlassian.jira.plugin.system.customfieldtypes:version</td>
</tr>
<tr>
<td>Multi Version</td>
<td>com.atlassian.jira.plugin.system.customfieldtypes:multiversion</td>
</tr>
<tr>
<td>Single Select</td>
<td>com.atlassian.jira.plugin.system.customfieldtypes:select</td>
</tr>
<tr>
<td>Multi Select</td>
<td>com.atlassian.jira.plugin.system.customfieldtypes:multiselect</td>
</tr>
<tr>
<td>Multi User</td>
<td>com.atlassian.jira.plugin.system.customfieldtypes:multiuserpicker</td>
</tr>
</tbody>
</table>

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 (user
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.

```sql
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:

```sql
select * from customfieldvalue where id= (select id from customfield where cfname='multisel3') and stringvalue not in (select username from userbase);
```

Then you can change the custom field type:

```sql
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:

```sql
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:

```sql
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.vm`
- `atlassian-jira/WEB-INF/classes/templates/plugins/searchrequestviews/searchrequest-excel-footer.vm`
- `atlassian-jira/WEB-INF/classes/templates/plugins/searchrequestviews/searchrequest-description-header.vm`

Refer to Microsoft® Office HTML and XML Reference for further information on the syntax of the template contents.

### Severity

<table>
<thead>
<tr>
<th>Status</th>
<th>Article ID: JIRAKB148766913</th>
</tr>
</thead>
</table>

http://support.atlassian.com/browse/JSP-20921

**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 5 hours (or 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 element you want to edit in the `web.xml` file is:

```xml
<web-app ...>
  ...
  <session-config>
    <session-timeout>30</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**

Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

**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 templates into the new installation of JIRA. If the JSPs or templates have changed in the newer version, you will have to port your
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>` needs to be placed in the wanted `<issue-tabpanel>` tag.

### Changing the Due Date Input Format

The [Look and Feel page](https://confluence.atlassian.com/jirahelp-look-and-feel) allows you to customize the way times and dates are presented to users throughout the JIRA user interface.

When specifying dates and times, they should be based on the [Java SimpleDateFormat](https://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html).

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.

#### Relative time is used in date/time formats

Issue date/time fields show a relative instead of absolute date/time format (for example: Yesterday 12:00 PM instead of 20 May 2013 12:00 PM). You can still see the absolute date/time by hovering over the field.

*The date/time format reverts to absolute after a week.*

If you want to switch off this format, set the `jira.lf.date.relativize` application property to 'false'. See [Advanced JIRA Configuration](https://confluence.atlassian.com/jirahelp-look-and-feel) for more information.

#### Configuring date picker formats

JIRA system administrators can configure the format of date pickers used throughout the JIRA user interface via options on the [Advanced Settings page](https://confluence.atlassian.com/jirahelp-look-and-feel).

Be aware that these options are different from the [Date/Time Formats](https://confluence.atlassian.com/jirahelp-look-and-feel) configuration options on the [Look and Feel page](https://confluence.atlassian.com/jirahelp-look-and-feel), which only customize JIRA's presentation of times and dates to users.

The date or date/time formats for date pickers are defined by a pair of properties (one for Java and the other for JavaScript). The two properties in this Java/JavaScript pair must match in order for the date (or date/time) picker they define to function correctly.

- For Java formats, specify date/time formats based on the [Java SimpleDateFormat](https://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html).
- For JavaScript formats, specify date/time formats based on the [Unix date format](https://en.wikipedia.org/wiki/Unix_time).

Here are some example US-based date configurations:

<table>
<thead>
<tr>
<th>Preferred Date</th>
<th>Value of the <code>jira.datel timepicker.java.format</code> property</th>
<th>Value of the <code>jira.datel timepicker.javascritp.format</code> property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-10-01</td>
<td>yyyy-MM-dd</td>
<td>%Y-%m-%d</td>
<td>ISO 8601 format</td>
</tr>
<tr>
<td>Oct/1/10</td>
<td>MMM/d/yy</td>
<td>%b/%e/%y</td>
<td></td>
</tr>
<tr>
<td>10/01/10</td>
<td>MM/dd/yy</td>
<td>%m/%d/%y</td>
<td></td>
</tr>
<tr>
<td>Oct 1, 2010</td>
<td>MMM d, yyyy</td>
<td>%b %e, %Y</td>
<td></td>
</tr>
<tr>
<td>10/01/2010</td>
<td>MM/dd/yyyy</td>
<td>%m/%d/%Y</td>
<td></td>
</tr>
</tbody>
</table>

Here are some examples of date/time configurations:

<table>
<thead>
<tr>
<th>Preferred Date/Time</th>
<th>Value of the <code>jira.datetimepicker.java.format</code> property</th>
<th>Value of the <code>jira.datetimepicker.javascritp.format</code> property</th>
<th>Comments</th>
</tr>
</thead>
</table>
Changing the Project Key

⚠️ You were meant to be redirected to Editing a Project Key, but something went wrong. Please use site search to find the information you were after. If you can’t find it, let us know.

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.

⚠️ Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

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 templates 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:

```xml
<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):

```xml
<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:

```xml
```
>\$\text{htmlEncode(}\$\text{component.getString('name'))}</option>

to:

>\$\text{htmlEncode(}\$\text{component.getString('name')) -$}
\$\text{htmlEncode(}\$\text{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**

- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size and Content of the Components Select List

**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.

Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see [Atlassian Support Offerings](https://www.atlassian.com/software/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 templates 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):
Make sure to back up the velocity file before changing it. Keep in mind the notes from Modifying JIRA Templates and JSPs.

### RELATED PAGES
- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size and Content of the Components Select List

### Changing the Size of the Text Area Custom Field

⚠️ Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

#### 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 templates 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.
There is a feature request to allow this customisation from within JIRA at [JRA-20248](https://jira.atlassian.com/browse/JRA-20248).

**Changing the Temporary Directory**

To move the temp directory, edit `<JIRA>/bin/catalina.sh`:

```bash
if [ -z "$CATALINA_TMPDIR" ] ; then
    # Define the java.io.tmpdir to use for Catalina
    CATALINA_TMPDIR="$CATALINA_BASE"/temp
fi
```

Replace the ""/temp with your temporary file directory.

**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
1. Creating user groups

   1. Create a new user group say `Group_External` (Administration -> Users & Groups -> Group Browser)
   2. Add the External Users to that group
   3. 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.

![Group Settings](image)

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 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.
Connecting to SSL services

The content on this page relates to platforms which are not supported by JIRA. Consequently, Atlassian cannot 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 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 or Integrating JIRA with Apache using SSL.

Add SSL Certificates automatically!
We now have a JIRA SSL Atlassian Labs plugin for this process. Please install and use the plugin before going through these docs.

On this page:
- Problem Symptoms
- The Cause
- Resolution
  - Obtain and Import the Server's Public Certificate
  - Alternative KeyStore Locations
  - Debugging

Problem Symptoms

Attempting to access URLs that are encrypted with SSL (for example HTTPS, LDAPS, IMAPS) throws an exception and JIRA refuses to connect to it. For example:

