# Crucible Documentation

1. bookmarks .......................................................... 5
2. Crucible Documentation Home ........................................ 5
   2.1 Crucible 101 ................................................................ 5
   2.2 About Crucible ........................................................ 11
       2.2.1 Background Reading ............................................. 12
   2.3 Crucible User's Guide .................................................. 12
       2.3.1 Getting Started with Crucible ................................. 13
       2.3.2 Using the Crucible Screens ................................. 20
           2.3.2.1 Browsing Source Files ............................... 22
           2.3.2.2 Using the Dashboard ................................. 23
               2.3.2.2.1 Browsing Your Reviews ................. 24
               2.3.2.2.2 Browsing Reviews, Source and Issues Activity 26
               2.3.2.2.3 Viewing Your Favourites .......... 29
           2.3.2.3 Crucible Icons ........................................... 31
           2.3.2.4 Viewing People's Statistics ....................... 32
           2.3.2.5 Browsing Projects ..................................... 35
               2.3.2.5.1 Viewing Project Statistics ............ 36
           2.3.2.6 Searching in Crucible ............................... 38
           2.3.2.7 Viewing Reports ...................................... 43
               2.3.2.7.1 Review Coverage Report ............... 45
           2.3.2.8 Browsing All Reviews ............................... 49
       2.3.3 Changing your User Profile .................................. 50
       2.3.4 Roles and Status Classifications ......................... 54
       2.3.5 Conducting a Review ........................................ 55
           2.3.5.1 Creating a Review .................................... 55
               2.3.5.1.1 Creating a Patch Review .......... 55
               2.3.5.1.2 Creating a Review within Crucible .... 62
               2.3.5.1.3 Creating a Review from FishEye .... 65
               2.3.5.1.4 Creating a Review from JIRA .... 67
               2.3.5.1.5 Creating a Review from a URL .... 68
               2.3.5.1.6 Creating a Snippet Review .......... 68
           2.3.5.2 Selecting the Files for the Review ............... 69
               2.3.5.2.1 Iterative Reviews .......................... 76
           2.3.5.3 Adding Reviewers ...................................... 76
           2.3.5.4 Issuing a Review ........................................ 78
           2.3.5.5 Performing the Review ............................... 80
The 15 most recent bookmarks in Crucible 2.5
There are no bookmarks to display.

Crucible Documentation Home

Crucible 2.5.x

User’s Guide

The Crucible User’s Guide is for developers, project managers, testers – anyone who uses Crucible. New to Crucible? Start by exploring the Crucible screens and creating a project, configuring repositories and refer to best practices for crucible configuration.

Administrator’s Guide

The Crucible Administrator’s Guide is for people with Crucible administration rights. It will help you set up Permission Schemes, email notifications, and JIRA integration. Admin tasks such as backup are also covered. You may also find the Knowledge Base, FAQ and Crucible Forum useful.

Installation Guide

The Crucible Installation Guide is for people who are installing Crucible for the first time. Check the supported platforms, then download and install Crucible. Where to next? If you are using other Atlassian products, take a look at the Integration Guide.

Upgrade Guide

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

Developer Resources

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

Crucible 101

Welcome to Crucible 101, an introductory guide to Crucible and a tour of the most interesting Crucible features. Use this page to guide your evaluation process or quickly get up to speed with Crucible.

<table>
<thead>
<tr>
<th>Crucible 101</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thanks for taking the time to try Crucible. To help you make the most of your time, try these easy instructions for configuring and using Crucible.</td>
</tr>
<tr>
<td>Software developers are the intended audience for this document.</td>
</tr>
</tbody>
</table>
Getting Crucible Up and Running

Carry out these steps to set up Crucible and you will be ready for your first code review in no time. **Setting up Crucible takes less than half an hour.**

Install

Basic installation is a breeze. (click to expand)

First things first, if you haven’t already got Crucible up and running carry out the following steps:

1. Download Crucible from the Atlassian Download Center.
2. Unzip the downloaded package to the desired installation directory.
3. Navigate to the `bin` sub-directory of your installation directory. Type/execute `run.bat` (under Windows) or `./run.sh` (under Linux) to launch Crucible. A Crucible console window will appear, listing the startup progress. Give it a few minutes to complete.
4. On the same machine go to [http://localhost:8060/](http://localhost:8060/) (or use the hostname or IP address in place of localhost from another machine). You will be prompted to enter a license key and set the admin password.

There are a lot of installation options. To learn more, see the complete documentation. It should only take you a few minutes to have a running instance of Crucible. If you are still struggling after twenty minutes please contact our support team for assistance. They’re ready to help at any time.

Connect to Your Repository

Crucible standalone supports SCM repository systems. (click to expand)

Once installed, you need to connect Crucible to a Source Code Management (SCM) repository, so that you can get reviewing right away. You can instantly connect to the following SCMs:

- Subversion
- Perforce

Using standalone Crucible, your repositories will be accessed only on-demand. However if using Crucible with FishEye you get much deeper SCM integration for Subversion, Git, Perforce, IBM ClearCase and CVS. FishEye allows advanced searching and rapid access to repository contents (it also requires pre-scanning your repositories). Note that your Crucible evaluation license enables FishEye.

See the configuration documentation for more.

Set Up Your Crucible Project(s)

Work streams in Crucible are organised into projects. (click to expand)

Your Crucible instance must have at least one project, in a fresh install this is named “Default Project”. It is a good idea to rename this to something more personal. If you plan on having multiple projects, you might also want to change the review key when renaming the Default Project (it defaults to ‘CR’), this should **not** be the same as any other keys (e.g. JIRA project, Bamboo plan) that you use. At Atlassian we prefix our JIRA project keys with CR-, e.g. the FishEye project’s key is CR-FE, but you can use whatever you like.

Crucible projects let you organise your reviews into logical groups, however they also serve several other purposes:

- Access control — is set at the project level
- Permission schemes — which roles can do what
- Default participants — e.g. auto add the team lead to every review
- Default length — set the number of week days to set the due date for a new review

You can also associate JIRA issue tracker(s) with your project by configuring JIRA Integration in Crucible.

Configure Mail (optional)

Email notifications are a great way to keep up with Crucible activity. (click to expand)
Crucible can deliver email feeds based on user-configured parameters. For example, you can watch the contributions of a specific colleague or changes to a branch of code you’re working on. To take advantage of this feature, you will need to tell Crucible about your SMTP server.

You can also configure custom templates for notification email messages and fine-tune the frequency of email delivery, for example, choosing a daily digest.

Move onto the next step to learn about Crucible's integration with JIRA, Atlassian's award-winning issue tracker.

**Configure JIRA (optional)**

- Crucible integrates with JIRA, Atlassian's enterprise issue tracker. (click to expand)

  1. If you already have a JIRA server set up, you can set up integration between your JIRA server and Crucible. Once established, activity on relevant JIRA issues will appear in the Crucible activity stream. JIRA issue keys will also be hyperlinked and a small information window will load when you mouse-over JIRA issue keys.
  2. Visit the Crucible documentation for instructions on how to set up JIRA integration in Crucible.

**Setup Complete!**

- Your Crucible instance is now established. (click to expand)

Congratulations! Our support records indicate that over 80% of support calls happen during this installation phase. Once you have made it this far, the rest of the evaluation ought to flow smoothly.

There are heaps of tweaks and configuration options that you can experiment with. Check out the Crucible Administrator's Guide for more information.

Otherwise, move on to the next step and launch right into lightweight code reviews.

---

**Learning the Basics**

Now that you've got Crucible set up, you can start using Crucible's code review features in earnest. Creating reviews, pre-commit and post-commit reviews, notifications, and much more.

**Creating Reviews**

**Reviewing Quick Start**

- Jump into code reviews instantly with Crucible. (click to expand)

  If you are getting familiar with Crucibles structure - roles, workflow and parts of a code review please refer to the following document before moving forward.

Using Crucible is easy and lightweight, facilitating rapid and effective reviews. Jump right in and create your first review now.

There are many carry out reviews with Crucible. In some form or another you need to go through the following steps. Each step is linked to documentation, but you probably won’t need it.

- **Create the review** - Select "Create Review" from the "Tools" menu in the top right of any page.
- **Add content to the review** - Choose some stuff to review, either from a repository you have configured or by uploading something.
- **Choose reviewers** - if you haven't added users or [configured ldap] you won't be able to do this, you'll get a warning when you try to start the review, but you can ignore that.
- **Start the review** - click the start button in the create dialog or in the top right hand corner of the review page
- **Add some comments** - use one of the many "add a comment" links or simply click on the lines of source your want to comment on
- **Summarise and close the review** - when you are finished with a review, it needs to be closed. Summarise is the state between "under review" and "closed", most of the time you will summarise and close at the same time. Alternatively you can abandon a review, which effectively moves it to the trash.

Now that you've got a feeling for the basic review concepts, its worthwhile taking a moment to customise your instance.

---

**Pre and Post Commit Reviews**
Create reviews before or after you check-in. (Click to expand)

Pre commit
Crucible allows you to review a change before it has been committed. To do this, you upload a patch file to the 'Patch' tab (or paste it in as text) when creating a review. You must first generate this patch file from your repository, using either commands built into your IDE, or via the repository command-line tools.

- Create a new review. From the 'Tools' menu in Crucible, select 'Create Review'.
- Click 'Pre-Commit - Upload a patch file to be reviewed'. The 'patch upload' dialog appears. Click 'Browse', locate your file, then click 'Upload'. Crucible will now search for matches in the files in its database. Crucible will analyse all the paths in the patch, find the branches containing all those paths, then anchor the patch to the trunk or the branch with the most recent commit activity.

Learn more about pre-commit (patch) reviews here.

Post commit

- Create a new review. From the 'Tools' menu in Crucible, select 'Create Review'.
- To add files to a review, click the 'Files' option on the left navigation bar of the Manage Files dialog. The Files view opens.

Learn more about post commit reviews here.

Creating Reviews Fields

- Fill in the fields that are relevant for your review. (click to expand)
  - When adding reviewers, Crucible will suggest reviewers who have worked on the relevant files, but also don't have a big review backlog.
  - You can set a specific due date for each review as you create them, or set a default duration for each review under a project.
  - You can easily link to JIRA issues in reviews and associate reviews with JIRA issues.

Working with Reviews

Comments turned into defects

- Identify defects in your code and tag them in Crucible. (click to expand)
  - You can easily highlight defects in the code under review by click on the 'Defect' checkbox when commenting.
  - All 'Defects' can be classified with a priority
  - Crucible tracks the time you have spent on each file in a review and also your progress through the review.

Context Windows

- A number of context windows appear when hovering your mouse over links. (click to expand)
  - Try hovering your mouse over the following links in Crucible to see the context windows:
    - Crucible usernames,
    - JIRA Issue Keys (when using JIRA; see the documentation).
    - CSIDs (when using FishEye; see the documentation).

Activity & People

Crucible adds a modern social web dimension to the usually impersonal data stored in your Source Code Management (SCM) repository.

Activity Streams

- You can see commits and updates from the users in Crucible rolling by. (click to expand)

  This information appears as a stream on the Dashboard and other index pages, sorted chronologically showing you the latest changes. Updates can be viewed as an inline stream or RSS feed. See the documentation for more.

People Lists

- In Crucible, you can view useful updates and statistics from your team. (click to expand)
On the People index page, you can see the commit history in global lines of code (LOC) that each person has contributed (expressed as a line graph) and their total number of commits. Also, the most recent piece of activity is shown as a clickable item. See the documentation for more.

**People Pages**
- Each person who makes code changes has a page. (click to expand)

You can click on a person's name to see detailed information about their code reviews, showing details of their work and summaries of their activity. Additionally, you can see their work on tracked issues and additions to the repository if using Crucible with FishEye and JIRA integration is set up. See the documentation for more.

**Your Personal Dashboard**
- See your own work at a glance and a stream of work items that are relevant to you. (click to expand)

Click the 'Dashboard' tab to see a stream of all your own activity; your reviews; your personal code commits (if you are using FishEye); and your tracked issue updates (if you are using JIRA). See the documentation for more.

**Favourites as Bookmarks**
- Everything in Crucible can be bookmarked. (click to expand)

In Crucible, you can add the following items as favourites:
- Reviews,
- Review Comment Threads,
- Projects,
- People,
- Changesets,
- Files or Folders,
- Repositories.

Once items are added to your favourites, a list is created. You can then view it or see a stream of all activity relating to your favourites.

See the documentation for more.

**Subscribe to Crucible Updates**
- Keep track of Crucible activity when your Crucible session is closed. (click to expand)

- Configure your Email preferences.
- Check out the handy Blocker report and JIRA Blocker Reports. Every now and then, someone in the team can become a bottleneck in the review process. The review blocker reports help by identifying team members that have a lot of reviews waiting in their inbox.
- Configure your personal dashboard and set up RSS feeds.

**Search**

**Search Options**
- Find a specific review or all of them with easy search options. (click to expand)

**Searching for Reviews:**
- Using the Filters Navigation Bar
  - 'Everyone’s Reviews’ Filters
  - Custom Filters
  - Review State Filter
  - Comment Search

See the documentation for more.

**Searching for comments:**
Crucible give you the ability to search and filter review comments. Search or filter by comment content, defect classification, specific role(s), date and much more. Learn more here.
When you type in the search box, matches are instantly shown below. (click to expand)

You can quickly find what you are looking for by typing one word or part of the name of what you are looking for. Crucible’s Quick Nav feature will immediately show matches and suggestions below, before you've even pressed Enter or activated a proper search. Try typing a review's issue ID to go directly to that page.

Advanced Features

Crucible will allow you to go beyond commenting on source code. Iterative reviews, reports and auditing, plugins, JIRA integration, IDE integration, and much more.

Iterative Reviews

Files are always up to date

- Crucible allows you to review iteratively, so you can keep focused on the most recent file. (click to expand)
  
  - When files in a review become outdated, Crucible informs you with the File Outdated Menu. You can then easily update the file with one click (Iterative Review).

Review Multiple Revisions

- Select the files that you want to review
  
  - When adding files to a review, you can also review multiple revisions of a file.

Reporting

Plugin Reports

- Crucible is extensible, allowing you to create your own reports and sent info to other systems. (click to expand)
  
  You can create your own report as a plugin and have it appear in the menu. For example, the ‘Review Blockers’ report that appears in the user interface is actually a plugin, added in to Crucible. You can use this plugin as a basis to create your own custom reports and add them in to Crucible. See the Crucible developer documentation for more.

Using the REST API

- Extend Crucible. (click to expand)
  
  With its own API (Application Programming Interface), Crucible is extensible. If you need Crucible to do more than it ships with and you've got programming chops of your own, the API allows you to build extensions to the application to suit your needs.

  See the API documentation for more.

Integrate with other systems

Integrate with JIRA issue Tracker

- Link your code reviews directly to your JIRA issues
  
  There are several aspects to the JIRA and Crucible integration. Each link below will take you into the details of each.

  Create Reviews from inside of JIRA:
  Learn how to create reviews from JIRA here

  Associate and create JIRA issues for your reviews:
  
  - 1. Create a JIRA Issue for your Review
  - 2. Create Your Review and Link it to a JIRA Issue
  - 3. Make a Comment or Defect on the Review
  - 4. Click to create a JIRA Sub-Task
  - 5. Resolve the JIRA issue through Crucible
**Integrate with your IDE**

Create and work with reviews in Eclipse and IntelliJ

Using the free Atlassian IDE connector you can create, work with, manage, and monitor all your Crucible reviews directly from Eclipse and IntelliJ.

Some highlights include (this is a sample from the Eclipse Connector):

Below are the highlights of viewing and acting upon Crucible reviews within Eclipse.

- Receive notification of new and updated reviews.
- View the review details in an Eclipse editor.
- Open the file under review, at the commented source code line.
- Create a post-commit review, a patch review or a pre-commit review.
- Summarise and close the review and perform other workflow actions.

**Tips and Suggestions**

Get reviewing quicker with these suggestions

- Wiki markup in your review comments - this is a way to make your comments stand out.
- Keyboard shortcuts - Keyboard shortcuts can help save some time on creating, editing, and working with reviews.

**Configuration Tips**

Unleash the flexibility of Crucible by configuring it exactly to your needs. (click to expand)

- In Crucible, you can set configure defaults per-project for review duration, review objectives and reviewers.
- Crucible allows you to customise your own permission schemes.
- Crucible lets you disable the moderator role, for lightweight, faster peer-to-peer reviews.

Thanks for taking the time to evaluate Crucible using this guide. To help continue your journey, our support staff are always ready to answer your questions in the Crucible Forum, or solve specific problems at our support portal http://support.atlassian.com

**About Crucible**

Crucible is a powerful addition to FishEye, making it easy to review code changes, make comments, and record outcomes in an efficient, distributed, and process-neutral way.

**Introduction**

Crucible is a tool that facilitates code review. It can be as valuable to organisations that already have a formal inspection process as it is to teams that don't review at all.

Regular peer review is a proven process with demonstrable return on investment (ROI). The benefits vary from team to team but commonly include:

- Identifying bugs and defects early.
- Sharing expertise and encouraging knowledge transfer.
- Improving system-wide knowledge.
- Encouraging adherence to internal standards and style conventions.
- Identifying individual strengths and weaknesses.

One of the less apparent, but nonetheless important, benefits that comes from a transparent code review process is that quality improves simply from the knowledge that code may be critically reviewed. Developers take more care with style, readability, comments, and commit-messages because their peers are going to see them.

Despite these and many other clear benefits, code review is often seen as ‘impractical on time sensitive projects’, ‘only valuable in large teams working on mission critical applications’, or at worst ‘a total waste of time foisted on developers by management’. Formal code review can feel like an expensive use of time, because the review process can:

- Be burdened by excessive paperwork and other administration.
• Interrupt your current task and make you less productive.
• Include meetings where participants fail to prepare, so that the meeting becomes a walkthrough rather than a critical review.
• Become an ego battle or point-scoring exercise dominated by a vocal minority.

These issues do not affect the immense potential value of code review. They are simply problems with some review processes.

Crucible’s mission is to streamline the process aspects so development teams can access the benefits. Crucible achieves this by:

• Making reviews asynchronous.
• Bringing reviewing to your desk (wherever that might be).
• Eliminating most of the administration.
• Limiting the ability for individuals to dominate the dialogue.
• Providing an archival record of reviews.

Crucible increases the quality, quantity, and frequency of code reviews thereby reducing bugs, helping knowledge sharing and fundamentally improving system quality.

Starting Points

Visit the Crucible Feature Tour to understand how Crucible can benefit you.

You can run Crucible with a FishEye-compatible source code repository set up, such as CVS, Subversion, or Perforce. For more information, please read the FishEye documentation.

Read the Installation Guide to get started quickly.

For Crucible troubleshooting, see the FAQ.

Background Reading

The following resources are recommended for background reading on peer code reviews:

• White paper on effective code review by Karl Wiegers.
• Book, Peer Reviews in Software: A Practical Guide by Karl Weigers.
• Software Engineering Institute web page: Software Inspections.
• NASA Software Assurance Technology Center web page: Software Formal Inspections.

Crucible User’s Guide

This page is an index of the content in the Crucible User's Guide. Click on a link below to see the desired page.

• Getting Started with Crucible
• Using the Crucible Screens
  • Browsing Source Files
  • Using the Dashboard
    • Browsing Your Reviews
    • Browsing Reviews, Source and Issues Activity
    • Viewing Your Favourites
  • Crucible Icons
  • Viewing People’s Statistics
  • Browsing Projects
    • Viewing Project Statistics
  • Searching in Crucible
  • Viewing Reports
    • Review Coverage Report
  • Browsing All Reviews
• Changing your User Profile
• Roles and Status Classifications
• Conducting a Review
  • Creating a Review
    • Creating a Patch Review
    • Creating a Review within Crucible
    • Creating a Review from FishEye
    • Creating a Review from JIRA
    • Creating a Review from a URL
    • Creating a Snippet Review
  • Selecting the Files for the Review
    • Iterative Reviews
  • Adding Reviewers
Getting Started with Crucible

This page contains a basic overview of Crucible workflows, followed by a simple example showing a code review between two people.

Crucible is a flexible application that caters for a wide range team sizes and work styles. You will need to know about the basic roles used in Crucible.

Roles:
There are several roles that review participants can take up:

- **Author**: Usually the creator of the code; the person who will act on the review's outcome.
- **Reviewer**: A participant that will comment on the source files in the review, raising points and discussion on the work that was done.
- **Moderator**: Usually the person who starts the review and is responsible for deciding the outcomes and closing it.