```
javax.net.ssl.SSLHandshakeException: PKIX path building failed: 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)
```

This is the same as the following error that's generated in Chrome when visiting a page that's encrypted with a self-signed certificate, except Java can't "Proceed anyway", it just refuses the certificate:
The Cause

Whenever JIRA attempts to connect to another application over SSL (e.g.: HTTPS, IMAPS, LDAPS), it will only be able to connect to that application if it can trust it. The way trust is handled in the Java world (this is what JIRA is written in) is that you have a keystore (typically `$JAVA_HOME/lib/security/cacerts`) or also known as the trust store. This contains a list of all the known CA certificates and Java will only trust certificates that are signed by those CA certificate or public certificates that exist within that keystore. For example, if we look at the certificate for Atlassian:

![Certificate Chain](image)

We can see the `*.atlassian.com` certificate has been signed by the intermediate certificates, DigiCert High Assurance EV Root CA and DigiCert High Assurance CA-3. These intermediate certificates have been signed by the root Entrust.net Secure Server CA. Those three certificates combined are referred to as the certificate chain. As all of those CA certificates are within the Java keystore (`cacerts`), Java will trust any certificates signed by them (in this case, `*.atlassian.com`). Alternatively, if the `*.atlassian.com` certificate was in the keystore, Java would also trust that site.

This problem comes from a certificate that is either self-signed (a CA did not sign it) or the certificate chain does not exist within the Java keystore. Subsequently, JIRA doesn't trust the certificate and fails to connect to the application.

Resolution

In order to resolve this, the public certificate need to be imported in the Java keystore that JIRA uses. In the example above, this is `*.atlassian.com` and we cover how to install it below.

Obtain and Import the Server's Public Certificate

1. Download and install the Portecle app onto the server that runs JIRA.  
   ![Portecle](image)
   This is a third-party application and not supported by Atlassian.
2. Ensure the `<JAVA_HOME>` variable is pointing to the same version of Java that JIRA uses. See our Setting JAVA_HOME docs for further information on this.
   ![Setting JAVA_HOME](image)
   If running on a Linux/UNIX server, X11 will need to be forwarded when connecting to the server (so you can use the GUI), as below:
   ```
   ssh -X user@server
   ```
3. Select the Examine menu and then click Examine SSL/TLS Connection:
4. Enter the SSL Host and Port of the target system:
5. Wait for it to load, then select the public certificate and click on PEM:


7. Go back to the main screen and select the **Open an existing keystore from disk** option, then enter the password (the default is changeit).
8. Select the **Import a trusted certificate into the loaded keystore** button:

9. Select the certificate that was saved in step 6 and confirm that you trust it, giving it an appropriate alias (e.g.: confluence).
   a. You may hit this error:
b. If so, hit OK, and then accept the certificate as trusted.

10. Save the Key Store to disk:

11. Restart JIRA.

12. Test that you can connect to the host.

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.

However, setting this is **not recommended** because if Java is told to use a custom keystore (e.g. containing a self-signed certificate), then Java will not have access to the root certificates of signing authorities found in `$JAVA_HOME/jre/../sun/x509/cacerts`.

---

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
A_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).

Debugging

Problems are typically one of two forms:

- The certificate was installed into the incorrect keystore.
- The keystore does not contain the certificate of the SSL service you're connecting to.

See Also

- Configuring an SSL Connection to Active Directory
- Running JIRA over SSL or HTTPS
- Integrating JIRA with Apache using SSL

Creating an Unassigned Issue

All new issues are unassigned by default. If you wish to change this:

- Go to the General Configuration page of the Administration section. Now simply edit the configuration and turn OFF the 'Allow Unassigned Issues' flag. For more detail please refer to the documentation relating to this function.

By default, new projects also have their 'Default Assignee' set to 'Unassigned.' This is located in the Roles section and can be changed there.

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 these 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:

```html
<a href="[JIRA BASE URL]/secure/CreateIssueDetails!init.jspa?[ARGUMENTS]">[DESCRIPTION]</a>
```

where

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[JIRA BASE URL]</td>
<td>The Base URL of the JIRA you wish to create issues in</td>
<td><a href="http://jira.atlassian.com">http://jira.atlassian.com</a></td>
</tr>
<tr>
<td>[ARGUMENTS]</td>
<td>List of key value pairs separated by '&amp;' which represent the field and its</td>
<td>pid=10420&amp;issuetype=4</td>
</tr>
<tr>
<td></td>
<td>value to be set in the create issue screen</td>
<td></td>
</tr>
<tr>
<td>[DESCRIPTION]</td>
<td>The link description visible to users</td>
<td>create issue in Test Project</td>
</tr>
</tbody>
</table>

**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 that 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.

```html
...<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

<table>
<thead>
<tr>
<th>Display Name</th>
<th>Key</th>
<th>Value Type</th>
<th>Value Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>pid</td>
<td>Project Id</td>
<td>'10420'</td>
</tr>
<tr>
<td>Issue Type</td>
<td>issuetype</td>
<td>Issue Type Id</td>
<td>standard JIRA issue type values range from '1' to '4'</td>
</tr>
<tr>
<td>Summary</td>
<td>summary</td>
<td>Plain Text</td>
<td>'issue+created%20via+link'</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Priority</td>
<td>priority</td>
<td>Priority Id values range from '1' to '5'</td>
<td></td>
</tr>
<tr>
<td>Due Date</td>
<td>duedate</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>Components</td>
<td>components</td>
<td>Component Id '10014'</td>
<td></td>
</tr>
<tr>
<td>Affects Version/s</td>
<td>versions</td>
<td>Version Id '10015'</td>
<td></td>
</tr>
<tr>
<td>Fix Version/s:</td>
<td>fixVersions</td>
<td>Version Id '10015'</td>
<td></td>
</tr>
<tr>
<td>Assign To</td>
<td>assignee</td>
<td>Username 'admin' or '<a href="mailto:sam@atlassian.com">sam@atlassian.com</a>'</td>
<td></td>
</tr>
<tr>
<td>Reporter</td>
<td>reporter</td>
<td>Username 'admin' or '<a href="mailto:sam@atlassian.com">sam@atlassian.com</a>'</td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td>environment</td>
<td>Plain Text 'this+is+the+environment'</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>description</td>
<td>Plain Text 'this+is+the+description'</td>
<td></td>
</tr>
</tbody>
</table>

**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**

```html
To create an improvement issue in the Test project, click here
```

**Output**

To create an improvement issue in the Test project, click here
To create a task with summary 'say hello world', click here

To create a task with multiple values selected for a field, click here

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 are irrelevant to them (i.e. where they are unable to create issues). 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 they have create permission, 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.

After uncommenting the lines, start up JIRA again, and you should see the "Reporter (show only projects with create permission)" option added to the Add New Permission page:

![Add New Permission](image)

Note that this is different than the original "Reporter" permission.

**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:

```sql
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

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

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/Administration/ManageIssueType!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?

Please note that adding Javascript to custom fields is a customisation and not maintained as a supported part of JIRA.
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 those 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:

<table>
<thead>
<tr>
<th>System Field</th>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fix Version</td>
<td>Resolve Issue</td>
</tr>
<tr>
<td>Assignee</td>
<td>Assign Issue</td>
</tr>
<tr>
<td>Due Date</td>
<td>Schedule Issue</td>
</tr>
<tr>
<td>Reporter</td>
<td>Modify Reporter</td>
</tr>
<tr>
<td>Security Level</td>
<td>Set Issue Security</td>
</tr>
</tbody>
</table>

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?.

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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 OSWorkflow 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:
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 RA-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/IssueFieldConstants.java.
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.

```velvet
#controlHeader ($action $field.id $i18n.getText($field.nameKey)
$fieldLayoutItem.required $displayParameters.get('noHeader'))
```

Here is where the changes start:

```velvet
<!-- 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

```velvet
#if ($versions & $!$versions.empty)
```

but these are the lines that change what is displayed. A "break" command would be useful in Velocity.
All the other lines in this file are unchanged except for the closing \end line
In case that was a bit too detailed, here is the diff for JIRA 3.8.1:


[jira@toolsdev plugins]$ diff -c
-/atlassian-jira-enterprise-3.8-standalone/atlassian-jira/WEB-INF/classes/templates/jira/issue/field/versions-edit.vm
***
15:49:33.000000000 -0800
---
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
+         Unknown<br>
+       #end
+       #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="$!unknownVersionId"
+          #end
+       #if (!$fieldLayoutItem.required)
+         <option value="$!unknownVersionId"
+  ***************
*** 25,30 ****
--- 53,59 ----
</optgroup>

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
#end
</select>
+   #end <!-- readonly -->
#else
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:

```html
<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.

⚠️ We strongly recommend deactivating the user instead of deleting it since deleting a user may cause database integrity problems.

**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:
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`:

```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 5 rows of options (refer screenshot below), 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.

Atlassian does not support customizations to JIRA. For more information about Atlassian support, see Atlassian Support Offerings.

The Steps

1. In Custom fields screen, click on Edit of the custom field.
2. When the page is loaded, refer to ?id= at the address field of the browser for the custom field ID, refer the screenshot below on the highlighted red area for the ID

3. Replacing last five digit of ‘customfield_10010’ on the sample script below to match the ID you have obtained on the step above:

```javascript
<script type="text/javascript">
mselectbox = document.getElementsByName('customfield_10010');
mselectbox[0].setAttribute("size", "20");
</script>
```
4. The size of the customfield will change based on the attribute configured on the parameter, example above, the size of multi select field will be expand to 20 rows when the "(size,"20") is set, feel free to replace the number to suit your own preference.
5. Put the script above to Description of custom field (shown below) and update.

```
A description of the particular custom field.
You can include HTML, make sure to close all your tags.
```

Result
The size will change based on the parameter set on the script.

```
 1
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
14
15
16
17
18
19
20
```

How to change the location of stdout and stderr logs

If JIRA is running as service in Windows, stdout_.log and stderr_.log files will be created. These files will grow rapidly and over time can become very large. You can change the location of these files to a different location, if needed. One of the following methods can be followed:

**Method 1**

To change the stdout_.log and stderr_.log location,

1. Remove the JIRA service.
2. Open the service.bat file 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, for example set PR_LOGPATH=D:\logs
4. Run service.bat install JIRA_SERVICE_NAME to install JIRA as a service again.

Method 2

1. Open the JIRA service configuration as explained at Setting Properties and Options on Startup - via CommandLine.
2. Modify the location of the stdlog and stderr paths as shown in the screenshot below:

3. Apply the changes.
4. Restart JIRA service for the changes to take effect.

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.

1. The function to use is Update Issue.
2. The field to use is Resolution.
3. The value to choose is None.

How to configure comment field to become mandatory in workflow transition

Symptoms

A user may want to know how to configure the comment field to be mandatory on every transition screen.

i Tips: There is a third party plugin that could help user to achieve it.

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Pre-requisite:

Please install JIRA Misc Workflow Extension plugin, which is available from this link.

⚠️ Note: check on JIRA compatible version.

Workaround

1. Install the plugin manually (Administrative > Plugins > Install Plugins > Upload Plugin).
2. Please edit the workflow and add the Comment Required Validator from validator transition.
3. Configure the custom error message and choose any group to be allowed or available to comment.

Related Content

Expand to see related content

How to configure comment field to become mandatory in workflow transition

Help us improve!

Error rendering macro 'kbsurvey' : null

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:


How to convert types using Jelly

Scenario

When programming in Jelly, you can get in a situation as illustrated by the following code snippet:

```
Unknown macro: {htmlcomment}
```

http://support.atlassian.com/browse/JSP-24361

How to convert types using Jelly
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 not the same type, you should get an error like this:

```
Could not run script.
Extra Information: [hide]
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 successfully.

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:
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

How to deactivate comments for closed issues

Symptoms

As a JIRA Administrator, you want to make sure that it is not possible to comment on closed issues.

Resolution
Go to your Workflow Editor and add a step property to the `Closed` status:

![Workflow Diagram]

There, add the property `jira.permission.comment.user` and make the property value `denied`:

<table>
<thead>
<tr>
<th>Property Key</th>
<th>Property Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>jira.issue.editable</td>
<td>false</td>
</tr>
<tr>
<td>jira.permission.comment.user</td>
<td>denied</td>
</tr>
</tbody>
</table>

**Environment**

This was tested on JIRA 5.0.6 and 5.2.11

**Related Content**

- Expand to see related content

**How to disable the Resolve issue screen while resolving issues**

If you don't want JIRA 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.
```

http://support.atlassian.com/browse/JSP-20302

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

Please also note that you would need to add Version to the project. The tab will only show up if the Fix version field is actually used.
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:

```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" 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 or 7.0 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 `<div>` 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 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:

```
issue.column.heading.priority
```
Note that this has been changed in 4.3 and later.

Also note that this does not change Advanced search. See [JIRA-27414](https://jira.atlassian.com/browse/JIRA-27414) - Add advanced search field names to i18 file (Open).

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](#).

For more information for JIRA 4.4+ please refer to the Renaming System Fields in JIRA JIRA Knowledge base article.

## 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](https://jira.atlassian.com/browse/JRA-5198) and vote on this feature.

- **This FAQ is to document down the workaround for JRA-5198, all credit 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 implementing the below workaround

### The Steps

1. Shutdown JIRA
2. Backup Database
3. List down the current Statuses order using query below:

   ```sql
   SELECT pname,SEQUENCE FROM issuestatus order by SEQUENCE;
   ```

4. You will get result like below:

<table>
<thead>
<tr>
<th>pname</th>
<th>SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>1</td>
</tr>
<tr>
<td>In Progress</td>
<td>2</td>
</tr>
<tr>
<td>Reopened</td>
<td>3</td>
</tr>
<tr>
<td>Resolved</td>
<td>4</td>
</tr>
<tr>
<td>Closed</td>
<td>5</td>
</tr>
</tbody>
</table>

5. Update the current status sequence numbers to 10, 20, 30 instead of 1,2,3 so it's easier to modify later.

   ```sql
   update issuestatus set SEQUENCE =10 where SEQUENCE='1';
   ```

6. Repeat step above for other statuses, you will get result like below:

<table>
<thead>
<tr>
<th>pname</th>
<th>SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>10</td>
</tr>
<tr>
<td>Status</td>
<td>Sequence</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td>In Progress</td>
<td>20</td>
</tr>
<tr>
<td>Reopened</td>
<td>30</td>
</tr>
<tr>
<td>Resolved</td>
<td>40</td>
</tr>
<tr>
<td>Closed</td>
<td>50</td>
</tr>
</tbody>
</table>

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 or Re-index 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:

   ```xml
   <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 customize the size of customfield (Free Text Field) at `%JIRA_HOME%/atlassian-jira/WEB-INF/classes/templates/plugins/fields/edit/edit-textarea.vm`

http://support.atlassian.com/browse/JSP-20022
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.

![Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.]

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 templates 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-searcher-edit.vm`

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 ..&nbsp;{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**

- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
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.

Severity: Low
Article ID: JIRAKB223904557 https://support.atlassian.com/browse/JSP-63506

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.

http://support.atlassian.com/browse/JSP-18314

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.

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".

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.
import com.atlassian.jira.ComponentManager
curr = ComponentManager.getInstance().getJiraAuthenticationContext().getUser()
assig = issue.getAssignee()
result = (curr != assig)

4. Publish the changes to your workflow.

See also How to Allow Users to Assign Issues Only in a Specific Transition.

How to update custom field values during workflow transition.

This article includes the usage of third-party plugins that are not supported by Atlassian

By default, updating a custom field during a transition is not possible in JIRA, but there is workaround involving the JIRA Suite Utilities Plugin.

Step 1: install JIRA Suite Utilities Plugin
Please install JIRA Suite Utilities plugin, which is available from this link.

Note: check on JIRA compatible version.

Step 2: update your project's workflow transition
1. Navigate to the appropriate workflow
2. Create a Draft or Copy of the existing workflow
3. Navigate to the appropriate transition's Post Functions tab
4. Add Update Issue Custom Field to the post function.
5. Specify the custom field and the desired value to be updated.
   a. For example:

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


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:

```plaintext
# 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:\nusername_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:
```xml
~/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` command 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

<table>
<thead>
<tr>
<th>Version</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Username is converted to lowercase automatically</td>
</tr>
<tr>
<td>1.0</td>
<td>First Released</td>
</tr>
</tbody>
</table>

### 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

**Wanting to use JIRA as a support system or helpdesk?** Check out JIRA's [A Service Desk](#): it provides a customizable customer help portal, powerful SLA management and realtime reporting.

### Introduction

This document shows how to set JIRA up as a support system or helpdesk.

We all know JIRA is a great project tracker. It's great at tracking tasks and managing issues, and that's what a helpdesk is all about! Lots of customers are using JIRA for a helpdesk because of its versatility and extensability. Below we'll tackle the ways JIRA can address the key features required in a helpdesk.
Security

*Project Security versus Issue Security*

Typically as an end user you can only see issues that you, or your company has raised. You can secure your issues either via project security or issue security.

*Project Security*

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. This also works well for an internal helpdesk, where you can set up a project for each department.

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.

If you pick project level security, you'll also get the benefit of project level workflows, issue types, and screen schemes.

*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.
Ticket Management

Workflow
Take some time to learn about Configuring Workflow, and set one up that caters to your customers. A few tips:

- Make transition names action names. Think of them as the titles of the buttons.
- Make status names the names of states. Think of them for your reporting, for how long a ticket waits in a status.
- Use conditions to hide certain buttons from certain groups, both for simplicity and security. For example your different escalation paths might be executed by different groups.

**Shared Filters and Dashboards**

Create your work queue with Shared Dashboards and Shared Filters. Keep in mind that some filters can be personalized, using 'currentAssignee' or other Advanced Search features.

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 Atlassian Visualizes our Support Queue.

**JIRA Agile**

JIRA Agile is great for support! It offers the following features, for triage and lifecycle management:

1. **Story Points** are a good way to consider triaging tickets. More complex tickets - such as performance or setting up a new system - can get a higher number of story points. This can help gauge capacity.
2. **Use the Rapid Board** to figure out where your backlog and churn are. For example, you may have few new tickets coming in, but a lot waiting for customers to get back to you. The Rapid Board is a great way to visualize where your work is throughout its lifecycle.

**Escalation and SLA**

For a proper helpdesk, you'll need to manage your service commitments, whether they are internal or public. Here are a few tips on how to do that:

**Jelly Scripts**

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.

**Filter Subscriptions**

To notify managers or senior staff about when an issue is in breach of or about to breach a service commitment, consider filter subscriptions. See Receiving Search Results via Email for a description. 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.

**Extending JIRA with a Service**

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.

For an example of code that uses JIRA's API to escalate issues please see: Simple Escalation.

**SLA Plugin**

Check out Vertygo's SLA plugin in the Atlassian Marketplace and SLAdiator - service level agreement real-time monitor for JIRA.
Reporting and Analytics

Reporting is a key component to a great helpdesk.

**Time Tracking**

The *Time Tracking Report* is a great way of understanding where your time is spent. Consider it in combination with *Defining a Component*, so that you can bill back to certain services or departments. For an external helpdesk, or one where you need to hide the time tracking field, you may need to implement a custom field. See ’Admin-only editable field‘ from our developer docs for a guide on restricted custom fields.