You will also need to understand how workflow is conducted in Crucible. This is configurable, but the most basic example follows.

Crucible Workflow:
There are a number of different ways in which you can use Crucible for code reviews. The following diagram shows the basic workflow that applies to most Crucible code reviews.

*Diagram: Workflow for One-to-One Reviews*
Next, we explore the workflow in a two-person code review in Crucible.

**Example Workflow: Two Participant Code Review**

This is a simplified set of instructions for executing a one-to-one review involving two people. In this example, the code author wears "three hats", acting as **review creator**, **moderator** and **code author**, managing the review process as well as taking final responsibility for closing the review. The second person is the reviewer.

On this page:

- Example Workflow: Two Participant Code Review
  - 1. The Author Starts the Review
  - 2. The Reviewer Comments on the Code
  - 3. The Author Responds to the Comments
  - 4. The Author Closes the Review

For instructions on Crucible workflow with more than two people, see [this page](#).
1. The Author Starts the Review

To begin, the code author sets up the review. There are a number of ways to do this, but for this example, the author starts from the FishEye Source view of the file he wants to review:

Screenshot: Opening a review from the FishEye Source view

![Screenshot](image)

From the FishEye Source view, the author clicks the ‘Reviews’ drop-down menu above the source view, then selects ‘Create New Review’. If there are multiple projects, the Project Selection dialogue opens.

Screenshot: The Project Selection dialogue

![Screenshot](image)

In the Project Selection dialogue, you are prompted to choose a project for this review from the drop-down list. Once the selection is made, the author clicks the ‘Create New Review’ button. The Edit Review Details dialogue opens, where the author can create and issue the review.

Screenshot: Creating a review in the Edit Review Details dialogue

![Screenshot](image)
In the Edit Review dialogue, the author enters information needed for the review. This includes entering a title and description for the review, selecting reviewers, a due date and the key for a related JIRA issue (if any). The project, moderator and author are pre-selected (for this example, the author should select himself as a moderator).

The author can also add more content to the review, if they wish, by clicking the 'Add Content' button. See Selecting the Files for the Review.

When finished, the author clicks 'Save'. The review will now be created in a draft form.

Screenshot: A new Crucible review

The draft review opens. In the draft stage, the author can check the contents of the review files to ensure they are correct and put in any notes for reviewers as comments. During the draft phase, no notification email is sent out to reviewers. Once the author is finished with the draft phase, he clicks 'Start Review'.

The review will now be started and notification email will go out to all participants. Crucible will now send out an email notification to all the participants. This lets them know that the review is under way and prompts them to take action, providing a URL for direct access to the review. (You can also subscribe to an RSS feed.)

2. The Reviewer Comments on the Code
The reviewer will receive an email from Crucible (or an RSS feed update) with a link that they can follow to the review.

Screenshot: A Crucible review notification email

CR-FE-1752 state changed to Review by Geoff Crain

---


Title: Test review for Ed

Statement of Objectives:

State: Review

Author: Geoff Crain
Moderator: Geoff Crain
Reviewers: (1 active, 0 completed*)
  Edwin Dawson

When the reviewer clicks the link in the notification email, the Crucible Review screen opens.

Screenshot: The Crucible Review screen
On the Crucible Review screen, the code changes under review are displayed. The reviewer clicks file names to expand the code for in-line reviewing. As the reviewer reads the changes, they can simply click on any line to enter a comment there (multiple lines can be selected by clicking and dragging).

The reviewer clicks the ‘Post’ button when each comment is finished.

The reviewer repeats this process for all files in the review. Reviewers can leave the session and resume it later; their work is automatically saved.

When the reviewer has finished their code review work, they click the ‘Complete’ button.

By default, an email is sent to participants every time a comment is posted. This is an individual setting. Each reviewer can configure their own profiles to adjust the list of events that will trigger email notifications.

3. The Author Responds to the Comments

During the review process, the author/moderator can also make contributions, responding to reviewer comments and making corrections.

Screenshot: Comment threads in Crucible
4. The Author Closes the Review

When all reviewers have "Completed" their reviews, the author/moderator is notified via email. The author/moderator clicks the link in the notification email, returning to the Review screen.

The author/moderator will then add any final comments, then click the 'Summarise' button when finished. The Crucible Summarize Review screen opens.

Screenshot: Summarizing a review in Crucible

On the Crucible Summarize Review screen, the author/moderator enters an optional summary of the review's results, then clicks 'Close' button. This closes the review, signalling the end of work. A final email notification will be sent to the review participants, informing them that the review is now closed. The closed review screen will load, displaying the summary and archiving the completed review as read-only.
If the author/moderator ever needs to resume work on the closed review, they can simply click ‘Reopen’ when viewing this screen. Doing this will return the review’s status to “open”.

For more information on workflow in Crucible and best practices for code reviews, see Requesting and Conducting a Review.

**Using the Crucible Screens**

This page contains an overview of the Crucible interface and the actions that can be carried out in the application.

**On this page:**

- Tour of the Crucible Interface
- Left Navigation Sidebar
  - Related Links

**Tour of the Crucible Interface**

When you first log in to Crucible, the Dashboard Screen opens, as shown in the screenshot below. This view shows recent general activity in Crucible.

*Screenshot: The Dashboard Screen in Crucible*
The table below explain the top-level tabs in the Crucible User Interface. Click on the name of a tab for more information.

<table>
<thead>
<tr>
<th>Element name</th>
<th>Function</th>
<th>Appears</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dashboard Tab</td>
<td>Displays reviews and system activity related to you.</td>
<td>All screens.</td>
</tr>
<tr>
<td>Source Tab</td>
<td>Displays contents of connected source repositories.</td>
<td>Only when FishEye is used with Crucible.</td>
</tr>
<tr>
<td>Source Dropdown Menu</td>
<td>Displays a list of links to repositories recently visited by the user</td>
<td>Only when FishEye is used with Crucible for logged-in users.</td>
</tr>
<tr>
<td>Projects Tab</td>
<td>Displays reviews and content from specific projects.</td>
<td>All screens.</td>
</tr>
<tr>
<td>Projects Dropdown Menu</td>
<td>Displays a list of links to projects recently visited by the user</td>
<td>All screens for logged-in users.</td>
</tr>
<tr>
<td>People Tab</td>
<td>Displays metrics on the users of the Crucible instance.</td>
<td>All screens.</td>
</tr>
<tr>
<td>People Dropdown Menu</td>
<td>Displays a list of links to user pages recently visited by the user</td>
<td>All screens for logged-in users.</td>
</tr>
<tr>
<td>Reviews Tab</td>
<td>Allows you to search and report on reviews.</td>
<td>All screens.</td>
</tr>
<tr>
<td>People Dropdown Menu</td>
<td>Displays a list of links to reviews recently visited by the user as well as links to the Crucible Inbox and Outbox</td>
<td>All screens for logged-in users.</td>
</tr>
</tbody>
</table>

**Left Navigation Sidebar**

The navigation bar at the left of the screen applies specific filters to what is shown in the centre pane. See the page on Browsing Your Reviews for more information.

The left navigation sidebar can be hidden or displayed by clicking the blue 'information' icon at the top left of the sidebar.

**Related Links**

- Browsing Your Reviews
- Browsing Source Files
- Browsing Projects
- Viewing People's Statistics
- Viewing Reports
- Searching in Crucible
Browsing Source Files

When FishEye is installed with Crucible, you have the additional ‘Source’ tab available in the navigation tabs at the top of the screen.

To browse source files,

1. Click the ‘Source’ tab. The ‘Repositories’ view will be displayed, showing summary information if you have multiple repositories set up. See the ‘Viewing all repositories’ screenshot below.
2. Click the desired repository to view its contents. See the ‘Viewing a repository’ screenshot below.
3. Browse the repository for the desired source file using the directory tree in the left menu. Click the file that you want to view. The file will be displayed in the main panel. See the ‘Viewing a file’ screenshot below.
4. You can view various information about the file, as outlined in the table below:

<table>
<thead>
<tr>
<th>Sub-Tab Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revisions</td>
<td>When viewing a file, shows the latest revisions of the file.</td>
</tr>
<tr>
<td>Files</td>
<td>When viewing a folder, shows the contents of the directory.</td>
</tr>
<tr>
<td>Activity</td>
<td>Shows recent activity on the item. There are a number of sub-options here:</td>
</tr>
<tr>
<td></td>
<td>* All Activity — The default view, showing commits, reviews and JIRA issues.</td>
</tr>
<tr>
<td></td>
<td>* Commits — Shows commits in the activity stream.</td>
</tr>
<tr>
<td></td>
<td>* Reviews — Shows review activity in the activity stream.</td>
</tr>
<tr>
<td></td>
<td>* Scroll to Changeset — Opens the changeset ID specified in the text field (press Enter to carry out the action).</td>
</tr>
<tr>
<td></td>
<td>* Filter — Applies constraints to the current activity stream.</td>
</tr>
<tr>
<td></td>
<td>* Show Revisions — If this is selected, then changeset items are automatically expanded to show modified files.</td>
</tr>
<tr>
<td></td>
<td>* Earlier Activity (Left Arrow icon) — Loads a page of earlier activity.</td>
</tr>
<tr>
<td></td>
<td>* Later Activity (Right Arrow icon) — Loads a page of later activity.</td>
</tr>
<tr>
<td>Users</td>
<td>Shows the commit history of the different users that have committed changes on the item.</td>
</tr>
<tr>
<td>Reports</td>
<td>Shows activity charts for the item. Various chart options can be selected in the left navigation bar.</td>
</tr>
<tr>
<td>Source</td>
<td>Shows the contents of the file.</td>
</tr>
<tr>
<td>Query</td>
<td>Allows you to run an advanced search.</td>
</tr>
</tbody>
</table>

To download files, firstly click through the desired file. From there, you will see a control bar directly above the code content which contains the ‘FishEye’ item. Clicking this leads to a drop-down menu where ‘Download Raw File’ is available. You can use this to download the file in context only.

Screenshot: Viewing all repositories
Using the Dashboard

The Dashboard is the first screen that you see when you log into FishEye/Crucible. It is the home page for the instance and can be accessed by clicking the FishEye/Crucible icon or by clicking the 'Dashboard' tab at the top.
The Dashboard itself has a sidebar and an information pane with three tabs.

- **'My Reviews'** sidebar — The 'My Reviews' sidebar contains information about reviews/snippets that you are involved with, e.g. how many reviews require your approval. Read more about the sidebar in Browsing Your Reviews.
  
  - Click on any of the links in the sidebar to navigate to that information in the 'Reviews' tab of the Dashboard.
  
  - Hover your mouse over the sidebar and click the collapse icon (��) to hide any of the sections. Click the expand icon (▌) to expand any collapsed sections.
  
  - Click the 'i' icon (ℹ️) to hide/show the entire sidebar.

- **'Activity'** tab — The 'Activity' tab displays reviews, source and issues activity in your FishEye/Crucible instance. This includes all activity by you, as well as all activity in repositories, directories and users that you have selected as favourites. Read more about the 'Activity' tab in Browsing Reviews, Source and Issues Activity.

- **'Reviews'** tab — The 'Reviews' tab displays all reviews that you are involved with. Read more about the 'Reviews' tab in Browsing Your Reviews.

- **'Favourites'** tab — The 'Favourites' tab displays all of the Crucible code reviews, changesets, files, people and repositories that you have selected as favourites. Read more about the 'Favourites' tab in Viewing Your Favourites.

### Browsing Your Reviews

This instructions on this page describe how to browse your reviews in Crucible using the Dashboard sidebar. The links on the Dashboard sidebar will present different lists of reviews in the 'Reviews' tab of the Dashboard.

The 'Reviews' tab at the top of the screen (not in the Dashboard) provides an additional shortcut to view all reviews (or a custom filtered list, if you have set up a filter) and to generate review blocker reports. See Browsing All Reviews for more information about this.

On this page:

- Browsing Your Reviews
- Browsing Everybody's Reviews
- Browsing a Custom List of Reviews

**Browsing Your Reviews**

To browse your reviews
1. Click the 'Dashboard' tab at the top of the page. The Dashboard will be displayed (see screenshot below).
2. Click any of the links under the 'My Reviews' or 'My Snippets' sections in the Dashboard sidebar, to open the desired reviews list in the 'Reviews' tab of the Dashboard.
   - 'Inbox' — Click to show all reviews in 'To Review', 'To Summarize', 'In Draft' and 'Require My Approval' states.
   - 'To Review' — Click to show all reviews where you are a reviewer and haven't yet completed your review work.
   - 'To Summarize' — Click to show all reviews where you are a moderator and haven't yet summarised and closed the review.
   - 'In Draft' — Click to show all reviews that you have created but have not yet been moved to the 'Approval' state or the 'Require Approval' state.
   - 'Require My Approval' — Click to show all reviews where you are a moderator and need to approve the review.
   - 'Outbox' — Click to show all reviews in 'Out for Review' and 'Completed' states.
   - 'Out for Review' — Click to show all reviews that you are a participant of, that have review work that is yet to be completed by other reviewers.
   - 'Completed' — Click to show all reviews that you are a participant of, and have been completed.
   - 'Archive' — Click to show all reviews in 'Closed' and 'Abandoned' states.
   - 'Closed' — Click to show all reviews that you are a participant of, that have been summarised and closed.
   - 'Abandoned' — Click to show all reviews that you are a participant of, that have been abandoned. You may wish to delete these reviews.
   - 'My Open Snippets' — Click to show all open snippets created by you.
   - 'My Snippets' — Click to show all snippets created by you.

**Tip:** Hover your mouse over the sidebar and click the ▼ icon or ▲ icon to expand or collapse a section in the Dashboard sidebar, e.g. 'My Reviews' section. Click the  icon next to 'My Reviews' to hide/show the entire sidebar.

**Browsing Everybody’s Reviews**

To browse everybody’s reviews

1. Click the 'Dashboard' tab at the top of the page. The Dashboard will be displayed (see screenshot below).
2. Click the 'Reviews' tab next to the 'Activity' tab on the Dashboard (not at the top of the page). Note, you can also click the 'Reviews' tab on the Dashboard (next to the 'Activity' tab) to view all reviews in your Inbox.
3. Click any of the links in the Dashboard sidebar to open the desired reviews list in the 'Reviews' tab.
4. Browse reviews for all people by clicking the links under the 'Everyone’s Reviews' and 'Everyone’s Snippets' sections in the sidebar. See Browsing All Reviews for more information.

**Browsing a Custom List of Reviews**

To browse a custom list of reviews

1. Click the 'Dashboard' tab at the top of the page. The Dashboard will be displayed (see screenshot below).
2. Click the 'Reviews' tab next to the 'Activity' tab on the Dashboard (not at the top of the page).
3. Use the 'Custom Filter' or 'Comment Search' in the sidebar to search for a custom list of reviews. See Searching in Crucible for more information.

Screenshot: Browsing your reviews on the dashboard
The Dashboard has an activity stream that displays the following information:

- **reviews activity** — This includes the addition of review comments, opening and closing reviews, etc.
- **source activity** — This includes files being committed to a repository.
- **issues activity** — If you have linked a JIRA server with your Crucible project, the activity stream will also include updates to linked JIRA issues.

Your activity stream will only display information from projects, reviews, people, repositories, etc, that you have selected as favourites as well as your own activity. For more information on favourites, see [Using Favourites](#).

The following instructions describe how to view the activity stream and filter it to display specific information.

### On this page:

- Browsing All Activity
  - Browsing Source Activity only
  - Browsing Reviews Activity only
  - Browsing Issues Activity only
  - Watching an Activity Stream

### Browsing All Activity

To browse all activity
1. Click the 'Dashboard' tab at the top of the page. The Dashboard will displayed (see screenshot below).
2. By default, the 'All Activity' tab will be displayed (i.e. reviews, source and issues activity in your FishEye/Crucible instance). This includes all activity by you, as well as all activity in repositories, directories and users that you have selected as favourites.
3. Click any of the tabs under the 'Activity' tab to filter the activity stream, as follows:
   - Click the 'Commits' tab to filter the activity stream to display only source activity, i.e. commits. See Browsing Source Activity only.
   - Click the 'Reviews' tab to filter the activity stream to display only reviews activity. See Browsing Reviews Activity only.
   - Click the 'Issues' tab to filter the activity stream to display only JIRA issues activity. This tab will only display if you have connected a JIRA instance to your FishEye/Crucible instance. See Browsing Issues Activity only.
4. Click the 'Show Revisions' button to show file revisions for all files displayed in the activity stream. The button will be darkened when revisions are shown. This button can be used in any of the tabs for the activity stream.
5. Untick the 'Show My Activity' checkbox to filter out your activity from the activity stream displayed. This toggle can be used in any of the tabs for the activity stream.

Screenshot: Crucible Dashboard

Browsing Source Activity only

Source activity includes files commits to repositories that you have selected as favourites.

Click the 'Commits' tab to filter the activity stream to display only source activity (see screenshot below).

Browsing Source Activity on the Dashboard
Browsing Reviews Activity only

Reviews activity includes updates to reviews in all projects that you have selected as favourites. See Browsing Your Reviews for more information about browsing reviews.

Click the 'Reviews' tab to filter the activity stream to display only reviews activity (see screenshot below).

Browsing Reviews Activity on the Dashboard
**Browsing Issues Activity only**

Issues activity includes updates to issues in JIRA projects that are associated with your favourite Crucible projects. For more information about integrating JIRA with Crucible, see **JIRA Integration in Crucible**.

Click the 'Issues' tab to filter the activity stream to display only issues activity (see screenshot below).

**Browsing Issues Activity on the Dashboard**

![Dashboard](image)

**Watching an Activity Stream**

You can "watch" an activity stream in FishEye/Crucible. Watching the activity stream allows you to receive emails when updates occur in the activity stream. You can view all of your watches and configure the frequency of your watch emails in your user profile. See **Changing your User Profile** for more information.

Note, the option to add a watch will only be available if the administrator has enabled watches for the repository.

**To watch an activity stream,**

1. Navigate to the activity stream that you want to watch.
2. Click the 'Tools' menu and click 'Watch'. The page will reload and a watch will be set up for the activity stream (the watch icon will be coloured, not grey).
   - If you want to remove the watch, navigate the activity stream, click the 'Tools' menu and click 'Watch'. The watch will be removed (the watch icon will be coloured, not grey).

   "You can also remove watches via your user profile."

**Viewing Your Favourites**

This page contains instructions on how to view, rename and remove your existing favourites in Crucible. You can select code reviews, changesets, files, people and repositories as favourites in Crucible. This allows you to personalise the information that you see in your activity stream.

See **Using Favourites** for instructions on how to add new favourites.

**On this page:**
• Viewing your Existing Favourites
• Renaming a Favourite
• Removing a Favourite

Viewing your Existing Favourites

To view your favourites,

1. Click the 'Dashboard' tab at the top of the page. The Dashboard will be displayed.
2. Click the 'Favourites' tab on the dashboard. The code reviews, changesets, files, people and repositories that you have selected will be displayed in a list on the dashboard (see screenshot).
3. Click the favourites icon (⭐️) next to a favourite to show the dialogue for renaming (i.e. changing the 'Label') the favourite or removing the favourite. See Using Favourites for more information.

Tip: You can add favourites wherever you see star icons next to code reviews, changesets, files, people and repositories, not just on this screen.

Screenshot: Viewing favourites

Renaming a Favourite

To view your favourites,

1. Click the 'Dashboard' tab at the top of the page. The Dashboard will be displayed.
2. Click the 'Favourites' tab on the dashboard. The code reviews, changesets, files, people and repositories that you have selected will be displayed in a list on the dashboard (see screenshot).
3. Click the favourites icon (⭐️) next to a favourite to show the 'Update favourite' dialogue (see screenshot below).
4. Enter the new name for the favourite in the 'Name' field and click the 'Save label' button. The favourite will be renamed on the dashboard.

Screenshot: The 'Update favourite' dialogue
Removing a Favourite

To view your favourites,

- Click the 'Dashboard' tab at the top of the page. The Dashboard will be displayed.
- Click the 'Favourites' tab on the dashboard. The code reviews, changesets, files, people and repositories that you have selected will be displayed in a list on the dashboard (see screenshot).
- Click the favourites icon (⭐) next to a favourite to show the 'Update favourite' dialogue (see screenshot below).
- Click the 'Remove' button. The code review, changeset, file, person or repository will be removed as your favourite (the icon will be greyed out) and will not display the next time you view your list of favourites.

Tip: You can remove favourites wherever you see star icons next to code reviews, changesets, files, people and repositories, not just on this screen, not just on this screen.