**Built-in JIRA Reports**

JIRA has a good number of built-in reporting. See Generating Reports.

**Extending the Reporting Structure**

*Business Intelligence*

If your company has a data warehouse, JIRA fits into this nicely. JIRA's database model is described here, and there are some great community-driven SQL tips here.

*EazyBI*

If you'd like JIRA to house its own business intelligence, the EazyBI plugin is a good option.

Multi-Channel Support

*Email*

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.

To customize the email templates with your branding, see Customizing Email Content.

To handle transitioning issues to a new workflow status when you receive an email response, see the auto-transition listener in the JIRA toolkit.

The Enterprise Mail Handler for JIRA (JIRA 5.0+) plugin provides comprehensive inbound mail processing (including ability to manipulate JIRA issues via Directives, live template editing, integrated event listener and more), for more information see the wiki.

*Chat*

Integrating JIRA with chat can happen in two ways. From a chat client to JIRA is available through the JIRA REST api. Alternatively, displaying chats in your working IM client is a great way to notify staff of a new or updated ticket. Check out JIRA's HipChat and Jabber integrations.

*Phone*

The create or comment handler can also handle attachments. Many phone systems will send voicemails over an email, so it's possible to create a phone support project that takes attachments over email, adding voicemail messages to new tickets. See Logging Phone Calls In JIRA.

Our ShipIt VI Winner, JIRA Caller ID, is still under development. Check in there for updates.

Knowledge Management

Integration with knowledge management is a key piece of a great support tool. Here are some of JIRA's Knowledge Management features:

*Remote Issue Linking*
See Configuring Issue Linking. You can add links to Confluence, other JIRAs, or different resources altogether, to measure and manage the support tickets that were resolved by a resource in a different system. This is a powerful tool for measuring and understanding which things resources are helping solve tickets.

**Suggestimate**

Suggestimate is a plugin that tracks other tickets that may have been already filed, to cut down on duplicates. Check it out in the Atlassian Marketplace.

**Customer Portal and Integrations**

**Rest API**

The SOAP and REST APIs for JIRA are quite powerful ways to integrate third party applications with JIRA. You can create a full portal application for creating and managing issues in a simple email portal, using the REST API. For inspiration, check out what's possible in the game Minecraft, using JIRA's REST API.

**Plugin Development**

JIRA is customizable in substantial ways, if you're ready to dive into plugin development. Get started over at our development hub.

Unknown macro: {HTMLcomment} [http://tinyurl.com/jira-sla-example](http://tinyurl.com/jira-sla-example)

**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 customizable workflow can also be very useful:

- Configuring Workflow

The workflow can be customized 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
- Adding Knowledge Base Functionality To JIRA

Related Best-Practice Discussions

- JIRA as a Support System
- JIRA as a Support System
- JIRA as a Support System
- JIRA as a Support System
- JIRA as a Support System
- JIRA as a Support System
- JIRA as a Support System
- JIRA as a Support System
- Confluence UI Guidelines
- Confluence UI Guidelines
- Confluence UI Guidelines
- Using JIRA for Agile Development
- JIRA as a Support System
- Using JIRA for Project Management

This document is a work in progress. Feel free to add any comments below.

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>
<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]">
        <!-- 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 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:

![Screenshot of JIRA permission settings]

Users will see a permission error after creating an issue - not very customer-friendly!

![Permission violation error message]

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:

```xml
<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`:

```xml
  <view name="permissionviolation">/redirectusers.jsp</view>
```

**Create a redirect JSP**

---

*Created in 2013 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](https://creativecommons.org/licenses/by/2.5/au/).*
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:

```jsp
<%@ 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.sendRedirect("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 link 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](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](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, 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 JIRA-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

  ```html
  <tr>
    <td align="middle" colspan="2">
      <input id="login" type="button" onclick="CheckForm();" value="$i18n.getText('common.concepts.login')" tabindex="4" />
    </td>
  </tr>
  
  function CheckForm() {
    var Username = document.loginform.os_username.value;
    document.loginform.os_username.value = Username.toLowerCase();
    document.loginform.submit();
  }

  #if ($allowPasswordReset == true)
  <tr>
    <td align="middle" colspan="2">
      <a href="/secure/ForgotPassword!default.jspa">$i18n.getText('common.concepts.forgotpassword')</a></td>
  </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
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:
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 Using Filters.

**3. Subscribe to your Filter**

Once you have 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.

**Receiving 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.

⚠️ Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see Atlassian Support Offerings.

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 templates 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 want 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

- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size and Content of the Components Select List
- 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, the control characters will need to be removed from the JIRA backup file. This can be done with the following:

1. Download `atlassian-xml-cleaner-0.1.jar`
2. Open a command prompt and locate the XML or ZIP backup file on your computer, ensuring that it is extracted if it's within a ZIP file. In this example, we will use `entities.xml`.
3. Run the application with the below:

```
$ java -jar atlassian-xml-cleaner-0.1.jar entities.xml > entities-clean.xml
```

This will create a copy of `entities.xml` as `entities-clean.xml` with the invalid characters removed. The `entities-clean.xml` will be able to be imported without the above errors now.

### 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 [JIRA-5332](https://jira.atlassian.com/browse/JIRA-5332) for a discussion on this feature request.

Atlassian does not support customisations to Velocity templates or other JIRA files. For more information about Atlassian support, see [Atlassian Support Offerings](https://www.atlassian.com/software/help).
To remove the field, edit `<atlassian-jira/WEB-INF/classes/templates/jira/issue/field/securitylevel-edit.vm`. Delete the lines:

```html
<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**

- Changing the Size of the Text Area Custom Field
- Removing NONE from the Issue Security Drop-Down List
- Removing Commas for Values Held in Number Field Custom Field Type
- Changing the Size of the Fix Versions and Affects Versions Select List
- Changing the Size and Content of the Components Select List

**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](http://support.atlassian.com/browse/JSP-19459) 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).

Unknown macro: {htmlcomment}

http://support.atlassian.com/browse/JSP-18371

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

By default, this information is restricted based only on Project Roles. You can change this to restriction based on Groups in the General Configuration by editing the Comment Visibility option.

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 Internal JIRA Administrators
  - If no users or groups exist in JIRA
- **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

Using Single Sign-On (SSO):

- If JIRA is configured for SSO through Crowd or another third-party service, only users from Crowd or the other service will be able to log in to JIRA.
- In order to log in as a user from the JIRA Internal Directory, roll back the changes made within Integrating Crowd with Atlassian JIRA or as made by the third-party authenticator.

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.

1. To find out which group(s) have been granted the **JIRA Administrators** global permission, run the following database query:
   
   ```sql
   select perm_parameter from schemepermissions where PERMISSION=0;
   ```

2. To find out which group(s) have been granted the **JIRA System Administrators** global permission, run the following database query:
   
   ```sql
   select perm_parameter from schemepermissions where PERMISSION=44;
   ```

3. 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):
   
   ```sql
   select child_name, directory_id from cwd_membership where parent_name='jira-administrators';
   ```

   If you are having issues with remote directory connectivity, you will need to use an account with a directory_id of 1.

   If you don't know the password for the user(s) returned by this query, move on to Scenario B.

   **If there are no Internal 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:
      
      ```sql
      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;
      ```

      (Oracle only): Execute a **COMMIT** so that the transactions are completed.

      4. Restart JIRA.

   **If no users or groups exist in JIRA**

   There may be no users or groups in your Internal Directory. If this is the case, you need to add one:

   1. Add a new 'localadmin' user with the password **sphere**:  

insert into cwd_user values (999999,1,'localadmin','localadmin',1,'2012-01-04 19:49:05-08','2012-01-04 19:49:05-08', 'local', 'local', 'admin', 'admin', 'local admin', 'localadmin@localadmin.com', 'localadmin@localadmin.com', 'uQieO/1CGMUIXX ftw3ynrsaYLShI+GTCpPS4IdUGWb1usFvHFuZ7CZvms6yMMvA817FvIHEqer6Mj4pCLKAFQ==');
insert into app_user values(999999,'localadmin','localadmin');

If you are using Oracle database use the following:

insert into cwd_user values (999999,1,'localadmin','localadmin',1,TO_DATE('2012-01-04 19:49:05','yyyy-mm-dd hh24:mi:ss'),TO_DATE('2012-01-04 19:49:05','yyyy-mm-dd hh24:mi:ss'), 'local','local','admin','admin','local admin','local admin','localadmin@localadmin.com','localadmin@localadmin.com','uQieO/1CGMUIXX ftw3ynrsaYLShI+GTCpPS4IdUGWb1usFvHFuZ7CZvms6yMMvA817FvIHEqer6Mj4pCLKAFQ==');