Screenshot: The 'Update favourite' dialogue

### Crucible Icons

This page contains a list of Crucible icons and an explanation what each one represents in the user interface.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🌐</td>
<td>View review-level comments</td>
</tr>
<tr>
<td>🔍</td>
<td>Go to the previous comment</td>
</tr>
<tr>
<td>🔍</td>
<td>Go to the next comment</td>
</tr>
<tr>
<td>✍️</td>
<td>Add a comment</td>
</tr>
<tr>
<td>🔍</td>
<td>Go to the previous file in this review</td>
</tr>
<tr>
<td>🔍</td>
<td>Go to the next file in this review</td>
</tr>
<tr>
<td>🔍</td>
<td>Expand all files</td>
</tr>
<tr>
<td>🔍</td>
<td>Collapse all files</td>
</tr>
</tbody>
</table>
Viewing People’s Statistics

This page contains instructions on how to use the People tab in Crucible to see charts and activity from people with accounts on the system.

On this page:

- Opening the List of People
- Viewing a Person’s Activity Screen
- Viewing Charts on a Person’s Activity

Opening the List of People

To view statistics on People in Crucible, (that is, code authors, committers and reviewers) click the People tab at the top of the page. The list of all People appears.

Screenshot: List of all People in Crucible (when using FishEye with Crucible)

The list of all people shows all users that have accounts on the system. By default, each user has a unique avatar that is randomly formed from the text in their email address. Users can choose to upload their own avatar image by uploading an image to an external service such as Gravatar, which Crucible supports. See the page on Changing your User Profile.

Viewing a Person’s Activity Screen

Click on a user to see a listing of activity from them as well as charts showing statistics for their activity. The People Activity screen opens.

Screenshot: The People Activity Screen in Crucible
In the right hand pane, we can see a list of all activity that relates to this user. You can click the icons to view full commit information in FishEye, click JIRA issue names to open the work ticket on an item, click the long button to see the list of files in context or click the star icon to add an item to your favourites.

In the left hand pane, we can see charts around this activity, such as the following: number of active reviews; charted history of lines of code; code committing activity and general statistics.

Some users may not appear to have the correct number of Files Changed or LOC, despite regularly committing. In this situation, if they have committed to a directory which is not covered by the regexes in your symbolic definition (i.e. they have committed to a directory that is neither trunk, branches or tags) then that directory will be counted as part of trunk. Also note that creating tags and branches themselves does not count toward the totals.

**Viewing Charts on a Person's Activity**

To see information on a person's activity charted in detail, click the headings in the left-hand pane. Each heading will show more information on demand, when clicked. The information available and what it means is listed below.

⚠️ The Charts in this section are only available when using FishEye.

*Screenshot: People Activity Charts in Crucible*
• **Username** heading:
The username section shows the email address, then the first and latest commit dates for the person in context. It also shows username mappings from various systems if they have several usernames in play.

• **Reviews** heading:
The Reviews section shows several filters that you can click to constrain the review items shown in the right-hand pane. The options are *To Review, To Summarize, Out For Review, Open* and *Closed*.

• **Line History** heading:
The Line History section shows a graph with the number of lines committed to the repository, charted over time.

• **Commit Activity** heading:
The Commit Activity section shows three smaller charts; the first showing the volume of commits over a 52 week period; the second showing the relative number of commits on days of the week; the third showing the relative number of commits by the hour of the day when they were lodged.

• **Stats** heading:
The Stats section shows data points for the previous week and all-time. It shows number of commits, number of files changed and number of lines changed.
Browsing Projects

To browse the content in a project, click the Projects tab at the top of the page. The 'Projects' view opens (see 'The Crucible Projects Index' screenshot below).

A list of projects will be shown if there is more than one. Click the name of the desired project to open it. The 'Project Activity' page opens (see 'The Crucible Project View' screenshot below). In the left navigation bar, charts showing overall project statistics are displayed.

There are a number of sub-tabs on this page, listed in the table below.

<table>
<thead>
<tr>
<th>Sub-Tab Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| Activity     | - All Activity — The default view.  
- Commits — Shows commits in the project (visible when using FishEye).  
- Reviews — Shows reviews in the project.  
- Issues — Shows JIRA issues related to this project. (Only visible if you have set up JIRA Integration in Crucible.)  
- Show Revisions — Shows or hides revisions in the project (visible when using FishEye).  
- Earlier Activity (Left Arrow icon) — Loads a page of earlier project activity.  
- Later Activity (Right Arrow icon) — Loads a page of later project activity. |
| Reviews      | Shows recent reviews in the project. |

The Projects tab is only visible in Crucible. Read more about the definition of a project.

Screenshot: The Crucible Project View

Screenshot: The Crucible Projects Index
Viewing Project Statistics

This page explains the layout of the Project Summary page.

On this page:

- Project Name Panel
- Project Line History Panel
- Project Stats Panel
- Project Commit Activity Chart

When you click through to a Crucible Project from the Projects Tab, the ‘Project Summary’ screen opens.

Screenshot: The Crucible Project Summary Page

In the right hand pane, you can see an activity stream relating to this project. In the left hand pane, you can see various statistics charts relating to
the project in context. These appear in a reduced size until you click them, when they will expand to show more information.

**Project Name Panel**

This contains a short message explaining which Crucible Project and FishEye repositories are being accessed to show the activity stream on the page.

**Project Line History Panel**

This panel contains a chart showing the lines of code added to the repository, graphed over time.

*Screenshot: The Project Line History Panel*

![Project Line History Panel](image)

**Project Stats Panel**

This panel contains a chart showing numerical data for commits, files changed and lines change, graphed over time.

*Screenshot: The Project Stats Panel*

**Project Commit Activity Chart**

This panel contains a number of charts:

<table>
<thead>
<tr>
<th>Chart</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**52 week commits volume**
This chart shows the amount of commits, shown by week over a one year period.

**Commits by day**
This chart shows the amount of commits, graphed by day over the past week.

**Commits by hour**
This chart shows the amount of commits, graphed by hours over the past day.

**Commit calendar**
This chart shows the amount of commits (shown as darker colours to indicate more commits) graphed by month, over years that the repository has been running.

---

**Searching in Crucible**

This page contains instructions on how to use the **Reviews** tab in Crucible to perform searches.

*On this page:*

- Using the Filters Navigation Bar
  - ‘Everyone’s Reviews’ Filters
  - Custom Filters
  - Review State Filter
  - Comment Search
- Viewing Search Results
  - Review Search
  - Comment Search

*Screenshot: Search Filter Options*
'Everyone’s Reviews' Filters

Any reviews (even if the user has not participated in them) that have been created can be viewed, under the ‘Everyone’s Reviews’ section. The following options are available:

- All open reviews.
- All closed reviews.
- All reviews.

Custom Filters

To find a specific review, use the ‘Custom Filter’:

<table>
<thead>
<tr>
<th>Custom Filter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Find reviews by searching for words within the title.</td>
</tr>
<tr>
<td>Project</td>
<td>Find reviews under a particular project.</td>
</tr>
<tr>
<td>Author</td>
<td>Find reviews moderated by a particular authors.</td>
</tr>
<tr>
<td>Moderator</td>
<td>Find reviews moderated by a particular moderators.</td>
</tr>
<tr>
<td>Creator</td>
<td>Find reviews created by a particular creator.</td>
</tr>
<tr>
<td>Reviewer</td>
<td>Find reviews that are reviewed by a particular reviewer. This will default to the user logged in.</td>
</tr>
<tr>
<td>Reviewer Status</td>
<td>This is reliant on the above filter and is used to show reviews that have either been completed by the reviewer, not completed or all reviews.</td>
</tr>
<tr>
<td>Match Roles</td>
<td>To use all the above filters, choose ‘all’. To use any of the filters, choose ‘any’.</td>
</tr>
</tbody>
</table>

Review State Filter

You can use the checkboxes below with the above filters or on their own.

<table>
<thead>
<tr>
<th>Review State Filter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft</td>
<td>Reviews that are still in 'Draft' state.</td>
</tr>
<tr>
<td>Pending Approval</td>
<td>Reviews that have been moved out of 'Draft' state and are now waiting for the moderator to approve.</td>
</tr>
<tr>
<td>Under Review</td>
<td>The review is now 'Under Review'.</td>
</tr>
<tr>
<td>Summarize</td>
<td>The review is now in 'Summarize' state.</td>
</tr>
<tr>
<td>Closed</td>
<td>Reviews that are now 'Closed'.</td>
</tr>
<tr>
<td>Abandoned</td>
<td>Any reviews that are no longer relevant and can be deleted.</td>
</tr>
<tr>
<td>Rejected</td>
<td>Any reviews that a moderator has rejected.</td>
</tr>
<tr>
<td>Review needs fixing</td>
<td>A review will match this filter if the review enters into an undefined state because something went wrong with storing the review state. A moderator can use this filter to find the review and then change the state to something sensible.</td>
</tr>
</tbody>
</table>

Comment Search

To find review comments that contain particular text, use the ‘Comment Search’ box at the bottom of the filters bar. You can also click the ‘Advanced Search’ link to display the following comment-filtering options:

<table>
<thead>
<tr>
<th>Comment Search</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Find comments on reviews under a particular project.</td>
</tr>
<tr>
<td>Comment content</td>
<td>Find comments that contains the specified text.</td>
</tr>
<tr>
<td>Review PermaId</td>
<td>Find comments made on the specified review.</td>
</tr>
<tr>
<td>After</td>
<td>Find comments made after after a particular date.</td>
</tr>
<tr>
<td>Before</td>
<td>Find comments made after before a particular date.</td>
</tr>
<tr>
<td>Comment Author</td>
<td>Find comments made by a particular user.</td>
</tr>
</tbody>
</table>
You can:

- Tick the 'Defects' check-box to find comments that are flagged as Defects.
- Tick the 'Comments' check-box to find comments that are not flagged as Defects.
- Tick neither check-box (or both of them) to find all.

Find comments on reviews that are in a particular state. See Review State Filter (above).

Find defects that have been given a particular ranking (e.g. 'Major', 'Minor').

Find defects that have been marked as requiring re-review (or not).

Find defects that have been given a particular classification (e.g. 'Missing', 'Ambiguous').

The reviews are displayed in descending order of age. You can click any column heading to re-sort the search results.
**Comment Search**

The comment search results include defect classification charts in the left navigation pane.

*Screenshot: Comment Search Results*
Viewing Reports

This page contains instructions on how to use the Reports tab in Crucible to see lists of people whose action is required on open reviews. These are known as ‘blockers’.

FishEye Reports (on this page):

- Viewing the ‘Review Blockers’ report
- Viewing the ‘JIRA Blockers’ report
- Viewing the Review Coverage Report

Viewing the ‘Review Blockers’ report

To view a list of people who have open reviews assigned to them,

**To view the ‘Review Blockers’ report,**

1. Click the ‘Reviews’ tab at the top of the page.
2. Click the ‘Reports’ sub-tab.
3. Click the ‘Review Blockers’ link under the ‘Reports’ sub-tab. The ‘Review Blockers’ report will be displayed.
   - Click a user’s name to go to their ‘Activity’ screen.
   - Click a number in the ‘To Complete’ or ‘To Summarize’ column to go to the list of reviews waiting to be completed/summarised by the user.

*Screenshot: ‘Review Blockers’ Report*
Viewing the 'JIRA Blockers' report

The 'JIRA Blockers' report shows you a list of users whose action is required on open reviews, for a particular set of JIRA issues. The reviews must be explicitly linked to a JIRA issue or mention a JIRA issue key in the summary or the objectives.

To view the 'JIRA Blockers' report

1. Click the 'Reviews' tab at the top of the page.
2. Click the 'Reports' sub-tab.
3. Click the 'JIRA Blockers' link under the 'Reports' sub-tab.
4. Enter the details of your JIRA server and project, and click the 'Go' button. The 'JIRA Blockers' report will be displayed with the following information:
   - A list of JIRA issues for which one or more Crucible reviewers has not completed their review.
   - A list of users who have an incomplete Crucible review that relates to a JIRA issue.
   - A list of open JIRA issues for which a Crucible review is closed, and vice versa.
Crucible has useful reports that show you detailed statistics on review activity. The ‘review coverage’ report allows you to see how much of the code in your repository has been reviewed, which files and when. You can also access the reviews.

This feature requires FishEye integrated with Crucible.

On this page:
- Opening the Review Coverage Report
- Using the Summary Panel
- Using the Review Coverage Overview
- Using the Individual Committer Statistics Panel
  - Using the Changesets Coverage Panel

Screenshot: The Review Coverage Report
Opening the Review Coverage Report

To open the Review Coverage Report,

1. Click the 'Source' tab.
2. Select your repository. The repository you've chosen will set the scope for the Coverage Coverage report.
3. If desired, navigate down the tree to the desired path you want to view coverage on.
4. Click the Reports tab in the secondary toolbar.
5. Select 'Review Coverage Report' from the list of reports in the upper panel.

You can view coverage of any path by navigate down the tree to the desired path you want to view coverage on, before clicking on the 'Reports' tab.

Using the Summary Panel

The summary panel shows some choice metrics from your Crucible instance. The following information from your repository is arrayed:

- Overall review coverage percentage.
- Change in review coverage percentage since the last reporting period.
- Total number of reviews.
- Total number of comments.
- Total number of reported defects.
- Total number of Lines of Code (LOC).
- Total number of commits.
- Total number of committers.
- Total number of unreviewed lines.
- Total number of lines under review.
- Total number of reviewed lines.
- A ratio of the number of lines unreviewed against reviewed Lines of Code (LOC).

Screenshot: Summary Panel in the Review Coverage Report
**Using the Review Coverage Overview**

The Review Coverage Overview shows a timeline of reviews, compared against their percentage of coverage. Hover your mouse cursor over the data points on the graph to see granular information and click through to a detailed weekly report. You can click the tabs to view the coverage expressed as a percentage of lines of code, changesets or revisions.

*Screenshot: Overview Panel in the Review Coverage Report*

**Using the Individual Committer Statistics Panel**

The Individual Committer Statistics window lets you choose a user from your Crucible instance and see all the changesets by that committer.

*Screenshot: Individual Committer Statistics in the Review Coverage Report*
Using the Changesets Coverage Panel

The Changesets Coverage Panel lets you see changesets from your Crucible instance (for the time period of the report), and their level of review coverage. This information can be sorted by the columns in this view and uses colour coding to denote review coverage (listed in the table below).

**Colour Key**

<table>
<thead>
<tr>
<th>Colour</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>dark green</td>
<td>reviewed</td>
</tr>
<tr>
<td>light green</td>
<td>in review</td>
</tr>
<tr>
<td>red</td>
<td>not reviewed</td>
</tr>
</tbody>
</table>

_Screenshot: Changesets Coverage panel in the Review Coverage Report_
Browsing All Reviews

The instructions on this page describe how to browse all reviews on the 'Reviews' screen, by people or for projects that you have selected as favourites. This includes reviews that you are involved with. You can also generate reports on review blockers for all people.

The 'Reviews' tab is essentially a shortcut to viewing all reviews (or a custom filtered list, if you have set up a filter) and generating review blocker reports. If you want to view different sets of reviews, e.g. all reviews that require your approval, you need to view the 'Reviews' tab on the Dashboard instead of the 'Reviews' tab at the top of the screen. The Dashboard sidebar provides quick links for this. See Browsing Your Reviews for more information.

To browse all reviews
1. Click the ‘Reviews’ tab at the top of the screen (not the ‘Reviews’ tab on the Dashboard). The ‘Reviews’ page will be displayed, showing all open reviews (‘All Open Reviews’ will be highlighted in the sidebar) unless you have previously used a custom filter. See the screenshot below.

2. Browse reviews for all people by clicking the links under the ‘Everyone’s Reviews’ and ‘Everyone’s Snippets’ sections in the sidebar, as follows:
   - **‘Everyone’s Reviews’**— Click ‘See reviews by state...’ to expand the list of review categories in the sidebar. You can then click any of the links to view the relevant list of reviews:
     - ‘All Open Reviews’ — Click to view all open reviews, i.e. reviews that have not been summarised and closed yet.
     - ‘All Closed Reviews’ — Click to view all closed reviews, i.e. reviews that have been summarised and closed.
     - ‘All Reviews’ — Click to view all reviews, including open reviews, closed reviews and draft reviews.
   - **‘Everyone’s Snippets’**— Click ‘See snippets by state...’ to expand the list of snippet categories in the sidebar. You can then click any of the links to view the relevant list of snippet reviews, similar to the links described above.
     - ‘All Open Snippets’ — Click to show all open snippets.
     - ‘All Snippets’ — Click to show all snippets, i.e. open and closed snippets.

   If you want to browse your reviews, click the links under the ‘My Reviews’ and ‘My Snippets’ sections in the sidebar. See Browsing Your Reviews for more information.

   If you want to generate reports on review blockers for all people, click the ‘Reports’ tab. See Viewing Reports for more information.

**Screenshot: Browsing all Reviews on the ‘Reviews’ screen**

---

**Changing your User Profile**

This page contains instructions on user profile settings.

*On this page:*

- Display Settings Tab
Users can change their own account settings such as passwords, watches, email and display settings. These include the user avatar and profile image.

To view your user profile, log into Crucible, click the User Menu (labelled with your username) at the top of the screen, then select ‘Settings’.

**Screenshot: User Menu Settings**

You can change Crucible settings such as password, notifications and display settings.

To change these settings, log into Crucible and click the User Menu (labelled with your username) at the top of the screen, then select ‘Settings’.
**Display Settings Tab**

The options in this tab allow you to amend the display settings.

<table>
<thead>
<tr>
<th>Option</th>
<th>Default/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Length of tag list</strong></td>
<td>Default is 'Medium'. The option to show the list of tags for a file. This can be changed to show none ('Hide') or all ('Long').</td>
</tr>
<tr>
<td><strong>Show hidden directories</strong></td>
<td>Default is 'No'. Do not show the hidden directories within any folder lists.</td>
</tr>
<tr>
<td><strong>Show empty directories</strong></td>
<td>Default is 'Yes'. The option to see any empty directories within any folder lists.</td>
</tr>
<tr>
<td><strong>File History View Mode</strong></td>
<td>Default is 'Logical'. In Subversion repositories, Crucible is able to show all operations on a single logical file spread across a number of physical paths - i.e. operations in different branches. When this is set to 'Logical', Crucible will show all the operations across all branches. In 'Physical' mode, only the operations related to the physical path whose history is being viewed are shown.</td>
</tr>
<tr>
<td><strong>Timezone</strong></td>
<td>Default is the time zone of the Crucible server.</td>
</tr>
</tbody>
</table>

**Changelog**

<table>
<thead>
<tr>
<th>Option</th>
<th>Default/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Changesets per page</strong></td>
<td>The default is 30 per page. You can also choose 5; 10; 50; or 100.</td>
</tr>
<tr>
<td><strong>Maximum files shown in a changeset</strong></td>
<td>Default is 5. You can also choose 10; 20; or 'Show All'.</td>
</tr>
<tr>
<td><strong>Always expand changesets in streams</strong></td>
<td>Default is 'Yes'. If you choose 'No', then changesets in streams will be contracted.</td>
</tr>
<tr>
<td><strong>Show my activity on dashboard</strong></td>
<td>Default is 'Yes'. If you choose 'No', then your activity will not appear on the dashboard.</td>
</tr>
</tbody>
</table>

**Diff View**

<table>
<thead>
<tr>
<th>Option</th>
<th>Default/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diff mode</strong></td>
<td>Default is 'Unified'. Can be changed to 'Side-by-side' to change the way diffs are viewed.</td>
</tr>
<tr>
<td><strong>Line wrapping</strong></td>
<td>Default is 'None' i.e. long lines will never word-wrap. 'Soft' is when long lines will word-wrap.</td>
</tr>
<tr>
<td><strong>Highlighting Colours</strong></td>
<td>The default colour scheme uses bright colours for highlighting diffs in the code. If you prefer more muted colours, select 'Classic (muted)'.</td>
</tr>
<tr>
<td><strong>Context lines</strong></td>
<td>Default is '3'. You can also choose 'No Context'; 20; 100; or 'Full Context'.</td>
</tr>
</tbody>
</table>

**Source View**

<table>
<thead>
<tr>
<th>Option</th>
<th>Default/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default annotation mode</strong></td>
<td>Default is 'Age'. You can also select 'Author' or 'None'.</td>
</tr>
<tr>
<td><strong>Tab width</strong></td>
<td>Default is '8'. Can be changed to any value between one and ten.</td>
</tr>
</tbody>
</table>

**IDE Connector**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enable IDE icons</strong></td>
<td>Description.</td>
</tr>
<tr>
<td><strong>Port Number</strong></td>
<td>Description.</td>
</tr>
</tbody>
</table>

**Profile and Email Tab**

The settings in this tab allow you to change your email address and your display name.

<table>
<thead>
<tr>
<th>Option</th>
<th>Default/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Display Name</strong></td>
<td>Name displayed for the user currently logged in.</td>
</tr>
<tr>
<td><strong>Email Address</strong></td>
<td>The address all email notifications will be sent to.</td>
</tr>
<tr>
<td><strong>Email Format</strong></td>
<td>Default is text. Can be changed to be sent as HTML.</td>
</tr>
<tr>
<td><strong>Send Watch Emails</strong></td>
<td>Default is 'Immediately'. Can be changed to 'Daily' to receive a digest.</td>
</tr>
</tbody>
</table>
Change Password Tab

Option to be able to change your password if required.

⚠️ The passwords are case sensitive.

Author Mapping Tab

The 'Author mapping' tab allows you to make an association between you (as a logged-in user in Crucible) and an author in each repository.

This is only necessary if the name of the user within Crucible is different to the name within the repository. Crucible will by default check to see whether the usernames match.

Watches Tab

✅ Add a 'watch' on the Browse, File History or Changelog page

By adding a 'watch', you can ask to receive emails about changes made to the repository. To add a watch, click on the icon at the top right of any Browse, File History or Changelog page.

The 'Watches' tab in your Profile allows you to change the frequency at which the 'watch' emails are sent.

- 'Immediately' - the email is sent every time a change is made.
- 'Daily' - you will receive a daily email detailing these changes.

The default is 'Immediately'.

The option to add a watch may only be available if the administrator has enabled watches for the repository.

Reviews Tab

If the SMTP server is set up, then you will receive emails when different actions occur within Crucible.

You can change the options described below, to specify the stages at which emails will be sent.

Auto-mark files as 'read'

Auto-mark files as read  Default is 'Yes'. Can also be set to 'No'.

Review Notifications Events

The following options can be set to 'Immediate', 'Batch', or 'No':

<table>
<thead>
<tr>
<th>State change</th>
<th>Default is 'Immediate'. A Crucible review moves through different states e.g.: 'Draft', 'Under Review'. An email is sent when the state changes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment added</td>
<td>Default is 'Immediate'. An email is sent when a comment is added to a review.</td>
</tr>
<tr>
<td>Participant finished</td>
<td>Default is 'Immediate'. An email is sent (to the Moderator only) when any reviewer has completed their review.</td>
</tr>
<tr>
<td>General message</td>
<td>Default is 'Immediate'. An email is sent when a reviewer is added or removed from a review, after it has gone into the 'Under Review' state.</td>
</tr>
<tr>
<td>File revision added</td>
<td>Default is 'Immediate'.</td>
</tr>
</tbody>
</table>

⚠️ Batch Notifications will be sent out by Crucible every 30 minutes. All notifications will be rolled up into a single digest e-mail.

The following options can be set to 'Yes' or 'No':
<table>
<thead>
<tr>
<th>Uncomplete review if defect is raised</th>
<th>Default is 'Yes'. This allows reviews to be resurrected automatically to deal with new code or defects. Can also be set to 'No'.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncomplete review if revision is added</td>
<td>Default is 'Yes'. This allows reviews to be resurrected automatically to deal with new code or defects. Can also be set to 'No'.</td>
</tr>
<tr>
<td>My actions</td>
<td>Default is 'No'. If set to 'Yes', an email is sent every time you perform an action on a review.</td>
</tr>
</tbody>
</table>

**Customising Your User Avatar**

If your administrator has enabled an external avatar server (e.g. Gravatar), you can upload an avatar image of your choice. Note that your login name to the external server must be the email address that is specified in your user profile.

**Roles and Status Classifications**

This page explains the roles & status classifications in Crucible.

- **Roles in Crucible**
  - Author
  - Creator/Moderator
  - Reviewer
  - User
- **Status Classifications in Crucible**
  - Draft
  - Under Review
  - Summarized
  - Closed
  - Abandoned

**Roles in Crucible**

**Author**

The *author* is the person primarily responsible for acting on the outcomes of the review. In the vast majority of cases the author will be the person who made the code change under review.

**Creator/Moderator**

The *creator* is the person who creates the review. In most cases this person will also act as *moderator*. The *moderator* is the person responsible for creating the review, approving the review, determining when reviewing is finished, summarising the outcomes and closing the review. By default, the moderator is the creator.

**Reviewer**

A *reviewer* is a person assigned to review the change. Reviewers can make comments and indicate when they have completed their review. The moderator and author are implicitly considered to be participants of the review, but are not reviewers.

**User**

A *user* is a person using Crucible.

**Status Classifications in Crucible**

**Draft**

Draft Reviews are not yet completed or released to the reviewers.

**Under Review**

Reviews Under Review are either waiting for attention by reviewers or waiting to be summarized.

**Summarized**

Summarized reviews are past the reviewing phase. The moderator can still add conclusions or comments.

**Closed**
Closed reviews are complete.

**Abandoned**

Abandoned reviews are 'in the trash'. Reviews must be Abandoned before they can be deleted.

See also the *Glossary of terms* used in Crucible.

**Conducting a Review**

This page contains links to instructions how to create a review and manage the workflow through its various states to completion. Click on the desired topic to see more information.

- Creating a Review
- Selecting the Files for the Review
- Adding Reviewers
- Issuing a Review
- Performing the Review
- Summarising and Closing the Review
- Moving a Review to Another Project
- Deleting an Abandoned Review

For an overview of how to apply a workflow to Crucible, see *Defining your Workflow*.

For an explanation of the different roles that people play in a review, see *Roles and Status Classifications*.

**Creating a Review**

This page explains how to create a Crucible review.

There are a number of ways to create a review. Choose from the list below:

- Creating a Patch Review
- Creating a Review within Crucible
- Creating a Review from FishEye
- Creating a Review from JIRA
- Creating a Review from a URL
- Creating a Snippet Review

Whichever way you choose, the overall process looks like this:

| Full Size | Edit Diagram | Remove Diagram |

**Note that only people with the 'Create' permission can create a review.**

**Creating a Patch Review**

This page includes instructions on uploading patch files from your repository, how to load them into Crucible to be reviewed and use Crucible's Patch Anchoring to retrieve more lines of context from the original file.

*On this page:*

- Using Crucible Patch Anchoring to Automatically Add Full Context
- Creating a Patch File From Your IDE
  - Creating a Patch File in IntelliJ IDEA 7.0
  - If You Do Not Have the Create Patch Command Available Under IDEA
  - Creating a Patch File in Eclipse 3.3.1.1
- Creating a Basic Patch File From The Repository Command Line
  - CVS Patch Creation On The Command Line
  - Subversion Patch Creation Via The Command Line
  - Perforce Patch Creation Via The Command Line
- Creating Patches That Include All Lines of Code
  - Creating a Patch in CVS With All Lines of Code
  - Creating a Patch in Subversion With All Lines of Code
  - Creating a Patch in Perforce With All Lines of Code
Crucible allows you to review a change before it has been committed. To do this, you upload a patch file to the ‘Patch’ tab (or paste it in as text) when creating a review. You must first generate this patch file from your repository, using either commands built into your IDE, or via the repository command-line tools.

To create a patch in Perforce, you must ensure you have set P4DIFF to point to a GNU-compatible diff program.

By default, patch files will only show a few lines of code surrounding each change, rather than the entire file and its changes. Crucible's patch anchoring feature overcomes this limitation.

**Using Crucible Patch Anchoring to Automatically Add Full Context**

Crucible's Patch Anchoring feature allows you to add a regular patch to a review (showing only a few lines of context). Then, Crucible will automatically search for the relevant file content in the connected repositories. When it finds the files, it will seamlessly add in more context from the file so that you can view all of the lines of code (greatly enhancing the review process).

**To use patch anchoring:**

1. Create a new review. From the ‘Tools’ menu in Crucible, select ‘Create Review’.
2. Click ‘Pre-Commit - Upload a patch file to be reviewed’. The ‘patch upload’ dialog appears. Click ‘Browse’, locate your file, then click ‘Upload’. Crucible will now search for matches in the files in its database. Crucible will analyse all the paths in the patch, find the branches containing all those paths, then anchor the patch to the trunk or the branch with the most recent commit activity.

Crucible makes a 'best guess' in its processing – you should check that it has anchored the patch to the correct location in your repository.

**Screenshot: Crucible Patch Anchoring**

1. You can click ‘Edit’ to change the anchoring, by choosing a new match or removing the anchor. You can change the anchoring later, after the review is live.
2. Start the review. When viewing the diffs, you will be able to choose more than three lines of context from the ‘View’ menu.

**Screenshot: Editing Patch Anchoring Settings**
Creating a Patch File From Your IDE
Creating a Patch File in IntelliJ IDEA 7.0

To create a Patch File under IntelliJ IDEA, do the following:

Select a parent folder, sub-folder or file that you have altered in the Project tool window. Select 'Version Control' > 'Create Patch'. The following window appears:

Screenshot: The IDEA Create Patch window

Click 'Create Patch'. Choose a location to save the patch file and click 'Ok'.

If You Do Not Have the Create Patch Command Available Under IDEA

If you have not configured version control in IDEA, you may not have the 'Create Patch' option available. If so, use the following steps to create a patch file in IDEA:

1. Select a parent folder, sub-folder or file that you have altered in the Project tool window, right-click it and choose 'Local History' > 'Show History'.

Screenshot: The IDEA Show History dialog
2. In the Local History view, right-click the revision number, and choose 'Create Patch'.

*Screenshot: The IDEA Create Patch dialog*

3. In the Create Patch dialog, choose a location for the patch file and a file name, then click 'OK'.

*Creating a Patch File in Eclipse 3.3.1.1*

To create a patch file under Eclipse, do the following:

Find the parent folder, sub-folder or file that you have altered, right-click it and choose 'Team' > 'Create Patch'.

*Screenshot: Instigating a Patch in Eclipse*
In the Create Patch window, choose a location on your computer and type an appropriate file name (the file format is plain text).

Screenshot: The Eclipse Create Patch dialog
Creating a Basic Patch File From The Repository Command Line

**CVS Patch Creation On The Command Line**

To create a patch in CVS, use the `cvs diff -Nu` command from your workspace. For example:

```
cvs diff -Nu > patch.txt
```

Note that patch files created with this command will only include around three lines of code, before and after each change.

**Subversion Patch Creation Via The Command Line**

To create a patch in Subversion, use the `svn diff` command from your workspace. For example:

```
svn diff > patch.txt
```

⚠️ `svn diff` does not print any information about files copied in the workspace.

Note that patch files created with this command will only include around three lines of code, before and after each change.

**Perforce Patch Creation Via The Command Line**

To create a patch in Perforce, you must ensure you have set `P4DIFF` to point to a GNU-compatible diff program.

Then use `p4 diff -du` to generate a patch for changed files. For example:
Since Perforce diffs do not include added and deleted files so you should then do a p4 opened to find such files. For added files, append them individually to the patch using GNU diff:

```
diff -u path_to_added_file /dev/null >> patch.txt
```

In the example above, replace `path_to_added_file` with the actual path of your added file. You can follow a similar procedure with deleted files using p4 print to extract the previous version of the file.

Note that patch files created with this command will only include around three lines of code, before and after each change.

Creating Patches That Include All Lines of Code

To create a patch file that shows all lines of code as well as the changes, use the following commands from your repository command-line tools.

Creating a Patch in CVS With All Lines of Code

To create a CVS patch that shows all code (not just the changes and surrounding code), use this command:

```
cvs diff -N -U 10000 > patch.txt
```

The '10000' number refers to the number of code lines included in the patch, before and after each change. 'Patch.txt' represents your desired name for the new patch file.

Creating a Patch in Subversion With All Lines of Code

To create a patch in Subversion that shows all code (not just the changes and surrounding code), use this command:

```
svn diff --diff-cmd diff -x "-U 10000" > patch.txt
```

- The in-built diff feature in svn diff does not support specifying lines of context, so you must tell Subversion to use an external diff command.
- The second "diff" in the command above needs to be the name of your external diff command. You might need to specify the full path to that command, such as `/usr/bin/diff`.
- On the Windows platform, you may need a Unix-like emulator such as Cygwin, and install the optional diff command for that.

Creating a Patch in Perforce With All Lines of Code

Unfortunately, Perforce does not directly support creating patches that include all lines of code. A workaround is to checkout 'before' and 'after' versions of the file, and use GNU Diff to create a patch between the two files. That file could then be loaded into a Crucible review.

Creating a Review within Crucible

This page explains how to create a review from the Crucible interface.

On this page:

- Create a new review
- Choose where your review files will come from
- Add content to the review
- Edit the review details
- Adding an entire directory's contents to a Crucible review

Create a new review

Within Crucible, create a review by opening the Tools menu at the top right of the Reviews screen, then clicking Create Review. You will be prompted to select the Project for the review (if you have multiple projects). Choose a project and click 'Create Review'.

Screenshot: The create review dialog
Choose where your review files will come from

The 'Add Content to Review' screen appears, where you will now be prompted to choose where your review files will come from. Choose one of the options by clicking.

*Screenshot: The content menu for Crucible reviews*

Add content to the review

Once you select where your review files are coming from, you are prompted to select the files to be reviewed. Check the boxes next to any files you want to add.

*Screenshot: Adding content to your review*
Edit the review details

Once you have selected the files, click **Done** to go to the **Edit Review Details** screen, as shown below.

**Screenshot: Editing review details**

On the **Edit Review Details** screen, you can choose a title, reviewers, objectives, due date, linked reviews and issues. Once you're finished, click 'Done'.
The review will open in a preview form. Here, you can check all the details and click to edit any that aren't correct. Once you click **Start Review**, the review is live.

Adding an entire directory's contents to a Crucible review

To add an entire directory's contents to a Crucible review, you will need to search to find all the files. For example, using "select revisions from dir /some/dir where is head", or similar logic.

⚠️ It is currently not possible in Crucible to add all the contents of a directory to a review with one click.

### Creating a Review from FishEye

This page explains how to create a Crucible review from FishEye.

**On this page:**

- 1. Opening the FishEye Source View
- 2. Starting the Review
- 3. Choosing a Project
- 4. Selecting Files for Review

#### 1. Opening the FishEye Source View

To begin, the code author sets up the review. There are a **number of ways to do this**, but for this example, the author starts from the FishEye Source view of the file he wants to review:

*Screenshot: Opening a review from the FishEye Source view*
2. Starting the Review

From the FishEye Source view, the author clicks the 'Reviews' drop-down menu above the source view, then selects 'Create New Review'. If there are multiple projects, the Project Selection dialog opens.

Screenshot: The Project Selection dialogue

3. Choosing a Project

In the Project Selection dialogue, you are prompted to choose a project for this review from the drop-down list. Once the selection is made, the author clicks the 'Create New Review' button. The Manage Files dialog opens.

Screenshot: The Crucible Manage Files dialogue
4. Selecting Files for Review

In the Manage Files dialogue, the author selects the source files they want to include in the review, by clicking the checkboxes next to the desired files. Once finished, the author clicks 'Done'. The Edit Review dialog appears, where the author can create and issue the review.

The next step is to add reviewers.

Creating a Review from JIRA

This page explains how to create a Crucible review directly from JIRA, the Atlassian issue-tracker application.

To create a review from within JIRA, click the small Crucible icon next to the required changeset on the 'FishEye' tab.

Screenshot: Adding a Review from within JIRA

When you click the icon, you will be prompted to select the relevant project (if more than one project exists) in which to create your review. A new draft review will then be created, including the following information pre-filled:

- The content of the changeset becomes the content (i.e. files) to be reviewed.
- The author of the changeset becomes the author of the review, if Crucible is aware of this user. Otherwise the creator of the review.
becomes the author.  
- The creator of the review becomes the moderator.  
- The commit log message is used as both the Title and Statement of Objective.

All aspects of the review can be changed. To edit any of the above settings, click the title to see the 'Edit details' screen. Or you can click the Manage Files tab.

If you click the Crucible icon, you will see the 'Review' screen below:

Screenshot: Review Screen in Crucible

The next step is to add reviewers.

Creating a Review from a URL

You can set up a URL which you can then click to create a Crucible review.

The format of your URL is as follows:

```
```

The parameters are as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>csid</td>
<td>The changeset ID. You can specify one or more, of the form //repo/csid (where '%2F' is the URL-encoded form of is '/')</td>
<td>Yes</td>
</tr>
<tr>
<td>repo</td>
<td>The name of your repository.</td>
<td>Yes (unless supplied in the csid)</td>
</tr>
<tr>
<td>title</td>
<td>The title of your new Crucible review.</td>
<td>No</td>
</tr>
<tr>
<td>description</td>
<td>The description of your new Crucible review.</td>
<td>No</td>
</tr>
</tbody>
</table>

When you click the URL, you will be prompted to select the relevant project (if more than one project exists) in which to create your review. A new draft review will then be created, including the following information:

- The content of the changeset becomes the content (i.e. files) to be reviewed.
- The author of the changeset becomes the author of the review, if Crucible is aware of this user. Otherwise the creator of the review becomes the author.
- The creator of the review becomes the moderator.
- The commit log message is used as both the Title (unless you have explicitly defined a title in your URL) and Statement of Objective.

All aspects of the review can be changed. To edit any of the above settings, click the title to see the 'Edit details' screen. Or you can click the Manage Files tab.

The next step is to add reviewers.

Creating a Snippet Review
This page explains how to create a simple code review using the Crucible ‘Snippet Review’ feature. Snippet Reviews are designed to be lightweight ad-hoc code reviews.

**To create a snippet review**

1. Copy the desired code from the source to your system clipboard.
2. Click the **Create Snippet** button in Crucible.
3. Enter the details for the snippet review as follows:
   - Paste the code from the system clipboard into the page. Snippet reviews can only contain one discrete file, or code snippet.
   - Click the **Click to add title** text. The field will become editable. Enter a title for your review. If you don’t specify a title, one will be automatically created for you.
   - Select a project from the **Project** dropdown.
   - Select a programming language to be used for syntax highlighting from the **Syntax Highlighting** drop-down menus.
4. Click the **Create** button to create the snippet review. The review will be created.
5. Invite anyone that you want to participate in the snippet review by sending them a link to the review. Everyone can see snippet reviews, anyone can comment on the review, and anyone can close it.
6. Click the **Reply** link on any comments to respond to the comments.
7. Click the **Tools** menu and click **Close** to close the snippet review. The snippet review will be closed, i.e. no-one will be able to add new comments or reply to existing comments. Anyone can re-open, re-review or close snippet reviews. However, only the creator of a snippet review can delete it.

**Screenshots: Creating a Snippet Review (click to view larger images)**

Step 1
Step 2
Step 3

**Selecting the Files for the Review**

This page explains how to select files/changesets that will be included in a Crucible review.

On this page:

- Selecting Changesets for Review
- Selecting Files for Review by Exploring Repositories
- Selecting Files for Review using the Crucible Search
- Using the Suggestions Feature When Adding Files to a Review
- Adding Pre-commit Patch Files to a Review
- Adding Attachments to a Review

**To add content to a review,**
1. Log into FishEye/Crucible and either:
   - Create a review, as described on Creating a Review, or
   - Open an existing review, which you are the creator or moderator of, and click the Add Content button.

2. The 'Add Content to Review' dialogue will be displayed (see screenshot below). Select the method you want to use, to find the content for your review:
   - 'Browse Changesets' — Allows you to choose changesets from a Source Code Management (SCM) repository to add to a review. See Selecting Changesets for Review.
   - 'Explore Repositories' — Allows you to browse for files from a Source Code Management (SCM) repository to add to a review. This option only appears when FishEye is installed. See Selecting Files for Review using by Exploring Repositories.
   - 'Search for Files' — Allows you to search a Source Code Management (SCM) repository for files or changesets to add to a review. This option only appears when FishEye is installed. See Selecting Files for Review using the Crucible Search.
   - 'Suggest Files' — Analyses the list of files in the current review and makes suggestions based on certain logic (for example, suggesting a newer version of a file if one exists). This option only appears when FishEye is installed. See Using the Suggestions Feature When Adding Files to a Review.
   - 'Pre-commit' — Allows you to upload patch files to a review. See Adding Pre-commit Patch Files to a Review.
   - 'Attachments' — Allows you to upload any file to a review, including binary files and files outside of a Source Code Management (SCM) repository. See Adding Attachments to a Review.