Add new groups:

insert into cwd_group(id, group_name, lower_group_name, active, local, created_date, updated_date, description, lower_description, group_type, directory_id) values ( '888888','jira-administrators','jira-administrators',1,0,'2011-03-21 12:20:29','2011-03-21 12:20:29',NULL,NULL,'GROUP',1);
insert into cwd_group(id, group_name, lower_group_name, active, local, created_date, updated_date, description, lower_description, group_type, directory_id) values ( '777777','jira-users','jira-users',1,0,'2011-03-21 12:20:29','2011-03-21 12:20:29',NULL,NULL,'GROUP',1);

If you are using Oracle database use the following:

insert into cwd_group(id, group_name, lower_group_name, active, local, created_date, updated_date, description, lower_description, group_type, directory_id) values ( '777777','jira-users','jira-users',1,0,'2011-03-21 12:20:29','2011-03-21 12:20:29',NULL,NULL,'GROUP',1);

3. Add groups to the appropriate Global Permissions:

insert into schemepermissions (id, scheme, permission, perm_type, perm_parameter) values (444444,NULL,44,'group','jira-administrators');
insert into schemepermissions (id, scheme, permission, perm_type, perm_parameter) values (333333,NULL,1,'group','jira-users');

4. Add the group memberships for the 'localadmin' user:

insert into cwd_user values (999999,1,'localadmin','localadmin',1,'2012-01-04 19:49:05-08','2012-01-04 19:49:05-08', 'local', 'local', 'admin', 'admin', 'local admin', 'localadmin@localadmin.com', 'localadmin@localadmin.com', 'uQieO/1CGMUIXX ftw3ynrsaYLShI+GTCpPS4IdUGWb1usFvHFuZ7CZvms6yMMvA817FvIHEqer6Mj4pCLKAFQ==');
insert into app_user values(999999,'localadmin','localadmin');

If you are using Oracle database use the following:

insert into cwd_user values (999999,1,'localadmin','localadmin',1,TO_DATE('2012-01-04 19:49:05','yyyy-mm-dd hh24:mi:ss'),TO_DATE('2012-01-04 19:49:05','yyyy-mm-dd hh24:mi:ss'), 'local','local','admin','admin','local admin','local admin','localadmin@localadmin.com','localadmin@localadmin.com','uQieO/1CGMUIXX ftw3ynrsaYLShI+GTCpPS4IdUGWb1usFvHFuZ7CZvms6yMMvA817FvIHEqer6Mj4pCLKAFQ==');

Add new groups:

insert into cwd_group(id, group_name, lower_group_name, active, local, created_date, updated_date, description, lower_description, group_type, directory_id) values ( '888888','jira-administrators','jira-administrators',1,0,'2011-03-21 12:20:29','2011-03-21 12:20:29',NULL,NULL,'GROUP',1);
insert into cwd_group(id, group_name, lower_group_name, active, local, created_date, updated_date, description, lower_description, group_type, directory_id) values ( '777777','jira-users','jira-users',1,0,'2011-03-21 12:20:29','2011-03-21 12:20:29',NULL,NULL,'GROUP',1);

If you are using Oracle database use the following:

insert into cwd_group(id, group_name, lower_group_name, active, local, created_date, updated_date, description, lower_description, group_type, directory_id) values ( '777777','jira-users','jira-users',1,0,'2011-03-21 12:20:29','2011-03-21 12:20:29',NULL,NULL,'GROUP',1);
5. Enable the JIRA Internal Directory:

```sql
update cwd_directory set active = 1 where id = 1;
```

6. *(Oracle only):* Execute a COMMIT so that the transactions are completed.

7. Restart JIRA.

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 with the following SQL:

```sql
select u.user_name, d.directory_name from cwd_user u inner join cwd_directory d on u.directory_id = d.id order by directory_name, user_name
```

The password can be reset with either of the following:

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 - try the above recommended approach first.**

   1. 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*.

   ```sql
   update cwd_user set 
   credential='uQieO/1CGMUIXXftw3ynrsaYLSH+iGTcPS4LdUGWbIusFvHPfUzD7CZvms6yMMvA8I7FVHEqr6Mj4pCLKAFQ==' where user_name='XXXX';
   ```

   2. *(Oracle only):* Execute a COMMIT so that the transactions are completed.

   3. Restart JIRA.

**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.

As of JIRA 4.1.1, it is no longer possible to send data via the **Administration -> Support Request** page. Please see below for instructions on providing a manual XML Backup.

- Manual XML Backup
- Anonymising JIRA Data:
- Information about the Anonymiser:
- Problems:
  - Invalid XML Characters

Manual XML Backup

To perform a once-off backup, follow the steps below.

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'
As shown in the screenshot above, the backup will be stored within the export subdirectory of the JIRA Home Directory.

3. When the backup is complete, a message will be displayed, confirming that JIRA has written its data to the file you specified.

4. Attach the generated file on disk to a support request on support.atlassian.com.

Support requests are often resolved significantly faster if a data export is provided as it will allow our legendary supporters direct access to a copy of your instance. We understand that sometimes this may be a difficult option due to the sensitivity of your data and have written an anonymising tool to handle this particular scenario.

Anonymising JIRA Data:

The JIRA inbuilt backup functionality will produce a ZIP file containing either 1 or 2 XML files, depending on the version that is being used. These files are a copy of the entire contents of JIRA’s database, encoded in XML, that can be used to restore an instance - we have further detail on this in our Automating JIRA Backups documentation.

As of JIRA 4.4, the backup functionality will produce a ZIP file that contains 2 XML files. These files will be activeobjects.xml and entities.xml. Only entities.xml will need to be anonymised - please do not attempt to anonymise the activeobjects.xml. For versions prior to 4.4, only one XML file will be produced with the same naming convention as the ZIP it is compressed as (for example 1970-Jan-01-0001.zip will expand to 1970-Jan-01--0001.xml).

1. Ensure that the JAVA_HOME variable has been configured, as in our Setting JAVA_HOME documentation.
2. Download the JIRA Anonymiser.
3. Create a temporary directory.
4. Unzip the anonymizer in the temporary directory.
5. Unzip the JIRA backup ZIP file (for example 1970-Jan-01--0001.zip) in the temporary directory.
6. Anonymise the backup file with the below commands:

   $ java -Xmx512m -jar joost.jar <JIRA BACKUP>.xml anon.stx > <NAME OF ANONYMISED BACKUP>.xml

For example, this would be anonymising a JIRA backup with the naming convention from JIRA 4.4+:

   $ java -Xmx512m -jar joost.jar entities.xml anon.stx > anon-entities.xml

Depending on the size of the backup, additional memory may need to be allocated to the JVM. In order to do this, increase the value of the Xmx in increments of 128m.

7. Compress the generated anonymised XML backup file (e.g.: anon-entities.xml) and the activeobj
ects.xml (JIRA 4.4.x + only) into a ZIP or tarball.
8. Attach that ZIP or tarball onto the support issues as raised on support.atlassian.com.
9. The temporary directory can now be removed.
The screenshot below is a simple example of how it is run in the command prompt of Windows XP:

Information about the Anonymiser:
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.

Please check the anonymised backup, anon-backup.xml, to ensure it’s clean enough for the needs of your organisation before sending it to Atlassian.

Problems:

**Invalid XML Characters**
If, when the anonymiser runs, an error indicates that there are invalid XML characters in the XML backup of the database, run our utility to remove invalid XML characters first before anonymising.

**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.

Several 3rd party mail handlers also exist for JIRA that provide features for setting additional fields:

<table>
<thead>
<tr>
<th>Name</th>
<th>Compatibility</th>
<th>Fields Supported</th>
<th>License</th>
<th>JRA’s</th>
<th>Support Link</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>JIRA Enterprise Mail Handler</td>
<td>5.0 - current</td>
<td>ALL, see supported fields</td>
<td>Commercial</td>
<td>see list</td>
<td>link</td>
<td>see website</td>
</tr>
<tr>
<td>JIRA Advanced Mail Handler</td>
<td>3.7 - 3.12.3</td>
<td>see below</td>
<td>Free</td>
<td>JRA-7316</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Enterprise Mail Handler

JEMH is a fully featured mail handler, that supports all common JIRA fields, as well as complex types like cascade selects. Directives to set fields can be supplied in many ways through Field Processors, e.g. subject, body, XML, CSV, regular expression matching, SMTP mail X-header values. JEMH configuration is now fully UI based, and includes TestCase features to validate configuration.

Examples of common formats come bundled in the plugin (generate examples option in TestCases section). See also the wiki.