3. Follow the appropriate instructions in the sections below to add content to your review.

4. Click the 'Done' button to finish adding content to the review.

---

**Screenshot: The Add Content menu for Crucible reviews**

**Selecting Changesets for Review**

Click the 'Browse Changesets' option on the 'Add Content' dialogue to add changesets to your review.

**Screenshot: The Browse Changesets View in the Add Content dialogue**
By default, Crucible presents a list of the author's changesets in reverse chronological order. You can see other changesets by changing the options at the top of the screen.

Click the checkbox next to a changeset ID to add the entire changeset. Note, you cannot add individual file revisions to a review, although you can remove them once the changeset is added. Click 'Remove all revisions from review' to remove all.

Options for adding changesets:

- 'Repository' — This is a list of the repositories that contain the files that can be reviewed. If the repository you require is not in the list then it has not been added to FishEye. Please contact your Crucible/FishEye administrator.
- 'Author' — This contains a list of all the authors who have made changes within the repository. When creating a review, this will default if possible to the username of the user authoring this review and will therefore show their changesets.
- 'Branch' — This will only show files and recent changes on that branch from the repository set above.
- 'Tag' — This will only show files and recent changes tagged.
- 'Add to Review As' — Choose the form of the review. See Choosing the way files are added to the review below.
- 'Go to Changeset' — Allows you to jump to a particular change set by entering its title and pressing Enter.

Choosing the way files are added to the review:

When adding files to a review, you can set the form of review taking place in the 'Add to Review as' drop-down menu:

- 'Whole Files' — Adds the entire file with all content, rather than just a diff with context.
- 'Diffs' — This is the default behaviour. This allows you to add multiple revisions of a file to one review and compare them in-review, in context with the change history.
- 'Diffs to Last Reviewed Version' — This adds files with a diff to the last reviewed changeset.
- 'Diffs to... (a particular revision)' — This allows you to specify the file to show the differences between two specific versions of a file.
- 'Diffs to Last Branch Point' — This adds files with a diff to the revision each file was last branched.

Click the 'Done' button once you have finished selecting the desired files. The files will be added to your review and your review will be displayed.

Selecting Files for Review by Exploring Repositories

Click the 'Explore Repositories' option on the 'Add Content' dialogue to browse repositories for files to add to your review.
To find a file, browse the folders by clicking the relevant folder. The folders by default are sorted by path name but can sorted by last-commit or first-commit.

To choose a file for reviewing, click the checkbox to the left of the filename and if required the revision number.

To select a particular revision of a file, open the revision number list and select the option "Load Full History...". This will refresh the available options in the list.

Please note the following information when browsing files to add to a review:

- Empty folders will be greyed out.
- If the folders contain empty folders then a toggle option called 'Hide Empty' will appear under the 'Sort' options.
- To see or ignore deleted files, you can click the 'Hide' and 'Show' options located above the file names on the left.

Choosing the way files are added to the review:

When adding files to a review, you can set the form of review taking place in the 'Add to Review as' drop-down menu:

- 'Whole Files' — Adds the entire file with all content, rather than just a diff with context.
- 'Diffs' — This is the default behaviour. This allows you to add multiple revisions of a file to one review and compare them in-review, in context with the change history.
- 'Diffs to Last Reviewed Version' — This adds files with a diff to the last reviewed changeset.
- 'Diffs to... (a particular revision)' — This allows you to specify the file to show the differences between two specific versions of a file.
- 'Diffs to Last Branch Point' — This adds files with a diff to the revision each file was last branched.

Click the 'Done' button once you have finished selecting the desired files. The files will be added to your review and your review will be displayed.

Selecting Files for Review using the Crucible Search

Click the 'Search for Files' option on the 'Add Content' dialogue to use the Crucible search to find files to add to your review.

The 'Search' view is only available when using FishEye with Crucible.

Screenshot: Searching for files to add to a review
If you are not certain about which changesets/revisions/files to include in a review, use the Search view to find them. Adjust the search filters to find the files you need. If the simple filters are not enough, read about EyeQL queries in the FishEye documentation.

Read the FishEye documentation for more information about the searching your repository.

Choosing the way files are added to the review:

When adding files to a review, you can set the form of review taking place in the 'Add to Review as' drop-down menu:

- 'Whole Files' — Adds the entire file with all content, rather than just a diff with context.
- 'Diffs' — This is the default behaviour. This allows you to add multiple revisions of a file to one review and compare them in-review, in context with the change history.
- 'Diffs to Last Reviewed Version' — This adds files with a diff to the last reviewed changeset.
- 'Diffs to... (a particular revision)' — This allows you to specify the file to show the differences between two specific versions of a file.
- 'Diffs to Last Branch Point' — This adds files with a diff to the revision each file was last branched.

Click the 'Done' button once you have finished selecting the desired files. The files will be added to your review and your review will be displayed.

Using the Suggestions Feature When Adding Files to a Review

Click the 'Suggest Files' option on the 'Add Content' dialogue to view and add files suggested by Crucible to your review. You need to have already added some file(s) to review for Crucible to suggest additional files.

Images: Viewing and adding file suggestions to a review
Crucible can make intelligent suggestions when you are creating a review. The Suggestions feature logic is based around the following:

- **Most recent versions:** If a newer version of a file exists, Crucible will suggest that you add it to the review.
- **Similar files:** Files with a similar filename may be of relevance to your review; Crucible will show them to you.

### Choosing the way files are added to the review:

When adding files to a review, you can set the form of review taking place in the **Add to Review as** drop-down menu:

- **Whole Files** — Adds the entire file with all content, rather than just a diff with context.
- **Diffs** — This is the default behaviour. This allows you to add multiple revisions of a file to one review and compare them in-review, in context with the change history.
- **Diffs to Last Reviewed Version** — This adds files with a diff to the last reviewed changeset.
- **Diffs to... (a particular revision)** — This allows you to specify the file to show the differences between two specific versions of a file.
- **Diffs to Last Branch Point** — This adds files with a diff to the revision each file was last branched.

Click the **Done** button once you have finished selecting the desired files. The files will be added to your review and your review will be displayed.

### Adding Pre-commit Patch Files to a Review

Click the **Pre-commit** option on the **Add Content** dialogue to add pre-commit patch files to your review.

*Screenshot: Adding pre-commit patch files to a review*
For a full explanation of the ‘Patch View’ functions, read about creating a patch review.

Click the ‘Done’ button once you have finished selecting the desired files. The files will be added to your review and your review will be displayed.

**Adding Attachments to a Review**

Click the ‘Attachments’ option on the ‘Add Content’ dialogue to add attachments to your review.

*Screenshot: Adding attachments to a review*

You can upload additional files to be used in the review, including binary files, images or code files that are not stored in a version control repository. The ‘Upload’ view contains various controls to help you do this. These are listed below.

Choose the ‘Upload Method’ as either ‘Select file from the file system’ or ‘Paste text from clipboard’:

- Displays if ‘Upload method’ is ‘Select file from the file system’:
  - ‘Character Set’ (if any) — Click the edit icon (📝) to choose the character set being used. ‘US-ASCII’ is the default.
  - ‘File’ — Click ‘Browse’ to find the file that you want to add to the review.
- Displays if ‘Upload method’ is ‘Paste text from clipboard’:
Click the 'Upload' button, when you have made your selection. Once complete, a list of uploaded files is displayed at the bottom of the screen. To add another iteration of a file, make changes to the file and upload it again with the same filename. It will be added as a new version.

Click the 'Done' button once you have finished selecting the desired files. The files will be added to your review and your review will be displayed.

**Iterative Reviews**

Crucible allows you to review several revisions of a file within one review, seamlessly switching between them. Comments are linked and relative to a specific revision. This allows you to review every change that has occurred on a code file within a given period of time. This lets you see the evolution of the file through various revisions (within one Crucible review).

**Screenshot: Iterative Reviews**

**Adding Reviewers**

This page explains how to add reviewers to a new review, after it has been created. See [Creating a Review](#) for information about creating reviews.

On this page:

- Entering Basic Information
- Adding Reviewers
  - Adding Users to a Review
  - Crucible Suggests Reviewers
  - Inviting Non-Registered Users to the Review
  - Checking the Draft and Starting the Review
- Next Steps

**Entering Basic Information**

Once a review has been created, the Edit Review dialog opens.

**Screenshot: The Edit Review dialog**
In the Edit Review dialog, the author enters information needed for the review. This includes entering a title and description for the review, a due date and the key for a related JIRA issue (if any). The project, moderator and author are pre-selected (for this example, the author should select himself as a moderator.

You must also select reviewers.

**Adding Reviewers**

Before a review can be issued to reviewers, you must decide who can review it. When adding reviewers, you can add registered users immediately. The usernames will auto-complete, showing partial matches before you finish typing. You can quickly select one of the matches shown with the keyboard arrow keys, pressing Enter or Tab to add them to the review.

In addition, you can easily invite external users who do not yet have accounts in Crucible to take part by typing their email address into the Reviewers field.

**Adding Users to a Review**

Select users by typing names into the text field under Reviewers. Crucible will show a list of matches. Press Enter to select one after each entry. Clicking the 'Save' button will save the review as a draft for later issue.

You can also decide to allow any registered user to add themselves as a reviewer in the review. To enable this option, put a check next to 'Allow anyone to join'.

**Crucible Suggests Reviewers**

Crucible will automatically suggest reviewers, by analysing the users that have contributed to the files you've selected and also don't have a lot of open reviews. You can easily pick reviewers from the list of suggestions by clicking.

**Inviting Non-Registered Users to the Review**

You can invite users who don't have a Crucible account to join a review.

⚠️ There are two prerequisites:

1. FishEye's SMTP server must be configured and capable of sending email.
2. The setting 'Built-in Public Sign-up' must be set to 'ON'. This setting can be accessed by opening the 'Admin Menu', then clicking 'Security' under 'Global Settings' on the left navigation bar.

To invite an external user to a review,
You can enter multiple addresses separated by commas, allowing you to paste in a list of email addresses from your favourite email application.

When finished, the author clicks ‘Save’. The review will now be created in a draft form.

**Checking the Draft and Starting the Review**

The draft review opens. In the draft stage, the author can check the contents of the review files to ensure they are correct and put in any notes for reviewers as comments. During the draft phase, no notification email is sent out to reviewers. Once the author is finished with the draft phase, he clicks ‘Start Review’.

The review will now be started and notification email will go out to all participants. Crucible will now send out an email notification to all the participants. This lets them know that the review is under way and prompts them to take action, providing a URL for direct access to the review. (You can also subscribe to an RSS feed.)

Next Steps

You can now begin **Performing the Review**.

If you have a moderator controlling your review process, you can move onto **Issuing a Review**.

**Issuing a Review**
This page contains information about starting a review in Crucible.

On this page:
- Starting a review
- Editing review details once started

**Starting a review**

Issuing a review simply means formally starting it and inviting people to take part.

Once you have selected the reviewers, the next stage is to notify the reviewers and the author (if different to the moderator) that they can start reviewing. The review has been in 'Draft' state until this point. Only the moderator has the permission to start a review.

To start the review:
- If you are the moderator of the review, click the 'Start Review' button. Or;
- If you are not the moderator of your review, click 'Send to Moderator'. This changes the state to 'Require Approval' and notifies the moderator. The moderator can change any aspect of the review before starting it.

Once the review has been started, the review state becomes 'Under Review'.

**Screenshot: Starting a Review**

Note that only people with the 'Approve' permission can start a review.

**Editing review details once started**

You can edit the details of a review at any time by simply clicking the 'Edit Review' button in the left navigation bar to launch 'Edit Mode'. In Edit mode, you can quickly click red cross icons to remove files from the review. A single click returns you to regular Crucible functions, so you can more easily tune the content inside your reviews. Another button opens a dialog for rapidly adding more content to the review.

**Screenshot: Launching Edit Mode**

**Screenshot: Crucible Edit Mode for Review Content**
Performing the Review

This page describes how to find and manage the Crucible reviews that relate to you.

On this page:

- Browse Your Reviews Under the 'Dashboard' Tab
- Browse All Reviews Under the 'Reviews' Tab
- When Files Change During a Review
- Next Steps

Deciding what needs to be reviewed

The 'Statement of Objective' is a brief description of what the review is intended to achieve. Crucible does not dictate how or what to review. It simply provides a mechanism to record comments.

Browse Your Reviews Under the 'Dashboard' Tab

When you first load Crucible, the 'Dashboard' screen will load, which shows your current reviews and other activity related to you.

Use the Crucible 'Dashboard' to manage your reviews. Read the overview on filtering your view.

Active reviews are listed on each reviewer's dashboard under the default 'To Review' filter. Reviews are listed under 'Out for Review' until all reviewers indicate they are complete. Then the reviews move to the 'To Summarize' list.

Read more about using the Dashboard tab.

Browse All Reviews Under the 'Reviews' Tab

All reviews that involve you in any role are listed when you click 'Open' or 'Closed' in the left navigation bar. For instance, use the 'My Reviews' --> 'Open' filter to locate a review that doesn't require further action from you, but is still under way.

If email notifications are enabled (see SMTP settings in the FishEye documentation), reviewers will receive an email with information about the review. Click the link within the email to go directly to the review.

When Files Change During a Review

If a file in the repository changes during a review, Crucible will visually alert you by showing the 'File Outdated' menu:
From the 'File Outdated' menu, you can choose to view the latest revision of the updated file, or add the latest revision to the review:

Next Steps

- Adding Comments
- Flagging Defects
- Completing your Review
- Sending all of a Review's Comments via Email
- Using the Review History Dialog
- Tracking Crucible Review Metrics
- Using JIRA Integration in Crucible Reviews

Adding Comments

Comments can be added at the level of a review, revision, or line. You can also reply to a comment.

On this page:

- Locating existing comments
- Adding a Comment
- Draft Comments

Locating existing comments

The number shown next to a filename, in the left-hand column of the screen, indicates the number of comments that apply to that file. (The number of unread comments, if there are any, is shown in brackets.)
Adding a Comment

- To add a comment that applies to the whole review, select the 'Review Comments' line in the left-hand navigation panel, then click the following icon:

- To add a comment that applies to a revision/change, select the filename in the left-hand navigation panel, then click the following icon:

- To add a source-level comment, expand the source view then click a line of code. You can click and drag to select multiple lines from one revision or diff, or click individual lines to select/deselect them. The comment will appear in the source at the last line selected. Hover over the comment to highlight the selected lines.

- To select text on the page without adding a comment, hold down the Alt button while dragging the cursor.

- To reply to a comment, click the 'Reply' link at the bottom of the comment.

Only people with the 'Comment' permission can add comments.

Read about flagging defects too.

Screenshot: Adding a Comment
Draft Comments

You can save your comment as a draft and then edit it later. When you complete the review, you will be prompted to post, discard or edit any remaining draft comments.

Screenshot: Draft comments

Flagging Defects

Comments in Crucible can be used to flag a defect in the code under review.

to do this, simply tick the 'Defect' box when adding a comment and select a category from the drop-down list.

Screenshot: Defects

You may want to mark comments as defects to associate defect classifications, or simply to highlight to the author or moderator that the issue you raised in your comment requires attention.

1. Crucible intentionally does not mandate how defects are to be used. The Crucible administrator can customise the defect classifications.

2. You can only use the defect classifications on comments that are not a reply to an existing comment.

Completing your Review

Once each reviewer has added comments to the review and has nothing further to add, the next step is to Complete their individual review.
To complete your individual review, go to the review and click the 'Complete' button at the right of the screen, next to the 'Tools' menu:

**Screenshot: The Complete Button**

![Complete Button](image)

Only people with the 'Complete' permission can complete a review.

This notifies the moderator (via email if configured) that you have completed your review.

Reviewers can still continue to add comments until the moderator summarises the review. The moderator does not have to wait for all reviewers to complete their individual reviews before summarising.

If you have any draft comments, you will be prompted to post/discard/edit any comments before completing the review.

**Screenshot: Draft comments**

![Draft Comments Warning](image)

You have draft comments

- Draft comments that aren't posted will be deleted.

- View drafts
- Delete drafts
- Post drafts

**Screenshot: Review complete**

![Review Complete](image)
Sending all of a Review's Comments via Email

You can send all of the comments from a review to anyone you want via email. You may wish to do this to allow a person outside the review to quickly scan the content of the comments, or oversee the review activity. Alternatively, you may wish to send all participants this information to let them read the current status of the review and its comments in full.

**Sending a Review's Comments via Email**

To send all of a review's comments via email,

1. In Crucible, navigate to the review in question.
2. From the 'Tools' menu, select 'Email Review' (see Screenshot 1 below).
3. The 'Recipient' page appears (see Screenshot 2 below). On that page:
   - In the 'To' field you can enter multiple email addresses, separated by commas.
   - In the 'Recipient' field, you can type usernames from your Crucible instance to add them to the distribution list. You can also simply tick the 'Send to Review Participants' check-box to include all of the review's reviewers.
4. When you have finished the distribution list, click the 'Next' button.
5. The 'Recipient' page appears (see Screenshot 3 below). On that page:
   - In the 'To' field you can enter multiple email addresses, separated by commas.
   - When you have finished your message, click the 'Send' button.
6. The 'Status' page appears (see Screenshot 4 below), confirming that your email has been sent.

Users that are not logged in cannot send email, but instead can view the text content of the review's comments by clicking the 'View Text' option which will appear instead of 'Email Review'.
Screenshot 2: The 'Recipients' Screen in Crucible

Email Review Comments

1 Specify recipients  2 Message details  3 Status

Recipients
Specify the people who should receive the comments of this review. You can specify a list of comma separated email addresses and/or specify one or more Crucible users.

To: [Field]
Recipients: [Field]

Start typing a user name then press enter to select
[Option]

Screenshot 3: The 'Message' Screen in Crucible
Using the Review History Dialog

The Review History dialog shows a chronological list of interactions within a review. You can see rich information about those interactions and control their display. You can sort the information by date, actor, or action.

To open the Review History dialog,
1. Open a review in Crucible.
2. Click ‘Tools’ > ‘Review History’ at the toolbar at the top right corner of the screen.
3. The Review History dialog opens.

This information can also be displayed in the new timeline mode, a graphical visualisation that shows events on a horizontal graph over time (click the ‘Timeline’ tab at the top of the dialog to switch from ‘Details’ to timeline view). Click and drag inside the timeline view to scroll the graph left and right. You can also click on the section showing months to scroll over a greater time scale.

Additionally, you can get access to the entire review history through the ‘CSV export’ link in the upper right hand corner, allowing for easy data import into a spreadsheet or other application.

Screenshot: The Crucible Review History Dialog

![Screenshot: The Crucible Review History Dialog, Timeline View]
Tracking Crucible Review Metrics

Crucible tracks each participant's percentage completion through each review and the total time they have spent.

To learn about these features, see the following pages:

- Using Progress Tracking
- Using Time Tracking

Using Progress Tracking

This page contains instruction on how to use progress tracking in Crucible.

On this page:

- How progress tracking works in Crucible
- Viewing the progress tracking totals
- How to adjust progress tracking on a review
- Adjusting settings for progress tracking
- Further reading

How progress tracking works in Crucible

As you work your way through the files in a review, Crucible tracks the ones you have viewed. Whenever you open a file for review, Crucible will automatically mark it as read.

When participating in iterative reviews, progress tracking also takes lines of code and revisions into account.

Viewing the progress tracking totals

The 'Details' view shows a summary of the progress of each participant through the files in the review.

If there is only one file in the review, then the progress tracked will either show 0% or 100%.

How to adjust progress tracking on a review

You can mark a file as unread by clicking on its name to view the file's contents. In the source view, you have an option at the top left of the screen, 'Leave Unread'. If you select that, then the file you are looking at will not be added to your progress percentage.
Adjusting settings for progress tracking

Progress tracking is a configurable user preference that can be changed in the ‘User Settings’ menu, in the ‘Reviews’ sub-section. ‘Auto-mark files as read’ is on by default. When ‘Auto-mark files as read’ is set to ‘No’, marking files as read or unread is left to the user to manually manage.

Further reading

You may also want to learn about Crucible’s Time Tracking feature.

Using Time Tracking

This page contains instruction on how to use time tracking in Crucible.

On this page:

- How time tracking works in Crucible
- How to adjust the time tracked on a review
- Viewing the time tracking totals
- JIRA integration
- Further reading

How time tracking works in Crucible
Crucible 2.4 Documentation

Crucible will automatically track the time you spend in a Crucible review. When you open a file for review, a counter in the Review Details panel starts. The time is added to your total when you leave the review screen.

**Screenshot: Crucible Time Tracking**

![Crucible Time Tracking](image)

**How to adjust the time tracked on a review**

You can click and type in the time tracking control to adjust the time you have spent during the session.

**Viewing the time tracking totals**

The 'Details' view shows a summary of the progress and time tracked on each file.

**Screenshot: Crucible Tracking Totals**

![Crucible Tracking Totals](image)

**JIRA integration**

Using Crucible when integrated with JIRA, you can update time tracking from the following locations:

- The confirmation dialog for a reviewer completing a review,
- The confirmation dialog on closing a review,
- The regular toolbar location in Crucible.

**Screenshot: JIRA Time Tracking Integration**

![JIRA Time Tracking Integration](image)

**Further reading**

You may also want to learn about Crucible's Progress Tracking feature.

**Using JIRA Integration in Crucible Reviews**
This page contains information on how to use JIRA integration with Crucible reviews.

On this page:

- 1. Create a JIRA Issue for your Review
- 2. Create Your Review and Link it to a JIRA Issue
- 3. Make a Comment or Defect on the Review
- 4. Click to create a JIRA Sub-Task
- 5. Resolve the JIRA issue through Crucible

Before you begin, both your Crucible and JIRA instances must be configured to use make use of these JIRA integration features. Read more.

1. Create a JIRA Issue for your Review

To use JIRA integration, you must begin with a JIRA issue that you will use as the parent issue for the review. Crucible will create and resolve sub-tasks belonging to this parent issue. Once your parent issue is created, make a note of its issue key, e.g. FE-1968.

2. Create Your Review and Link it to a JIRA Issue

When creating your review, you have an option called 'Linked Issue'. Crucible may put a suggested JIRA issue key into this field automatically. You can specify a different issue key and click 'Link' to save it. You can also click the 'x' to clear the field and load a different issue key.

Screenshot: Selecting a JIRA Issue When Creating a Review

You can also link a JIRA issue to the review after the review is created. When viewing a review, the top-left hand corner of the screen shows meta-information. One of the links in this area is titled 'Linked JIRA Issue:' and then a suggested JIRA issue key. Click this link to associate that JIRA issue with this review.

3. Make a Comment or Defect on the Review

Once your review has a linked JIRA issue, create a comment or defect comment anywhere on the review. Once created, the comment actions will show a link titled 'Create Issue' (note that this link does not appear on replies — only on new comments). You can click that to instantly create a sub-task under the parent JIRA, which will take the content of the comment as its summary.

Screenshot: Selecting a JIRA Issue From a Comment
4. Click to create a JIRA Sub-Task

Clicking the 'Create Issue' link will allow you to create a JIRA sub-task under the parent JIRA issue, e.g.:

*Screenshot: The JIRA 'Create Issue' dialog*

- ‘Automatic’ (i.e. the default assignee for that JIRA project)
- the assignee of the subtask's parent issue
- the reporter of the subtask's parent issue
- ‘Unassigned’ (if your JIRA administrator has enabled Allow Unassigned)
- plus, if Trusted Applications have been configured between JIRA and Crucible,
  - the review author
  - the review moderator
  - the comment/defect author
  - yourself

Once created, the sub-task JIRA issue key, status and default action (i.e. ‘Resolve’) will be shown. If you hover your mouse over the JIRA issue key, an information window will show more information and controls relating to that JIRA issue.
Users are mapped to their own accounts when using Trusted Applications. If you don't have the permissions to carry out the default action ('Resolve', in this case), an error will occur.

5. Resolve the JIRA issue through Crucible

Once the work required on your sub-task is completed, simply click the action link provided to signal that this has occurred (e.g. 'Resolve'). The JIRA issue will be closed.

If you encounter problems or have trouble using JIRA integration, please read the FAQ page on this topic.

Summarising and Closing the Review

As the moderator, you can choose to summarize a review at any time.

Normally, we recommend that you wait for all reviewers to complete their reviews. The reviews that the reviewers have completed will be in your 'To Summarize' menu on the Dashboard.

To summarize a review,

- Click the 'Summarize' button at the right of the screen.
- Optionally enter a summary of the review.
- If you have no further comments to add, click the 'Close Review' button; otherwise, click 'Continue Without Closing'.
The above review is not yet complete
We can see that Geoff Crain has still not finished reviewing, because there is no green tick next to his name.

On clicking ‘Summarize’, the moderator may be prompted to confirm the action if there are incomplete reviews or draft comments in the review.

The requests for confirmation are warnings only
The review can still be summarized and closed.

Once the review is in the ‘Summarize’ state, the moderator can optionally add a review summary, i.e. describe the outcomes/tasks/etc.

Summarize the review outcomes (optional)
Testing successful. Thanks for your time, everyone!
The summary is sent to all participants and displayed at the top of the closed review.

- The moderator is the only participant who can add comments in 'Summarize' state. This gives the moderator the responsibility of the 'last word'.
- Reviews in the 'Summarize' state can be closed.
- Reviews in the 'Summarize' or 'Closed' state can be re-opened. Re-opening changes the review's state back to 'Under Review', allowing all participants to add comments.

Re-opening a review is not the recommended way to 're-review'. You should create a new review with the reworked changes and link it to its parent review (create a hyperlink back to the original review in the new Review's Objectives field).

Note that you need the 'Summarize', 'Close' or 'Re-Open' permission to summarize, close or re-open a review.

Moving a Review to Another Project

You can move reviews between projects once they have been created.

To move a review between projects,

1. Open the review. Click the 'Edit Review' button at the top of the screen.
2. The 'Edit Review' window will open, allowing you to change various aspects of the review.
3. Under 'Project' click the drop-down menu. This will allow you to select a new parent project for the review.
4. Click the 'Done' button at the bottom of the screen.

Screenshot: Changing a Review's Parent Project
Deleting an Abandoned Review

You can delete reviews that have been abandoned. To do this, follow the instructions below.

Deleted reviews cannot be retrieved.

Deleting Reviews from the ‘Abandoned’ list

To delete a review from the ‘Abandoned’ list:

1. From the ‘Crucible Dashboard’, click ‘My Abandoned Reviews’ in the left-hand navigation bar.
2. A list of abandoned reviews appears. Click the name of the review you wish to remove.
3. Once the review details are displaying, select ‘Delete’ from the ‘Tools’ menu. The review will be instantly deleted.

Screenshot: Deleting a Review in Crucible
Defining your Workflow

This document describes several forms of Crucible Workflow in detail. Depending on the size of your team, there are four different ways that a development team could use Crucible for code reviews. Choose the workflow which suits your team.

- Lightweight Code Commenting with Crucible (individual)
- One-to-One Reviews (Agile Pair)
- One-to-Many Reviews Without a Moderator (Agile Team)
- Formal Group Reviews (CMM Team)

**Lightweight Code Commenting with Crucible (individual)**

1. Author commits new work.
2. Author creates the review, and adds comments using the easy web interface.
3. Author summarizes and closes the review, saving the code comments in Crucible's database, which is stored outside the repository.

*Diagram: Workflow for Lightweight Code Commenting*
**One-to-One Reviews (Agile Pair)**

1. **Author** creates the review.
2. Author invites reviewer to take part in the review.
3. **Reviewer** creates comments on the code.
4. Author responds to reviewer comments.
5. Follow-up comments are made if necessary.
6. Reviewer finishes own review process.
7. Author summarizes and closes the review.

*Diagram: Workflow for One-to-One Reviews*
For more information on one-to-one reviews, see Getting Started with Crucible. The workflow process in Crucible is covered in detail within this document.

One-to-Many Reviews Without a Moderator (Agile Team)

1. Author creates the review.
2. Author invites reviewers to take part in the review.
3. Reviewers make comments on the code.
4. Author responds to reviewer comments, follow-up comments are made if necessary.
5. Reviewers complete their reviews.
6. Author summarizes and closes the review.

Diagram: Workflow for One-to-Many Reviews
Formal Group Reviews (CMM Team)

1. Author creates the review.
2. Moderator invites reviewers to take part the review.
3. Reviewers make comments on the code.
4. Author responds to reviewer comments.
5. Follow-up comments are made if necessary.
6. Each discussion point is settled by the Moderator.
7. Moderator summarizes and closes the review.

*Diagram: Workflow for Formal Group Reviews*
To see a simple example of how to use Crucible with two people, see Getting Started with Crucible.
Using Favourites

This page contains instructions on adding favourites in Crucible. You can select code reviews, changesets, files, people and repositories to be added to your favourites. This allows you to personalise the information that you see in your activity stream. We suggest you select items that you are currently working on as favourites, to create a more relevant personalised view.

See Viewing Your Favourites for instructions on how to view your existing list of favourites and how to rename and remove favourites.

On this page:

- Adding a Review to Your Favourites
- Adding a Review Comment Thread to Your Favourites
- Adding a Project to Your Favourites
- Adding a Person to Your Favourites
- Adding a Changeset to Your Favourites
- Adding a File or Folder to Your Favourites
- Adding a Repository to Your Favourites

Adding a Review to Your Favourites

To add a review to your favourites, hold the mouse cursor over the review name when it appears in a menu screen. The Review Hover menu appears. At the top right of the Review Hover menu, click the small grey cog icon that indicates the ‘Tools’ menu. The Tools menu opens. In the Tools menu, click ‘Add Star’. This will add it to your favourites.

Screenshot: Adding a Review To Your Favourites

Adding a Review Comment Thread to Your Favourites

To add a review comment thread to your favourites, click the link ‘Add Star’ next to the grey star icon at the bottom of the first comment of the thread. From then on, new comments will be shown in your favourites activity stream.

Screenshot: Adding a Review Comment Thread to Your Favourites
Adding a Project to Your Favourites

To add a project to your favourites, click the 'Projects' tab. The Projects view appears. Here, simply click the grey star icon that appears next to the desired project name. The star icon will turn yellow, showing that it is selected as a favourite.

Screenshot: Adding a Project to your Favourites

<table>
<thead>
<tr>
<th>Project name</th>
<th>Latest Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Project</td>
<td>on 24 Jun 2009</td>
</tr>
<tr>
<td>Confluence Hosted</td>
<td>on 25 Feb 2009</td>
</tr>
<tr>
<td>Jira Studio</td>
<td>on 22 Apr 2008</td>
</tr>
<tr>
<td>FishEye</td>
<td>on 24 Jun 2009</td>
</tr>
<tr>
<td>Clover</td>
<td>on 24 Jun 2009</td>
</tr>
<tr>
<td>Clover IDEA plugin</td>
<td>on 24 Jun 2009</td>
</tr>
<tr>
<td>Testing Project</td>
<td>on 23 Jun 2009</td>
</tr>
<tr>
<td>Clover Eclipse Plugin</td>
<td>on 24 Jun 2009</td>
</tr>
</tbody>
</table>

Adding a Person to Your Favourites

To add a person to your favourites, simply hold the mouse cursor over their username wherever it appears. The User Hover menu will appear. In the User Hover menu, click 'Follow'. This will add the person to your favourites.

Screenshot: Adding a Person to Your Favourites
Adding a Changeset to Your Favourites

To add a changeset to your favourites, firstly open the changeset desired from the 'Source' tab. Once the changeset is open in Crucible, simply click the grey star icon that appears next to its name. The name appears in the breadcrumb links at the top of the screen.

Screenshot: Adding a Changeset to Your Favourites

Adding a File or Folder to Your Favourites

To add a file to your favourites, firstly open the file or folder desired, from the 'Source' tab. Once the file or folder is open in Crucible, simply click the grey star icon that appears next to its name. The name appears in the breadcrumb links at the top of the screen.

Screenshot: Adding a File or Folder to Your Favourites

Adding a Repository to Your Favourites

Adding a repository to your favourites (requires FishEye), click the 'Source' tab. The the 'Source' view appears. Here, simply click the grey star icon that appears next to the name of the desired repository. The star icon will turn yellow, showing that it is selected.

Screenshot: Adding a Repository to Your Favourites

Using Keyboard Shortcuts in Crucible

Crucible provides a number of keyboard shortcuts, allowing you to quickly carry out certain actions without the mouse. Keyboard shortcuts are available for most of the commonly-used functions in Crucible.
To see a list of available shortcuts, firstly navigate to a review in Crucible. Now open the 'Tools' drop-down menu at the top right corner of the screen, and select the 'Keyboard Shortcuts' option.

See the tables below for full details:

### General Shortcuts

<table>
<thead>
<tr>
<th>Key</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>Opens reference list of keyboard shortcuts.</td>
</tr>
<tr>
<td>escape</td>
<td>Closes reference list of keyboard shortcuts.</td>
</tr>
<tr>
<td>alt</td>
<td>Hold down then click and drag to select source line contents.</td>
</tr>
<tr>
<td>shift + f</td>
<td>Toggle full screen review mode.</td>
</tr>
</tbody>
</table>

### Comment Navigation Shortcuts

<table>
<thead>
<tr>
<th>Key</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Go to next review comment.</td>
</tr>
<tr>
<td>p</td>
<td>Go to previous review comment.</td>
</tr>
<tr>
<td>shift + p</td>
<td>Go to first review comment.</td>
</tr>
<tr>
<td>shift + n</td>
<td>Go to last review comment.</td>
</tr>
<tr>
<td>l</td>
<td>Go to next thread (skips replies).</td>
</tr>
<tr>
<td>h</td>
<td>Go to previous thread (skips replies).</td>
</tr>
<tr>
<td>]</td>
<td>Go to next unread comment.</td>
</tr>
<tr>
<td>[</td>
<td>Go to previous unread comment.</td>
</tr>
<tr>
<td>r</td>
<td>Reply to a comment.</td>
</tr>
<tr>
<td>m</td>
<td>Toggle comment read/unread status.</td>
</tr>
</tbody>
</table>

### File Navigation Shortcuts

<table>
<thead>
<tr>
<th>Key</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>Go to next file.</td>
</tr>
<tr>
<td>k</td>
<td>Go to previous file.</td>
</tr>
<tr>
<td>shift + k</td>
<td>Go to first file.</td>
</tr>
<tr>
<td>shift + j</td>
<td>Go to last file.</td>
</tr>
<tr>
<td>u</td>
<td>Go to next unreviewed file.</td>
</tr>
<tr>
<td>i</td>
<td>Go to previous unreviewed file.</td>
</tr>
<tr>
<td>y</td>
<td>Set file reviewed and go to next unreviewed file.</td>
</tr>
<tr>
<td>shift + y</td>
<td>Toggle file reviewed/unreviewed status.</td>
</tr>
<tr>
<td>e</td>
<td>Expand current file.</td>
</tr>
<tr>
<td>c</td>
<td>Collapse current file.</td>
</tr>
<tr>
<td>shift + e</td>
<td>Expand all files.</td>
</tr>
<tr>
<td>shift + c</td>
<td>Collapse all files.</td>
</tr>
</tbody>
</table>

### Using RSS Feeds in Crucible
**Subscribing to an RSS Feed**

In Crucible, all pages with an activity stream and any page which has a list of reviews will have an RSS option.

To access the RSS feed for a page, open the 'Tools' drop-down menu at the top right corner of the screen, then click the 'RSS' option. This will open a page with the RSS feed displayed; you can also paste the URL from that page into your RSS reader of choice.

**Using Wiki Markup in Crucible**

Crucible supports Wiki Markup text formatting in comments and review descriptions.

The text markup notation on this page is a reference showing the available formatting commands.

⚠️ When using FishEye, you can also render Wiki Markup in commit messages.

### Headings

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>h1.Biggest heading</td>
<td>Turns text into a heading at size 1.</td>
</tr>
<tr>
<td>h2.Bigger heading</td>
<td>Turns text into a heading at size 2.</td>
</tr>
<tr>
<td>h3.Big heading</td>
<td>Turns text into a heading at size 3.</td>
</tr>
<tr>
<td>h4.Normal heading</td>
<td>Turns text into a heading at size 4.</td>
</tr>
<tr>
<td>h5.Small heading</td>
<td>Turns text into a heading at size 5.</td>
</tr>
<tr>
<td>h6.Smallest heading</td>
<td>Turns text into a heading at size 6.</td>
</tr>
</tbody>
</table>

### Text Effects

Text effects are used to change the formatting of words and sentences.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>bold</em></td>
<td>Makes text appear <strong>bold</strong>.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Makes text appear in <em>italics</em>.</td>
</tr>
<tr>
<td>+underline+</td>
<td>Makes text appear underlined.</td>
</tr>
<tr>
<td>??citation??</td>
<td>Makes text appear in citation form.</td>
</tr>
</tbody>
</table>
### Text Breaks

Wiki Markup allows you to insert breaks or different kinds of hyphens and dashes.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(empty line)</td>
<td>Produces a new paragraph</td>
</tr>
<tr>
<td>\ \</td>
<td>Creates a line break.</td>
</tr>
<tr>
<td>----</td>
<td>Creates a horizontal ruler.</td>
</tr>
<tr>
<td>--</td>
<td>Produces em dash — symbol.</td>
</tr>
<tr>
<td>---</td>
<td>Produces en dash – symbol.</td>
</tr>
</tbody>
</table>

### Links

Creating links is easy with Wiki Markup.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="CR-FE-100">Crucible Review CR-FE-100</a></td>
<td>Creates a link to a Crucible review or FishEye artifact using the internal key reference for the item.</td>
</tr>
</tbody>
</table>
[Atlassian Crucible](http://atlassian.com) creates a link to an external resource, special characters that come after the URL and are not part of it must be separated with a space. External links are denoted with an arrow icon.

Examples:

- [http://www.atlassian.com/crucible](http://www.atlassian.com/crucible)
- [Atlassian Crucible](http://atlassian.com/crucible)

Note: The square brackets [], around external links are optional in the case you do not want to use any alternate text for the link (i.e. just display the raw URL).

[mailto:mail@example.com] creates a link to an email address.

Example:

mail@example.com

[file:///c:/temp/foo.txt] creates a download link to a file on your computer or on a network share that you have mapped to a drive. To access the file, you must right click on the link and choose "Save Target As".

(anchor:anchorname) creates a bookmark anchor inside the page. You can then create links directly to that anchor. So a link like this: [My Page#here] will link to wherever in "My Page" there is an {anchor:here} macro, and the link [#there] will link to wherever in the current page there is an {anchor:there} macro.

### Lists

Lists allow you to present information as a series of ordered items. Use asterisks * for bulleted lists and hash symbols # for numbered lists.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* A bulleted list</td>
<td>A bulleted list</td>
</tr>
<tr>
<td>* Second item</td>
<td>Second item</td>
</tr>
<tr>
<td>** indented item 1</td>
<td>indented item 1</td>
</tr>
<tr>
<td>** indented item 2</td>
<td>indented item 2</td>
</tr>
<tr>
<td># A numbered list</td>
<td>A numbered list</td>
</tr>
<tr>
<td># Second item</td>
<td>Second item</td>
</tr>
<tr>
<td>## indented item 1</td>
<td>indented item 1</td>
</tr>
<tr>
<td>## indented item 2</td>
<td>indented item 2</td>
</tr>
</tbody>
</table>

### Images

Images can be referenced from remote sources only.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="http://www.host.com/image.gif" alt="http://www.host.com/image.gif" /></td>
<td>The image will be displayed from the remote source.</td>
</tr>
<tr>
<td><img src="http://www.host.com/image.gif" alt="http://www.host.com/image.gif" /></td>
<td>align=right,vspace=4!</td>
</tr>
</tbody>
</table>

### Tables

Tables allow you to organise content in a rows and columns, with a header row if required.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
</table>

The code above produces a table that looks like this:

<table>
<thead>
<tr>
<th>heading 1</th>
<th>heading 2</th>
<th>heading 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>col A1</td>
<td>col A2</td>
<td>col A3</td>
</tr>
<tr>
<td>col B1</td>
<td>col B2</td>
<td>col B3</td>
</tr>
</tbody>
</table>
**Advanced Formatting**

This section covers panels, code windows and showing plain text with no formatting.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(noformat)</td>
<td>Makes a preformatted block of text with no syntax highlighting. All the optional parameters of the {noformat} macro are valid for the {panel} macro as well. Example:</td>
</tr>
</tbody>
</table>

```
This is a no-formatted piece of text, so "no" _formatting_ is done here.
```

<table>
<thead>
<tr>
<th>(panel)</th>
<th>Embraces a block of text within a fully customizable panel. The optional parameters you can define are as follows.</th>
</tr>
</thead>
<tbody>
<tr>
<td>title: Title of the panel</td>
<td></td>
</tr>
<tr>
<td>borderStyle: The style of the border this panel uses (solid, dashed and other valid CSS border styles)</td>
<td></td>
</tr>
<tr>
<td>borderColor: The color of the border this panel uses</td>
<td></td>
</tr>
<tr>
<td>borderWidth: The width of the border this panel uses</td>
<td></td>
</tr>
<tr>
<td>bgColor: The background color of this panel</td>
<td></td>
</tr>
<tr>
<td>titleBGColor: The background color of the title section of this panel</td>
<td></td>
</tr>
</tbody>
</table>

Examples:

```
Some text in a basic panel
```

```
My Title

Some text with a title
```

```
My Title

a block of text surrounded with a panel
yet another line
```
The code macro displays a preformatted block for showing code with syntax highlighting. All the optional parameters of the {panel} macro are valid for {code}. The default language is Java but you can specify JavaScript, ActionScript, XML or SQL.