Advanced Mail Handler

The 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:

```vm
#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:

```vm
#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 ($description = "Step 1\n   Step 2\n   Step 3")
#set ($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:

```text
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
    &nbsp;
  #end

Edited - full time-stamp

#if ($created)
  $outlookDateManager.getOutlookDate($authcontext.getLocale()).formatDMYHMS($created)
#else
    &nbsp;
#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`:

```text
#if ($value) <span title="${title}"><time datetime="${iso8601}">${title}</time></span> #else &nbsp; #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

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 life-cycles. 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 organisation, eg. through shareable sub-projects (JIRA-1072). Please vote/add your thoughts to the issue to increase its popularity. Also, please refer to this document which explains the way Atlassian schedules new features.

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\stderr_*.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**

Wondering why your field isn't showing up on the View Issue screen, Edit Issue screen or Create Issue screen?

The JIRA Field Helper shows JIRA Admins exactly why a field isn't appearing on the screen and gives you explicit instructions on how to make it appear. It looks at the field's status for combination of the current project, issue type and screen that you are viewing.

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 internal database which is used for evaluation purposes. 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

• JIRA and PostgreSQL
  • How to Set Up SMTP Relay in Exchange 2007
  • How to Use System JRE Instead of Embedded JRE
  • Installation LDAP server on OpenDS and ApacheDS in JIRA (Windows Environment)
  • 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.

---

Now in the properties for the 'jakra' 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 to:

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
JIRA needs to know what kind of database it will be using, in order to generate database tables of the correct type.

### 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 internal database which is used for evaluation purposes. 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

**Background**

JIRA needs to know what kind of database it will be using, in order to generate database tables of the correct type.
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
<?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:
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.
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:

<table>
<thead>
<tr>
<th>Database</th>
<th>Limitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>1000. (See JIRA-19317 - Ofbiz generates queries with large IN clauses that result in ORA-01795 errors (Soaking) for more details.)</td>
</tr>
<tr>
<td>SQL Server</td>
<td>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.</td>
</tr>
<tr>
<td>MySQL</td>
<td>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.</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>No hard limits; SQL query size is limited to the OS maximum file size.</td>
</tr>
<tr>
<td>HSQLDB</td>
<td>No limits specified, but in practice limited by heap memory.</td>
</tr>
</tbody>
</table>

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 internal database which is used for evaluation purposes. This page describes how to obtain a SQL console for HSQLDB databases, which are built into JIRA and Confluence for evaluation purposes.

**Locate HSQLDB directory**

HSQL stores its database as text files in the filesystem. Typically these files 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 HSQLDB jar**

The hsqldb jar file is located in the lib sub-directory of the JIRA Installation Directory:
Shutdown JIRA/Confluence

If you haven't already, shut down any apps using the database.

Run The Console

Use the following command to bring up the HSQLDB console (replacing the JIRA_HOME and JIRA_INSTALL with the absolute paths)

```
java -cp JIRA_INSTALL/lib/hsqldb-1.8.0.5.jar
org.hsqldb.util.DatabaseManager -user sa -url
jdbc:hsqldb:JIRA_HOME/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:

Shutdown The Console

Once you have finished running SQL queries, shut down the console before starting JIRA/Confluence.

Troubleshooting

If you can successfully bring up the console, but no tables are shown, the jdbc:hsqldb: the usual reasons are

- JIRA/Confluence is still running. Ensure that the applications are shutdown before running the console.
- Path of the database file may be incorrect. You can update this by navigating to File > Connect and entering in the absolute path to JIRA_HOME/database/jiradb in the URL box and clicking Ok.
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:

```
org.ofbiz.core.entity.GenericEntityException: while inserting: [GenericEntity:ListenerConfig][clazz,com.atlassian.jira.event.listeners.cache.IssueCacheListener][name,Issue 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.j
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.)

```
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.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.core.startup.HostConfig.deployApps(HostConfig.java:493)
at org.apache.catalina.core.startup.HostConfig.check(HostConfig.java:1195)
at sun.reflect.GeneratedMethodAccessor341.invoke(GeneratedMethodAccessor.java:24)
```
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


at

org.ofbiz.core.entity.jdbc.SQLProcessor.executeUpdate(SQLProcessor.java:373)

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.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: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)
add your notes

MS SQL Server 2000 Startup errors

A user reports getting errors like these every time JIRA starts up:

```java
[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, if 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 described 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.

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](image1)

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](image2)

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 CHARACTER SET utf8 COLLATE utf8_bin;
   ```
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

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:

- Select typical install
- Configure MySQL with the Configuration Wizard
  - Detailed Configuration
  - Server Machine
  - Multifunctional Database
  - Chose C:\Installation path> for the InnoDB tablespace
  - Decision Support DSS/OLAP
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
  - username and password for the jirauser account set up above
  - driverClassName="com.mysql.jdbc.Driver"
  - url="jdbc:mysql://localhost/jiradb?autoReconnect=true&useUnicode=true&characterEncoding=UTF8"
  - 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
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup

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"
  - Install as Windows Service
  - Add \bin to path
  - Choose root password ************
- Execute configuration.
  - Success!
- Run MySQL Config Wizard. Choose options as above.
- Execute configuration
  - Success!
- Set up MySQL database and jira user as above.
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup
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:
  - DROP USER 'jirauser@'localhost';
  - CREATE USER 'jirauser@localhost' IDENTIFIED BY '********'; (making sure password has no & in it)
  - GRANT ALL PRIVILEGES ON jiradb.* TO 'jirauser@localhost';
  - quit;
- Edit conf/server.xml to the new password
- run C:\jira\atlassian-jira-professional-3.6.5-standalone\bin\startup
  - 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
  - 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/
  - 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'
```

OR

```
org.ofbiz.core.entity.GenericEntityException: while updating:
[GenericEntity:IssueType][id,1][iconurl,/images/icons/issuetypes/bug.png]
(SQL Exception while executing the following:UPDATE issuetype SET ICONURL=? WHERE ID=? (Cannot execute statement: impossible to write to binary log since BINLOG_FORMAT = STATEMENT and at least one table uses a storage engine limited to row-based logging. InnoDB is limited to row-logging when transaction isolation level is READ COMMITTED or READ UNCOMMITTED.))
```

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)
```

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 Administrator 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 - Errocode - 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:81) 
at 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(DatabaseCompatibilityEnforcer.java:32) 
at 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:
<table>
<thead>
<tr>
<th>JIRA 6.1 Documentation</th>
</tr>
</thead>
</table>

```
jturner@teacup:$ netstat -na | grep 3306
<table>
<thead>
<tr>
<th>Protocol</th>
<th>Local Address</th>
<th>Foreign Address</th>
<th>Status</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp</td>
<td>0 127.0.0.1:3306 0.0.0.* LISTEN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tcp</td>
<td>0 127.0.0.1:48211 127.0.0.1:3306 TIME_WAIT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tcp6</td>
<td>1 ::fff:127.0.0.1:34785 ::fff:127.0.0.1:3306 CLOSE_WAIT</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Or if `netstat` isn't available:

```
jturner@teacup:$ telnet localhost 3306
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
5.0.13-rc-Debian_1-lo!X($)V#H!ju (press ctrl-] here)
telnet> quit
```

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 COLLATE utf8_bin;
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)

Tip: make it so that only one production server can connect to the database, so that when you later bring up a clone of the production server as a staging server, you will be protected from having two JIRA instances using the same database.

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:

```bash
mysql -p -u [dbuser] -h 127.0.0.1 [dbname]
```

For example,

```bash
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:

```bash
mysql -p -u root
```

And enter following command:

```sql
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:

```bash
netstat -a |grep mysql
```

or,

```bash
netstat -a |grep 3306
```

If MySQL is listening, you should see the following message:

```bash
tcp 0 0 *:mysql *:* LISTEN
```

Alternatively, you also could check if your MySQL is listening on the default port by running this command:
4. If you can connect successfully from the command line but JIRA cannot connect it may because a non-default port is in use. Note that the `mysql` tool can connect to non-standard ports without you having to specify a port, so it isn't always a guarantee of connectivity.

### 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 `dbcofig.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:
JIRA 6.1 Documentation

```java
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.)
```

```java
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.OraSQLException: [OraDriver] #7 Unexpected end of inputstream in header.
```

```java
at com.inet.ora.Ora3Factory.createSQLException(Unknown Source)
at com.inet.ora.Ora3Factory.createSQLException(Unknown Source)
at com.inet.ora.OraConnection.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:

```xml
<connection-properties>SetBigStringTryClob=true</connection-properties>
```

for this driver it needs to be replaced with:

```xml
<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

![Warning](https://via.placeholder.com/150)

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.
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:

   ```xml
   <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
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.