Examples:

**Java with a title bar:**

```java
@title=Bar.java|borderStyle=solid
// Some comments here
public String getFoo(){
    return foo;
}
```

**A basic display with XML code:**

```xml
<test>
    <another tag="attribute"/>
</test>
```

**Miscellaneous Markup Features**

Emoticons and often-used images can be easily embedded with the following Wiki Markup Syntax:

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\X</td>
<td>Escape special character X (i.e. {)</td>
</tr>
<tr>
<td>:)</td>
<td>Graphical emoticons (smileys): 😊, 😊</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notation</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>:)</td>
<td>😊😊😊😊😊😊😊😊😊</td>
</tr>
<tr>
<td>:P,D,J,y,(n),(i),(x)</td>
<td>(y)</td>
</tr>
<tr>
<td>(+) (-) (?) (on) (off) (*) (*r) (*g) (*b) (*y)</td>
<td></td>
</tr>
</tbody>
</table>

**Using Gadgets in Crucible**

This page explains how to use the bundled gadgets in Crucible.

**On this page:**

- Overview of Crucible Gadgets
- Gadget Configuration 1: Add JIRA to FishEye as an OAuth consumer
- Gadget Configuration 2: Add Gadget to your Application's Gadget Directory
- Gadget Configuration 3: Add Gadget to the Application's Dashboard

**Overview of Crucible Gadgets**

As of the release of Crucible 2.3, you can show Crucible data in other Atlassian applications such as JIRA and Confluence by way of special gadgets.

Crucible has three gadgets bundled with it by default:

<table>
<thead>
<tr>
<th>Gadget Name</th>
<th>Description and Gadget URL</th>
</tr>
</thead>
</table>
### To Do Gadget
This gadget is a list of Crucible to-do items including reviews to do, comments to read or reviews to summarise.

The URL for this gadget is as follows:

```
http://HOSTNAME:8060/rest/gadgets/1.0/g/com.atlassian.fecru.fecru-gadgets-plugin:overdueReviews/gadgets/todo.xml
```

In this example, `HOSTNAME:8060` is the hostname of your Crucible instance.

### Hassle Gadget
This gadget shows you who you are still waiting on, in other words which reviewers haven’t completed your reviews.

The URL for this gadget is as follows:

```
http://HOSTNAME:8060/rest/gadgets/1.0/g/com.atlassian.fecru.fecru-gadgets-plugin:overdueReviews/gadgets/hassle.xml
```

In this example, `HOSTNAME:8060` is the hostname of your Crucible instance.

### Overdue Reviews Gadget
This gadget shows you reviews that are yet to be completed in the project, across all authors. This is useful for managers or team leads.

The URL for this gadget is as follows:

```
http://HOSTNAME:8060/rest/gadgets/1.0/g/com.atlassian.fecru.fecru-gadgets-plugin:overdueReviews/gadgets/overdueReviews.xml
```

In this example, `HOSTNAME:8060` is the hostname of your Crucible instance.

### Review Coverage
This gadget shows content from the innovative Review Coverage report, letting you investigate how much of your codebase has been reviewed.

The URL for this gadget is as follows:

```
http://HOSTNAME:8060/rest/gadgets/1.0/g/com.atlassian.fisheye.review-coverage-report/gadget/recent-changesets.xml
```

In this example, `HOSTNAME:8060` is the hostname of your Crucible instance.

Configuring gadgets is a three phase process. Firstly, you add your JIRA instance as an OAuth consumer. Secondly, you’ll add the gadget to the destination application, then finally you add the gadget to the application’s dashboard. In our example, we will show how to configure the Crucible gadgets for use in JIRA.

### Gadget Configuration 1: Add JIRA to FishEye as an OAuth consumer

Firstly, you need to add JIRA to FishEye as an OAuth consumer. To do this, open the Admin Screen, then click ‘Open Authentication (OAuth)’ under ‘Global Settings’ in the left navigation bar. The OAuth configuration screen opens. Click ‘Add OAuth Consumer’.

**Screenshot: The OAuth Administration Screen**

<table>
<thead>
<tr>
<th>OAuth Administration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Consumers</strong></td>
</tr>
</tbody>
</table>

An OAuth consumer is a web site or application that has permission to access FishEye’s data on behalf of the logged-in user. Typically, the consumer accesses the data via gadgets that support OAuth for authorization. Learn more.

Add OAuth Consumer

OAuth is a protocol used when one web application needs to identify itself to another. When authorization is done via OAuth, users can log in to one application (the service provider) and access that application’s information from another application (the consumer) without giving their password to the second application. Typically, the consumer accesses the data via gadgets that support OAuth for authorization.

Now, copy the URL for your JIRA instance into the field labelled ‘Consumer Base URL’ then click ‘Add’. The application in use (JIRA or Confluence) will be auto-detected.

**Screenshot: Adding Consumers to OAuth**
Gadget Configuration 2: Add Gadget to your Application's Gadget Directory

As a JIRA administrator you allow the use of these gadgets by adding them to the Gadget Directory. For each gadget, you will need to complete and enter the URL listed in the table above.

See the JIRA documentation for details on this process.

Gadget Configuration 3: Add Gadget to the Application's Dashboard

Finally, as a JIRA user, you need to add the gadget to your dashboard:

See the JIRA documentation for details on this process.

Once complete, the gadget will appear on your JIRA dashboard and display information drawn from Crucible and FishEye.

Confluence also allows gadgets to be added to its dashboard. See the General Gadgets Documentation for more information.
**Crucible: Hassle**

<table>
<thead>
<tr>
<th>reviewer</th>
<th>ID</th>
<th>Name</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CR-1</td>
<td>FE-1234: fix Widget stamping (1 unread comment)</td>
<td>in 6 days</td>
</tr>
<tr>
<td></td>
<td>CR-1</td>
<td>FE-1234: fix Widget stamping (1 unread comment)</td>
<td>in 6 days</td>
</tr>
<tr>
<td></td>
<td>CR-4</td>
<td>FE-4567: Disentangle states</td>
<td>in 10 hours</td>
</tr>
</tbody>
</table>

**Screenshot: The Overdue Reviews Gadget**

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR-3</td>
<td>FE-3456: Make stuff nicer</td>
<td>49 minutes ago</td>
</tr>
<tr>
<td>CR-4</td>
<td>FE-4567: Disentangle states</td>
<td>in 10 hours</td>
</tr>
</tbody>
</table>

**Screenshot: The Changeset Review Coverage Gadget**

<table>
<thead>
<tr>
<th>User</th>
<th>Message</th>
<th>% Reviewed</th>
<th>Files</th>
<th>Lines</th>
<th>Reviews</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Remove System.out.println()</td>
<td></td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>Avoid NPE in eyeql with no returns</td>
<td></td>
<td>1</td>
<td>9</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>NONE: If the 500 page throws an</td>
<td></td>
<td>1</td>
<td>123</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>NONE: fix tests</td>
<td></td>
<td>1</td>
<td>6</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>Remove TODO in error message</td>
<td></td>
<td>1</td>
<td>2</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3529: Fix NPE linking to re</td>
<td></td>
<td>2</td>
<td>16</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>Fix checkstyle</td>
<td></td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3529: Revert RSS links</td>
<td></td>
<td>1</td>
<td>2</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3529: Rewrite /snippet/cre</td>
<td></td>
<td>6</td>
<td>51</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3543: Fix bug where user!=</td>
<td></td>
<td>1</td>
<td>28</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>branch for 2.3.x</td>
<td></td>
<td>6904</td>
<td>609616</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3548: fix Crucible inclusion</td>
<td></td>
<td>3</td>
<td>8</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3445: snippet comment pr</td>
<td></td>
<td>9</td>
<td>129</td>
<td>-</td>
<td>May 19</td>
</tr>
<tr>
<td></td>
<td>CRUC-3440: review rework</td>
<td></td>
<td>1</td>
<td>2</td>
<td>-</td>
<td>May 19</td>
</tr>
</tbody>
</table>
Crucible Administrator's Guide

Once you have installed and configured Crucible, you can access the Administration pages at the following address (where 'HOSTNAME' is the name of the server where you installed Crucible).

http://HOSTNAME:8060/admin/

The 'Admin Menu' allows you to administer your Crucible instance and manage your repositories, users and back-end settings. For information on administering FishEye, please refer to the FishEye documentation.

Topics

- Administering Projects
- Backing Up and Restoring Crucible Data
- Creating a Permission Scheme
- Crucible and FishEye
- Customising Email Notifications
- Customising the Defect Classifications
- Customising the Welcome Message
- JIRA Integration in Crucible
- Migrating to an External Database
- Creating a User
- Trusted Applications
- Setting up Users and Security
- Enabling Access Logging in Crucible
- Configuring User Managed Mappings

Administering Projects

A Crucible project is a collection of reviews, typically reviews that all relate to the same application. In addition to providing a logical way of grouping reviews together, a project allows you to

- define default moderators, authors and reviewers for the reviews in that project.
- define which people are eligible to be reviewers for the reviews in that project.
- use permission schemes to restrict who can perform particular actions (e.g. 'Create Review') in that project.

Every Crucible review belongs to a project. Each project has a name (e.g. ACME Development) and a key (e.g. ACME). The project key becomes the first part of that project's review keys, e.g. ACME-101, ACME-102, etc:

By default, Crucible contains one project. This default project has the key 'CR' and the name 'Default Project'.

To view your projects,

1. From the 'Admin Menu', click 'Project List'.
2. The 'Projects List' page will be displayed.
   • Click the 'Create a New Project' link, which appears at the bottom of the list of existing projects, to create a new project. See Creating a Project.
   • Click the 'Edit' link next to an existing project's name, to edit the project. See Editing a Project.
   • Click the 'Delete' link next to an existing project's name, to edit the project. See Deleting a Project.

Creating a Project

A Crucible project is a collection of reviews, typically reviews that all relate to the same application. In addition to providing a logical way of grouping reviews together, a project allows you to

- define default moderators, authors and reviewers for the reviews in that project.
- define which people are eligible to be reviewers for the reviews in that project.
- use permission schemes to restrict who can perform particular actions (e.g. 'Create Review') in that project.

Every Crucible review belongs to a project. Each project has a name (e.g. ACME Development) and a key (e.g. ACME). The project key becomes the first part of that project's review keys, e.g. ACME-101, ACME-102, etc:

By default, Crucible contains one project. This default project has the key 'CR' and the name 'Default Project'.
To create a new project,

1. Click the menu labelled with your user name in the FishEye/Crucible header, and click the ‘Administration’ option. You will need to be logged in as an administrator to see this link. The FishEye/Crucible administration console will be displayed.

2. Click ‘Projects’ link in the left menu. The ‘Projects’ page will be displayed showing all of the projects that you have set up in Crucible.

3. Click the ‘Create a New Project’ link, at the bottom of the list of your existing projects.

4. The ‘Edit Project’ page will be displayed. Complete the fields on this page:
   - ‘Name’ — Type a short phrase that describes your project.
   - ‘Key’ — Type a few characters to uniquely identify your project. This key must consist of alphabetic and/or numeric characters and hyphens only.
   - ‘Default Repository’ — Select the repository which contains source code relating to this project.

   This repository is the one that will be searched by default when you add files to a review.

   - ‘Store the contents of files in reviews’ — Tick this checkbox if you want to store the contents of files included in reviews, in the reviews themselves.

   - ‘Permission Scheme’ — Select the relevant permission scheme for this project. (A permission scheme controls who can perform particular actions, e.g. ‘Create Review’)

   - ‘Enable the Moderator role for this project’ — Tick this checkbox if you want to enable the moderator role for this project.

   - ‘Default Moderator’ — Type the name of the person who will appear by default in the ‘Moderator’ field when you create a new review; or leave this field blank to force the review’s creator to choose a moderator.

   - ‘By default, allow anyone to join reviews after creation’ — Tick this checkbox, if you want to allow anyone to join reviews after creation. See Adding Reviewers.

   (Optional) Under ‘Default Reviewers’, select the people to whom new reviews in this project will be assigned by default:

   - ‘Users’ — Type the name(s) of individual users to whom new reviews will be assigned by default.

   - ‘Groups’ — Type the name(s) of groups to whose members new reviews will be assigned by default.

   (Optional) Under Allowed Review Participants, select who will be allowed to have a role (i.e. be an author/creator/moderator/reviewer) in this project’s reviews:

   - ‘Users’ — Type the name(s) of individual users who will be eligible to be authors/creators/moderators/reviewers for reviews in this project*.

   - ‘Groups’ — Type the name(s) of groups whose members will be eligible to be authors/creators/moderators/reviewers for reviews in this project*.

   * These users will be the only ones whose names appear when a review is assigned.

   - ‘Default duration in week days’ — Enter a value for the number of working days that you want the review to run for. Simply type the number of days into the text entry field marked ‘Default duration in week days’.

   - ‘Default Review Objectives’ — Enter review objectives that will be applied to all new reviews. Click the ‘Preview’ button to preview the text entered in this field.

5. Click the ‘Save’ button to create your new project.
When creating a project or editing a project's properties, you can set Crucible to save all revisions that are associated with a review to Crucible's database. This allows you to be able to view that file content whether or not the repository is online or accessible to Crucible. It also creates an enhanced audit trail should you require it, saving the review content regardless of whether or not it is deleted or lost from the repository.

Note that the storage of revisions must be set per-project. Also, the storage only applies to reviews created after Revision Storage is enabled. This means that for existing projects, pre-existing reviews will not be stored unless you look at them again after Revision Storage is enabled.

Enabling Revision Storage on a new project

To enable Revision Storing on a new project,
When creating a new project, you have the option to turn on revision storing on the 'Create New Project' page.
1. Under 'Default Content Review Repository', click the checkbox labelled 'Store the contents of files in reviews'.
2. Click 'Save' to finish.

Enabling Revision Storage on an existing project

To enable Revision Storing on an existing project,

1. From the 'Admin' screen, click 'Projects' from the left navigation bar.
2. Click 'Edit' next to the desired project.
3. Under 'Default Content Review Repository', click the checkbox labelled 'Store the contents of files in reviews'.
4. Click 'Save' to finish.

Screenshot: Enabling Revision Storage

Default Review Content Repository

<table>
<thead>
<tr>
<th>Default Repository:</th>
<th>CLOV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store the contents of files in reviews</td>
<td></td>
</tr>
</tbody>
</table>

Deleting a Project

Admin users can delete projects under Crucible. To do this, follow the instructions below.

Deleting Projects from the Project List

Before you begin:
- By default, Crucible contains one project. This default project has the key 'CR' and the name 'Default Project'. This project cannot be deleted.
- Deleted projects cannot be recovered.

To delete a project from the Project List:

1. From the 'Admin Menu', click 'Project List'.
2. A list of projects appears. With care, click the 'Delete' link situated to the right of the project you wish to remove. If empty, the project instantly disappears.
3. If the project contains reviews, you will be prompted to either delete all reviews in the project, or move them into the default project.

Screenshot: The Crucible Project Listing

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
<th>Default Repository</th>
<th>Default Moderator</th>
<th>Default Reviewer Users</th>
<th>Default Reviewer Groups</th>
<th>Default Review Time</th>
<th>Edit</th>
<th>Edit FishEye Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR</td>
<td>Default Project</td>
<td>CLOV</td>
<td></td>
<td></td>
<td></td>
<td>No time restriction</td>
<td>edit</td>
<td>view/edit</td>
</tr>
<tr>
<td>CR-CH</td>
<td>Confluence</td>
<td>CH</td>
<td></td>
<td></td>
<td></td>
<td>No time restriction</td>
<td>edit</td>
<td>view/edit</td>
</tr>
<tr>
<td>CR-ST</td>
<td>Jira Studio</td>
<td>JST</td>
<td>Don Brown</td>
<td></td>
<td></td>
<td>No time restriction</td>
<td>edit</td>
<td>view/edit</td>
</tr>
</tbody>
</table>
**Editing a Project**

Once projects are created, you can return to the project settings page to change the defaults such as repository, moderator, allowed reviewers, allowed groups and permissions.

**To edit project settings,**

1. From the 'Admin Menu', click 'Project List'.
2. The list of projects will be displayed. Click the 'Edit' link for the desired project, which appears to the right of the existing project name.
3. The 'Edit Project' page will be displayed. You can now adjust any of the given settings as desired.
4. In the 'Identification' section, you can change the plain language name (as displayed in the Crucible interface) and the project key (used when giving reviews their unique code names).
5. In the 'Default Review Content Repository' field, you can adjust the repository which contains source code relating to this project.
   - This repository is the one that will be searched by default when you add files to a review.
   - The check box here labelled 'Store the contents of files in reviews' will cause the source files under review to be stored in the Crucible database along with the comments and review data. This will retain a copy of all the source files that go under review even in the event of disconnecting the repository from Crucible.
6. In the 'Default Moderator' field, you can adjust the name of the person who will appear by default in the 'Moderator' field when you create a new review; or leave this field blank to force the review's creator to choose a moderator.
7. (Optional) Under 'Default Reviewers', you can adjust the people to whom new reviews in this project will be assigned by default:
   - Select the 'Let allowed review participants join a review' check-box if you wish to determine the default for the 'Allow anyone to join' option on the 'Adding Reviewers' screen.
   - In the 'Users' field, you can adjust the name(s) of individual users to whom new reviews will be assigned by default.
   - In the 'Groups' field, you can adjust the name(s) of groups to whose members new reviews will be assigned by default.
8. (Optional) Under 'Allowed Review Participants', you can adjust who will be allowed to have a role (i.e. be an author/creator/moderator/reviewer) in this project's reviews:
   - In the 'Users' field, you can adjust the list of individual users who will be eligible to be authors/creators/moderators/reviewers for reviews in this project.
   - In the 'Groups' field, you can adjust the list of groups whose members will be eligible to be authors/creators/moderators/reviewers for reviews in this project.
   - * These users will be the only ones whose names appear when a review is assigned.
9. In the 'Permission Scheme' drop-down list, you can adjust the relevant permission scheme for this project. (A permission scheme controls who can perform particular actions, e.g. 'Create Review'.)
10. In the 'Review Duration' section, you can define the default length of time (in week days) for reviews in this project.
11. In the 'Default Review Objectives Section', you can define some text that will appear by default in the Review Objectives field of each new review. This text can be edited, as any text contained in the Review Objectives text box can.

*Screenshot: The Edit Project screen in Crucible*
Enabling the Moderator Role

This page contains instructions on how to enable or disable the moderator role for all reviews under a given Crucible project.

**On this page:**

- Introduction
- Enabling the Moderator Role
- Removing the Moderator from an Existing Project
- Adding the Moderator to an Existing Project

**Introduction**

By default, Crucible projects do not have a moderator. This allows for a streamlined review handling process, where the review author is the sole person who starts and stops the review. Projects in Crucible can have the moderator role enabled or disabled.

The setting for enabling moderators can only be set by a Crucible user with admin privileges.

**Enabling the Moderator Role**
The moderator role is configurable in Crucible as a per-project setting. By default, all reviews have an author and a moderator. However, the moderator role can be disabled.

To enable or disable the moderator role on a project:

1. Open the Crucible Admin screen, by clicking the 'Administration' link at the bottom of any Crucible page.
2. The Crucible Admin screen opens. Click 'Project List' under 'Project Settings' in the left navigation bar.
3. The Projects List screen opens. Click the 'Edit' link in the 'Edit Crucible Settings' column.
4. The 'Edit Project' screen opens.

Removing the Moderator from an Existing Project

When you remove the moderator role from an existing project, be aware of the following points:

- Reviews created after the setting change will have the moderator role removed.
- Reviews created prior to the change will still retain the moderator they were assigned.
- If the removal of the moderator conflicts with other Crucible project settings, a warning will be shown on the Project List page.