**How to Use System JRE Instead of Embedded JRE**

JIRA installer comes with its own JRE, hence JIRA won't use system JRE when its run. If you want to configure JIRA to use system JRE please follow the steps below:

1. Shutdown JIRA
2. Edit `<JIRA_Installation_Directory>/bin/setenv.bat` or `setenv.sh` on your favorite text editor
3. Input the following lines:
   For setenv.bat:
   
   ```bash
   SET JRE_HOME="<Full Path of Your JRE Directory>"
   ```
   
   For setenv.sh:
   
   ```bash
   JRE_HOME="<Full Path of Your JRE Directory>"
   ```
   
4. Change the JRE_HOME value with the absolute path of your system JRE.
5. Save
6. Restart JIRA

How
The bundled JRE version is 1.6 update 26. Please check your System Information page (JIRA Admin > System Information) and see if JIRA run in this JAVA version instead of the installed JRE on your system.

Installation LDAP server on OpenDS and ApacheDS in JIRA (Windows Environment)

This page contains an example of how to install/configure LDAP server on OpenDS and ApacheDS based in Windows environment in JIRA.

Comparison between OpenDS and ApacheDS in terms of Specifications and Offered Features

<table>
<thead>
<tr>
<th></th>
<th>OpenDS</th>
<th>ApacheDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Licensing</td>
<td>Open Source and Free</td>
<td>Open Source and Free</td>
</tr>
<tr>
<td>Language Written</td>
<td>Java</td>
<td>Java</td>
</tr>
<tr>
<td>Supported Environment</td>
<td>Linux and UNIX distributions, Microsoft Windows, Mac OS X,</td>
<td>Linux and UNIX distributions, Microsoft Windows, Mac OS X,</td>
</tr>
<tr>
<td>Installation Method</td>
<td>• Installer</td>
<td>• Installer</td>
</tr>
<tr>
<td></td>
<td>• Archive zip</td>
<td>• Archive zip</td>
</tr>
<tr>
<td>Java Platform</td>
<td>Java Runtime Environment 1.5 and above</td>
<td>Java Runtime Environment 1.6 and above</td>
</tr>
</tbody>
</table>

How
You will need to download and install Apache Directory Studio which is a Eclipse tool act as the LDAP Browser/Editor in order to use ApacheDS.
## Offered Features

- Management of recurrent and scheduled tasks
- Monitoring of data in the Control Panel
- Collation matching rules for I18N
- Assured replication
- Configuration of replication groups

The Apache Directory Server is an embeddable LDAP server implemented in pure Java. It has several features that make it unique among LDAP servers. Some of these features are as follows:

- Designed as an LDAP and X.500 platform; plugable components and subsystems make ApacheDS extremely modular and ideal for experiments with various aspects of the LDAP protocol.
- The server's frontend is completely separable from its backend and vice-versa making it very flexible for implementing virtual directories, proxy servers and gateways to X.500 directories.
- Several backends can be implemented and plugged into the server's partition nexus. The server supports a BTree based partition out of the box but any backing store can be used to implement a partition as long as it conforms to interfaces.
- The server exposes aspects of administration via a special system backend. LDAP can be used to manage these concerns through the system naming context at ou=system.
- The server contains a server side JNDI LDAP provider as the facade for the entire backend subsystem. JNDI operations are directly translated by this provider into operations against the nexus and the target partitions storing server entries.
- The server's networking code, MINA (Multipurpose Infrastructure for Network Applications) was designed for pluggable protocol providers, of all sorts and not just LDAP. MINA gives ApacheDS the ability to handle large amounts of concurrency.
- LDAP Stored Procedures and Triggers are scheduled for the next major version of ApacheDS.
- LDAPv3 compatible certified by the OpenGroup
Unique Usage Point

- Setup process is a breeze and straightforward since mostly is already pre-configured
- Love the Control Panel GUI interface which is easy to understand and very user friendly
- Suitable for beginner that are new to LDAP environment
- Offered Apache Directory Studio act as a powerful LDAP IDE (similar like Eclipse, Netbeans) which is useful when it comes to advanced configuration and debugging
- High scalability and flexibility in terms of implementation

Basically both of these LDAP directory server environment it is quite similar in terms of specification and supported environment too as is widely available for almost all operating system since it is both running in Java based application. Best of all it is open source and free.

Installation

1. Click on Video tutorial
2. Click on Document tutorial

This is based in Windows Environment only

OpenDS

1. Download OpenDS 2.2 Update 1 archive zip and extract it out to a particular location
2. Run setup.bat file and then basically follow either one of the tutorial below to setup your LDAP server

<table>
<thead>
<tr>
<th>Video tutorial</th>
<th>Document tutorial</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://java.net/projects/opends/pages/2_4_OverviewOfTheQuickSetupTool">https://java.net/projects/opends/pages/2_4_OverviewOfTheQuickSetupTool</a></td>
<td></td>
</tr>
</tbody>
</table>

ApacheDS

1. Download ApacheDS 2.0.0-M13 and Apache Directory Studio 2.0.0-M7 either 32-bit or 64-Bit
2. Install ApacheDS first with the installation wizard then do the same for Apache Directory Studio (pretty straightforward)
3. After that launch Apache Directory Studio, you can basically follow the instruction to create a LDAP server in here: http://directory.apache.org/studio/users-guide/ldap_browser/gettingstarted_create_connection.html

Setting it up with JIRA

Setup Environment:

<table>
<thead>
<tr>
<th>OpenDS</th>
<th>ApacheDS</th>
</tr>
</thead>
</table>
| • Windows 7 Professional 64-bit  
  • JIRA 5.2.11 Standalone connected to PostgreSQL 8.4 database  
  • JRE 1.6  
  • OpenDS 2.2 Update 1 | • Windows 7 Professional 64-bit  
  • JIRA 5.2.11 Standalone connected to PostgreSQL 8.4 database  
  • JRE 1.6  
  • ApacheDS 2.0.0-M13  
  • Apache Directory Studio 2.0.0-M |

General Setup Guidelines

1. Setup LDAP server in both OpenDS and ApacheDS (refer the installation instruction above)
2. Configure the users and groups in both server
3. Fresh install JIRA 5.2.11 connected with a fresh empty Postgres database
4. Go to Administrator > Users > User Directories > Add Directory
5. Choose Directory Type: LDAP and select Apache Directory Server 1.5.x for ApacheDS and OpenDS for OpenDS
6. Fill up all other required field: Name, Directory Hostname, Port and Base DN
7. Select Read Only as the LDAP Permissions

Created in 2013 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Make sure the **attributes** and **object class** you set in your LDAP server is correctly match the attributes that you bind in JIRA (under User and Group Schema Settings).

**Example:**
- The default bind user name attribute for OpenDS in JIRA (User Schema Settings) is **uid** so in your OpenDS LDAP server you have to set your username under **user id** attribute.
- The default bind user name attribute for ApacheDS in JIRA (User Schema Settings) is **cn** so in your ApacheDS LDAP server you have to set your username under **common name** attribute.

### Differences in terms of User Interface and Configuration

<table>
<thead>
<tr>
<th>UI</th>
<th>OpenDS</th>
<th>ApacheDS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
<td><img src="image" alt="OpenDS Directory Server Control Panel" /></td>
<td><img src="image" alt="ApacheDS Directory Server Control Panel" /></td>
</tr>
</tbody>
</table>

#### Connection Handlers

<table>
<thead>
<tr>
<th>Address:Port</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>LDIF</td>
</tr>
<tr>
<td>0.0.0.0:151</td>
<td>SNMP</td>
</tr>
<tr>
<td>0.0.0.0:389</td>
<td>LDAP</td>
</tr>
<tr>
<td>0.0.0.0:636</td>
<td>LDAPS</td>
</tr>
<tr>
<td>0.0.0.0:1689</td>
<td>JMX</td>
</tr>
</tbody>
</table>

#### Data Sources

<table>
<thead>
<tr>
<th>Base DN</th>
<th>Backend ID</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>dc=example,dc=com</td>
<td>userRoot</td>
<td>3</td>
</tr>
</tbody>
</table>

**Diagram:**
- OpenDS Directory Server Control Panel
- ApacheDS Directory Server Control Panel
- Administration Connector: Port 4444 (LDAP)
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.

**Error on Dashboard about Gadget plugin**

Some users have reported that installing JIRA 5.x on the Amazon Free servers succeeds except for an error on the Dashboard screens about gadget plugins not being found. Adding more memory to the VM made this error go away. More information at JIRA System Plugin Timeout While Waiting for Plugins to Enable

**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:

```bash
apt-get install slapd ldap-utils
```

Entering the following details when prompted (customize for your organization):

<table>
<thead>
<tr>
<th>Domain name</th>
<th>atlassian.com</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization name</td>
<td>Atlassian</td>
</tr>
<tr>
<td>Admin password</td>
<td>secret</td>
</tr>
<tr>
<td>LDAP v2 protocol</td>
<td>no</td>
</tr>
</tbody>
</table>

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`:

```bash
apt-get install migrationtools
```

1. Edit `/etc/migrationtools/migrate_common.ph`, and make the following changes:
1. Default DNS domain
   -$DEFAULT_MAIL_DOMAIN = "padl.com";
   +$DEFAULT_MAIL_DOMAIN = "atlassian.com";

1. Default base
   -$DEFAULT_BASE = "dc=pdl,dc=com";
   +$DEFAULT_BASE = "dc=atlassian,dc=com";

1. Turn this on for inetLocalMailRecipient
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 DUACfgProfile [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:

```
./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:

```
ldappasswd -x -v -S -W -D "cn=admin,dc=atlassian,dc=com" "uid=jturner,ou=People,dc=atlassian,dc=com"
```

New password:

You should now be able to connect anonymously, or as an authenticated user:
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:
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
*          /usr/bin/gij-wrapper-4.1
+          /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 Installation Diagram](image)

**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 until recently no supported Oracle JDK and JRE (formerly Sun JDK and JRE) was available for this operating system. It is our view that the cost of supporting JIRA on OS X would be better spent fixing bugs and developing other requested features.

**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](link).
   - 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.

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 customizing 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 Optimizing 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 affect 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 user daemon. 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
<?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>
  <string>/usr/local/apache-tomcat-5.5.23/bin/catalina.sh</string>
  <string>run</string>
</array>
  <array>
    <key>RunAtLoad</key>
    <true/>
    <key>ServiceDescription</key>
    <string>JIRA autoloaded as a service</string>
    <key>UserName</key>
    <string>pw</string>
  </array>
</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

Request for interest

Hi there,

Great to see your interest in Clustering/Load Balancing for JIRA!

Atlassian is looking to build a solution for this to help large Enterprise customers that run JIRA in mission...
Does JIRA support clustering for scaling?

Not at this time.

However we do understand that customers are interested in how JIRA can scale. To address this we have written a document that explains how JIRA performs in different configurations and it also provides a guide of how to scale JIRA in a large enterprise. So whether you are a new JIRA evaluator that wants to understand how JIRA can scale to your needs, or you’re a long time existing JIRA administrator that is interested taking JIRA to the next level we have a document that will point you in the right direction.

To scale JIRA in your organization, there are two key approaches, which can be used in combination to scale JIRA across your entire organization:

1. Scaling a single JIRA instance.
2. Federation: Using multiple, connected JIRA instances

We detail both techniques to get the most out of JIRA for your organization at our Scaling JIRA guide.

What if I really want JIRA clustering?

Although not supported if this is something you must have then in your environment you may wish to work with one of our Atlassian Experts to come up with a tailor made solution. Please remember that this will not be supported by Atlassian. This means that if you experience a problem that can not be replicated in a non-clustered environment then that is where our support will most likely stop.

Note: If you wish to run JIRA in a clustered environment you will need a license for each node. You can place your order from our secure online order form.

Does JIRA support high-availability?

While not natively supported it is possible to build an environment with a JIRA failover strategy. This best practices guide: Failover for JIRA, assembles some of the best advice from our customers, our partners, and internal staff on setting up a failover solution for JIRA.

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:

```
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
INFO:             header=accept=text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
INFO:             header=accept-language=en-us,en;q=0.5
INFO:             header=accept-encoding=gzip,deflate
INFO:             header=accept-charset=ISO-8859-1,utf-8;q=0.7,*;q=0.7
INFO:             header=keep-alive=300
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:
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 15:43:27,582 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.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)
JIRA 6.1 Documentation

at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
  at
  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(NativeMethod)
  at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
  at
  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(NativeMethod)
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)
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.

How to Hide "Can't access your account?" Link

Administrator might want to hide the forgot password link from the log-in page to ensure that only JIRA administrator that could reset users password without enabling "External User Management" option.

In order to do this, please try the following steps

1. If using JIRA 4.0.2 or earlier:

2. If using JIRA 4.1 or later:


4. Unzip jira-gadgets-plugin-x.jar.

5. Inside of the jar file is /gadgets/login.xml, the definition for the login gadget.

6. Find the following lines

   ```javascript
   var rememberMe = AJS.$("#rememberme");
   forgotPassword = AJS.$("#forgotpassword");
   if (!login.allowCookies) {
   ```

7. Modify it to:

   ```javascript
   var rememberMe = AJS.$("#rememberme");
   forgotPassword = AJS.$("#forgotpassword");
   forgotPassword.hide();
   if (!login.allowCookies) {
   ```

8. Zipped it back

9. Restart JIRA

How to display custom field of the sub-task in the parent issue screen?

Resolution

**JIRA 5.x**

By adding the custom field id in the Administration > Advanced Settings page.

Steps to follow:

1. Create a custom field through Administration --> Custom Fields
2. In the Custom Fields page, click on the Gear button at the right of the custom field's row:

3. Hover over Edit and at the bottom of the page you will see the custom field's id (e.g. id=10208)
4. Go to Administration --> General Configuration --> Advanced (button at the bottom of the page) --> jira.table.cols.subtasks.
5. Click on the right column where it says (by default) issuetype, status, assignee, progress and add customf ield_10208
6. Click on Update.

If you get no errors then you have probably done the trick. Navigate to an issue containing sub-tasks and you should see your custom field in the sub-task list (provided that the sub-task holds values in that field, of course).

Tested version
The above has been tested in JIRA 5.0.7, 5.2.4

JIRA 4.4.x

By editing the value of jira.table.cols.subtasks property in the jpm.xml file edit within the <default-value> tags. For example, in this situation if you wanted due date to come up you would change it like this and restart JIRA:

<jira.table.cols.subtasks>
  <issuetype, status, assignee, progress, duedate</default-value>
</jira.table.cols.subtasks>

Prior to JIRA 4.4

By editing the value of jira.table.cols.subtasks property in the jira-application.properties could solve the problem. For example:

jira.table.cols.subtasks = priority, issuetype, status, resolution, versions, assignee, customfield_10000

A restart of JIRA needs to be performed after making the modification.
How to change the font size for printing an issue

To use specific font size for printing purposes, custom css specifying the desired font size can be add into the Announcement Banner. The steps are as follow:

1. Go to 'Administration' > 'System' > 'User Interface' > 'Announcement Banner'
2. Enter the following code into the Announcement Banner section and change your desire font size:

```html
<html>
<head>
<style>
@media print
{body{font-size:9pt !important;}}
</style>
</head>
```

3. Print preview an issue view to check if the JIRA font has changed into desired size.
How to recover the comments viewable by a Project Role which has been deleted

1. You have a project role named "OLD" which was created some time ago.
2. Under circumstances, you decide to delete it and replace by a new project role named "NEW", or you have accidentally delete it.
3. Some users have comment viewable by "OLD" project role on some issues, and now those comments disappear on the issues. (This is the expected behavior, since the project role does not exist anymore)
4. You want to retrieve the comments and move it to the new project role "NEW"
5. You know at least one issue's issue-key and comment detail which is related to the project role "OLD".

Before you begin:

⚠ Always back up your data before performing any modification to the database. If possible, try your modifications on a test server.

Steps to recover the comments:

Step 1:

You should know any issue's issue key which is having this problem. For example, "TEST-1" in this case, then you can run the following query in your MySQL database:

```sql
select * from jiraaction where issueid = (select id from jiraissue where pkey = 'TEST-1')
```

This is for checking the rolelevel for the comment which has been viewable by project role "OLD" (The rolelevel and old comment for "OLD" still exists in your database even after you have deleted the project role) Please see the attachment below:

<table>
<thead>
<tr>
<th>id</th>
<th>rolelevel</th>
<th>author</th>
<th>actiontype</th>
<th>actionbody</th>
<th>history</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10002</td>
<td>admin</td>
<td>comment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10001</td>
<td>admin</td>
<td>comment</td>
<td></td>
<td>10101</td>
</tr>
<tr>
<td>3</td>
<td>10000</td>
<td>admin</td>
<td>comment</td>
<td></td>
<td>10101</td>
</tr>
</tbody>
</table>

In this example, the rolelevel is 10101

Step 2:

Find out the rolelevel from last query and run:

```sql
update jiraaction set rolelevel=(select id from projectrole where name='NEW') where rolelevel=10101
```
This is for updating all the comments which have been restricted viewable by "OLD" to the new project role "NEW".

**How To Remove the Message 'Some issue types are unavailable due to incompatible field configuration and/or workflow associations.' from Edit Issue Screen**

Currently, JIRA will shows the 'Some issue types are unavailable due to incompatible field configuration and/or workflow associations' when editing Issue Type in Edit Issue Screen. User might puzzled by this message as they are unaware of the configuration that is done by Administrator. The reason of this message is discussed in [here](#). This message can be suppress by:

1. edit the parameter `issue.field.issuetype.incompatible.types` in the `JIRA_Install/atllassian-jira/WEB-INF/classes/com/atlassian/jira/web/action/JiraWebActionSupport.properties`
2. remove the content of the `JIRA_Install/work` directory
3. restart JIRA