If in doubt about the impact of adding or removing the moderator, you can create a new project (and set the moderator status during the project's creation).

Adding the Moderator to an Existing Project

If you add the moderator role back in to an existing project, be aware of the following points:

- Reviews created after the setting change will have the moderator role added.
- Reviews created prior to the change will still have no moderator.
- If the addition of the moderator conflicts with other Crucible project settings, a warning will be shown on the Project List page.

If in doubt about the impact of adding the moderator, you can create a new project (and set the moderator status during the project's creation).

Setting Default Review Objectives

To set default review objectives for all the reviews in a given project, carry out the following steps:
2. In the ‘Default Review Objectives Section’, you can define some text that will appear by default in the ‘Review Objectives’ field of each new review. Click ‘Done’ to save your changes. This text can be edited, as any text contained in the Review Objectives text box can.

Screenshot: Default Review Objectives in Crucible

Setting the Default Review Duration for a Project

You can set a default time period (duration) that all reviews under a given project will run for. Reviews that are overdue will show up in red on the reviewer’s dashboards.

To set a default duration for all reviews in a project,

1. From the ‘Admin Menu’, click ‘Project List’.
2. The list of projects will be displayed. Click the ‘Edit’ link for the desired project, which appears to the right of the existing project name.
3. The ‘Edit Project’ page will be displayed. You can now adjust any of the given settings as desired.
4. In the ‘Review Duration’ section, you can define the default length of time (in week days) for reviews in this project. If you leave the field blank, then no time restriction is applied.

Note that the ‘Review Duration’ only affects the default due date that appears when creating a review. The review’s creator or moderator can specify a different date if they wish.

To see instructions for the other items on this page, see the documentation for Editing a Project.

Backing Up and Restoring Crucible Data

Crucible data can be backed up from the admin interface or command line. This page contains the command syntax, options and the required procedure to backup and restore your Crucible instance.

On this page:
- Backing Up Crucible Data
  - The Crucible Admin Interface Backup Process
  - The Crucible Command Line Backup Process
  - Components of a Crucible Backup
  - Backup Command Line Options
    - Command Line Examples
    - Advanced Backup Command Line Settings
    - Known Limitations
  - Scheduling Crucible Backups
- Restoring Crucible Data
  - The Crucible Data Command Line Restoration Process
  - Restore Command Line Options
  - Advanced Command Line Restore Settings
    - Notes on Migrating Backup Data
    - Command Line Example: Migrating Backup Data to MySQL

Backing Up Crucible Data

The Crucible Admin Interface Backup Process
1. Navigate to the Crucible 'Admin' screen (click the ‘Administration’ link in the footer of any Crucible page).
2. On the Admin screen, click ‘Backup’ under the ‘System’ heading in the left navigation bar. The Backup screen opens.
3. On the Backup screen, the 'File Path' field indicates where the backup file (in .zip format) will be stored. You can manually edit this path to change it. Under the heading ‘Include’, a list of checkboxes is shown, with the following items:
   - Plugins and their configuration data
   - SQL database
   - Web templates
   - Uploaded files and local copies of files under review.
   - Repository and application caches.

   Repository and application caches contain temporary data stored from repository scans and library caches that improve startup time. Both will be recreated automatically by re-scanning the source repositories, so the backup files can be reduced by a significant amount by excluding these (if the cost of re-scanning is acceptable).

4. Once you have chosen your options, click ‘Create Backup Now’.

   Screenshot: The Crucible Backup Screen

   ![Screenshot of the Crucible Backup Screen]

The Crucible Command Line Backup Process

Your Crucible instance must be running during the backup.

1. Open a command line interface on the Crucible server computer.
2. Navigate to the FISHEYE_HOME/bin/ directory.
3. Run the backup command on the command line with the desired options.
4. The backup is created as a new Zip archive file and placed in the FISHEYE_INST/backup/ directory.

   Note that if your Crucible instance uses a custom FISHEYE_INST directory, make sure the environment variable is properly set when running the backup command.

Components of a Crucible Backup

The Crucible backup is highly configurable and allows for many different configurations. This table shows the various components of the backup, what they are for and how they can be used.

<table>
<thead>
<tr>
<th>Component</th>
<th>Purpose</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Database</td>
<td>Refers to the SQL content database (used by both FishEye and Crucible and containing all user profile data, reviews and their comments).</td>
<td>Backed up by default.</td>
</tr>
<tr>
<td>Cache</td>
<td>The cache contains data that reflects the state of FishEye's repositories. Without it, FishEye must re-scan its repositories after a backup is restored. The cache also contains OSGI library data that increases startup time. These too can be excluded and will be generated automatically when the application is started.</td>
<td>The cache is not backed up by default as it tends to be large (running a risk of pushing the maximum file size for Java backups), whilst also representing replaceable data.</td>
</tr>
</tbody>
</table>
Plugins

Plugins are 3rd-party extensions that you may have installed, and configuration for all plugins (this includes configuration for Crucible's set of standard plugins).

Configuration data for all plugins are backed up by default, as well as all plugins installed in `FISHEYE_INST/var/plugins/user`.

Templates

In this context, these are custom freemarker templates that you or your users have created. They live in `FISHEYE_INST/template`.

Templates are backed up by default. You can choose to exclude them from the backup if your templates directory is covered by some other backup mechanism.

Uploads

In this context, uploads refers to files which are added to Crucible via the web interface (such as patch file reviews). It also includes each repository-backed file that went under review, when Crucible is configured to make a local copy of every reviewed file.

Uploads are backed up by default. You can choose not to back them up for example when the `FISHEYE_INST/var/data/uploads` directory is already covered by some other backup mechanism.

Note that the backup will always include the configuration data (`config.xml`), your license file and the FishEye user data.

Backup Command Line Options

These examples are for use in a Linux-like operating system. When using these commands on Windows, use the filename `fisheyectl.bat` and use the correct slashes. Run the command from the `FISHEYE_HOME/bin/` directory.

The basic syntax of the backup command is as follows:

```
$ ./fisheyectl.sh backup [OPTIONS]
```

To see inline help for all backup options, run the following command in the `FISHEYE_HOME/bin/` directory:

```
$ ./fisheyectl.sh backup --help
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Switch</th>
<th>Description</th>
<th>Default setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quiet mode</td>
<td>-q</td>
<td>Suppresses output</td>
<td>No</td>
</tr>
<tr>
<td>Output filename</td>
<td>-f</td>
<td>Specify a different path and filename to the <code>FISHEYE_INST/backup/backup_YYYY-DD-MM_HHmm.zip</code> file. When filename is omitted, the backup filename contains the date and time.</td>
<td>FISHEYE_INST/backup/ is the default directory.</td>
</tr>
<tr>
<td>Compression level</td>
<td>--compression OR -c</td>
<td>Sets the Zip compression level, from 1-9. Runs at level 6 if no argument is passed.</td>
<td>Yes (6)</td>
</tr>
<tr>
<td>Anonymise</td>
<td>-a</td>
<td>Anonymises the SQL database by replacing all text with 'x'. This is only useful when sending a backup to Atlassian as part of a support case. Please do not anonymise data unless the Support Engineer handling your support case has specifically requested the data anonymised (as often anonymised data will not help reproduce the issue).</td>
<td>No</td>
</tr>
<tr>
<td>Cache Backup</td>
<td>--cache</td>
<td>Include the repository caching files in the backup. These hold information gained from scanning the repositories and can be quite large (many gigabytes). However, it can shorten the time needed to re-scan the repositories after data is restored.</td>
<td>No. By default, the cache data is excluded from backups.</td>
</tr>
</tbody>
</table>

Command Line Examples

These examples are for use in a Linux-like operating system. When using these commands on Windows, use the filename `fisheyectl.bat` and use the correct slashes. Run the command from the `FISHEYE_HOME/bin/` directory.

Backing up with compression of 9, quiet mode and setting an output location

```
$ ./fisheyectl.sh backup --compression 9 -q -f /application_backups/fisheye/20090215.zip
```

Backup including cache data (also includes all default components)

```
$ ./fisheyectl.sh backup --cache
```
Restoring a backup with cache data (also restores all default components)

$ ./fisheyectl.sh restore --cache

Advanced Backup Command Line Settings

In some cases it might be preferable to only backup a limited set of items. This could be useful when your instance uses an external database such as MySQL or PostgreSQL and your DBA has already configured automatic backups in the database. The commands below allow this.

<table>
<thead>
<tr>
<th>Option</th>
<th>Switch</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclude Plugins</td>
<td>--no-plugins</td>
<td>Excludes plugins from the backup.</td>
<td>No. By default, plugins are included in every backup.</td>
</tr>
<tr>
<td>Exclude Templates</td>
<td>--no-templates</td>
<td>Excludes templates from the backup.</td>
<td>No. By default, templates are included in every backup.</td>
</tr>
<tr>
<td>Exclude Uploads</td>
<td>--no-uploads</td>
<td>Excludes uploaded files (such as patch reviews, stored in Crucible's internal database) from the backup.</td>
<td>No. By default, uploads are included in every backup.</td>
</tr>
<tr>
<td>Exclude SQL Database</td>
<td>--no-sql</td>
<td>Excludes the SQL content database used by both FishEye and Crucible.</td>
<td>No. By default, this data is included in every backup.</td>
</tr>
<tr>
<td>Show help</td>
<td>--help OR -h</td>
<td>Shows inline help on the command line.</td>
<td>No</td>
</tr>
</tbody>
</table>

Known Limitations

Please note that the below limitations are common for any Java based backup tool.

Archives Containing Over 65535 Files

Versions of Java earlier than v1.6 (b25) are incapable of handling zip files that contain more than 65,535 files. The solution for this problem is to either upgrade to a version of Java later than v1.6 (b25), or ensure that the archive does not exceed the threshold (contains less than 65,535 files). The FishEye cache (not included in backups by default) can be a contributor of many small files. Hence, exclude the cache from backups if this is likely to be a concern.

Archives Larger Than 4GB

Java has trouble reading and writing zip files that are larger than 4GB. As of release 1.5 Java appears capable of reliably creating archives that are over 4GB, but remains unable to extract them. For details see Sun's bug report. Also be aware of the fact that some file systems (including FAT32) have trouble with files larger than 4GB.

As a workaround, make sure you do not create archives that are larger than 4GB. The FishEye cache (not included in backups by default) can be a contributor of a lot of small files (although these tend to compress very well). If you still want to archive everything and end up with an archive that is too large, consider creating separate backups for the FishEye cache and uploaded files respectively.

Scheduling Crucible Backups

To set a schedule for automatic backups, open the administration screen and click 'Backup' under 'System' on the left navigation bar. The 'Backup' page opens. Now, click the link 'Manage Scheduled Backups' at the bottom of the page. The 'Scheduled Backups' page opens.

On the 'Scheduled Backups' page, click 'Edit' to adjust the backup schedule. Set the desired options and click 'Save'.

The options for scheduled backups are detailed in the table below.

<table>
<thead>
<tr>
<th>Option name</th>
<th>Description</th>
<th>Allowed Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable Scheduled Backups</td>
<td>Stops regular backups from taking place.</td>
<td>On (disabled) or Off (enabled)</td>
</tr>
<tr>
<td>Backup path</td>
<td>The path where the backup .zip file will be stored.</td>
<td>Any system or network path that FishEye or Crucible can access.</td>
</tr>
<tr>
<td>Backup file prefix</td>
<td>Characters that will be added to the beginning of the backup file name.</td>
<td>Any string of characters that can be used as part of a filename on the local operating system.</td>
</tr>
<tr>
<td>Backup file date pattern</td>
<td>Sets a date for the next (or initial) backup to take place.</td>
<td>Any valid date in the format yyyy_MM_dd (year, month, day of the month).</td>
</tr>
</tbody>
</table>
Backup frequency | Sets how often the backup will take place. | Can be set to 'every day', 'every Sunday', 'Monday to Friday' and 'first day of the month'.
---|---|---
Backup time (HH:mm) | The time when the backup will take place. | Any valid 24-hour time in the format HH:mm (hours, minutes).
Include | Specifies which items must be included in the backups (these components are explained at the top of this page). | As per the options for regular on-demand backup (These components are explained at the top of this page).

Be aware that scheduled backups can fill up disks unless you regularly move or delete old archives.

Restoring Crucible Data

The Crucible Data Command Line Restoration Process

There is currently no way to restore a backup from the web interface because Crucible must be shut down during a data restore.

Restoring a backup will irreversibly overwrite the data of your installation with the data from the backup archive.

If you made a backup from production which connected to an external database, and restore this backup to a test server without specifying another database to restore too, you will drop and restore to your production database. Thus when restoring to a test server, always ensure you specify the correct database to restore to (or restore to an in-built database).

1. Install Crucible into a new, empty directory (this must be the same version that the backup was created from, or later).
   - Note that you cannot restore data into versions of Crucible which are older than the version that created the backup.
2. Make sure the Crucible instance is not running.
3. Open a command line interface on the Crucible server computer.
4. Run the restore command on the command line with the desired options.
5. The specified elements will be restored.
6. Start the Crucible instance.
7. When using FishEye integrated with Crucible, you will need to re-index your repositories after restoring data, unless the backup archive was created with the --cache option.

Restore Command Line Options

These examples are for use in a Linux-like operating system. When using these commands on Windows, use the filename fisheyectl.bat.
and use the correct slashes. Run the command from the FISHEYE_HOME/bin/ directory.

The basic syntax of the restore command is as follows:

```
$ ./fisheyectl.sh restore -f /path/to/backup_2009-10-02_1138.zip [OPTIONS]
```

To see inline help for all backup options, run the following command in the FISHEYE_HOME/bin/ directory:

```
$ ./fisheyectl.sh restore --help
```

Restores a FishEye/Crucible backup instance.
If you are using an external database (as opposed to the default built-in database), make sure the JDBC driver file is present in the FISHEYE_INST/lib directory when running restore.

<table>
<thead>
<tr>
<th>Option</th>
<th>Switch</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suppress output</td>
<td>--quiet OR -q</td>
<td>Suppress the output messages from the restore program on the command line.</td>
<td>No</td>
</tr>
<tr>
<td>Choose file to restore from</td>
<td>--file PATH/FILENAME OR -f PATH/FILENAME</td>
<td>Restore the backup from PATH/FILENAME.</td>
<td>Yes (required)</td>
</tr>
<tr>
<td>Show inline help</td>
<td>--help OR -h</td>
<td>Displays help for options on the command line.</td>
<td>No</td>
</tr>
</tbody>
</table>

**Advanced Command Line Restore Settings**

By default, the restore program will restore all items found in the backup archive (so if you included the caches using the `--cache` option, these will automatically be restored). However, it is possible to only restore a subset of items from the backup, by explicitly specifying the item names on the command line and only those will be restored.

<table>
<thead>
<tr>
<th>Option</th>
<th>Switch</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restore FishEye cache</td>
<td>--cache</td>
<td>Restore the repository cache backup.</td>
</tr>
<tr>
<td>Restore plugins</td>
<td>--plugins</td>
<td>Restore 3rd-party plugins and their configuration data.</td>
</tr>
<tr>
<td>Restore templates</td>
<td>--templates</td>
<td>Restore freemarker templates from the backup (the restored instance will use the built-in templates).</td>
</tr>
<tr>
<td>Restore uploads</td>
<td>--uploads</td>
<td>Restore uploads (e.g. patch files uploaded into Crucible and contents of files under review).</td>
</tr>
<tr>
<td>Restore Crucible reviews</td>
<td>--sql</td>
<td>Restore the SQL database containing user profiles, reviews and review comments.</td>
</tr>
<tr>
<td>Set database type</td>
<td>--dbtype OR -t</td>
<td>SQL database type ('mysql', 'postgresql' or 'built-in'). Only required when restoring to a database location different to that used at backup time.</td>
</tr>
<tr>
<td>Set JDBC URL</td>
<td>--jdbcurl OR -j</td>
<td>JDBC URL of the SQL database. Only required when restoring to a database location different to that used at backup time (not applicable for 'built-in').</td>
</tr>
<tr>
<td>Set JDBC username</td>
<td>--username OR -u</td>
<td>JDBC username of the SQL database. Only required when restoring to a database location different to that used at backup time (not applicable for 'built-in').</td>
</tr>
<tr>
<td>JDBC password</td>
<td>--password OR -p</td>
<td>JDBC password of the SQL database. Only required when restoring to a database location different to that used at backup time (not applicable for 'built-in').</td>
</tr>
<tr>
<td>JDBC class</td>
<td>--driver OR -d</td>
<td>Specifies the JDBC driver class name needed to access the SQL database. Only required when restoring to a database location different to that used at backup time and when using a different JDBC driver than the standard driver associated with the database specified through <code>--dbtype</code>. (Not applicable for 'built-in').</td>
</tr>
</tbody>
</table>

**Notes on Migrating Backup Data**

When the process restores a SQL database, it looks at the configuration data (config.xml) included in the backup archive to learn which database product was used and how to connect to it. When Crucible uses the built-in HSQLDB database (which is the default), the restored
instance will also use that. However, when the restored instance will use a different database than the backed up instance (for instance, HSQLDB was used at the time the backup was created, but it needs to be restored on MySQL), use the command line options to point the process to the new database.

**Command Line Example: Migrating Backup Data to MySQL**

These examples are for use in a Linux-like operating system. When using these commands on Windows, use the filename `fisheyectl.bat` and use the correct slashes. Run the command from the `FISHEYE_HOME/bin/` directory.

Restoring to a Crucible instance that uses a different database (ensure the mysql driver jar file is present in the `FISHEYE_INST/lib` directory)

```
$ ./fisheyectl.sh restore 
--username john 
--password smith 
--jdbcurl jdbc:mysql://localhost:3306/crucible 
--dbtype mysql 
--file /path/to/backup_2009-10-02_1138.zip
```

Creating a Permission Scheme

This page contains information on how to create a permission scheme in Crucible.

**On this page:**

- Introduction to Crucible Permissions
- Creating a Permission Scheme
- Editing a Permission Scheme
- List of Crucible Permissions
- Further Reading

**Introduction to Crucible Permissions**

A *permission* is the ability to perform a particular action in Crucible, e.g. 'Create Review'.

A *permission scheme* assigns particular *permissions* to any or all of the following:

- Particular Users.
- Particular Groups.
- All logged-in users.
- Anonymous Users
- People in particular Review Roles, such as:
  - 'Author';
  - 'Reviewer';
  - 'Creator';
  - 'Moderator'.

The scheme's permissions will apply to all reviews belonging to the *project(s)* with which the scheme is associated.

You can create as many permission schemes as you wish. Each permission scheme can be associated with many projects or just one project, allowing you to tailor appropriate permissions for individual projects as required.

**Creating a Permission Scheme**

To create a permission scheme,
1. From the 'Admin Menu', click 'Permission Schemes'.
2. The 'Permission Schemes' page will be displayed, showing a list of existing permission schemes. Click the 'Create a New Permission Scheme' link, which appears below the list.
3. In the 'Name' field, type a short phrase to uniquely identify your project (see screenshot 1 below).
4. Click the 'Create' button to create your new permission scheme. The 'Edit Permission Scheme' page will be displayed for your new permission scheme (see screenshot two, below).
   - Your new permission scheme will have the default assignees shown in the table above.
5. To edit the assignees for a permission, click the 'Edit' link corresponding to the permission. The 'Edit Permission Scheme' page will be displayed.
6. Choose the appropriate assignee(s) for this permission:
   - Note: for ongoing ease of management, it is recommended that you grant permissions to groups or participants rather than to individual users.
   - To assign this permission to anonymous users, select the 'Allow Anonymous users' check-box.
   - To assign this permission to all logged-in users, select the 'Allow All logged in users' check-box.
   - To assign this permission to a particular user, type their username into the 'Individual users' field (hint: you can type just part of the name, then press <Enter> to select from a list of matching usernames).
   - To assign this permission to a particular group of users, type the group name into the 'Groups' field (hint: you can type just part of the group name, then press <Enter> to select from a list of matching groups).
   - To assign this permission to users who belong to a particular participant ('Reviewer' / 'Moderator' / 'Author' / 'Creator'), select the corresponding check-box.
7. Click the 'Save' button.

Next step: see Associating a Permission Scheme with a Project.

Screenshot 1: Adding a Permission Scheme

<table>
<thead>
<tr>
<th>Name</th>
<th>Projects using this scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>Default Project</td>
</tr>
<tr>
<td>Name:</td>
<td>Top Secret</td>
</tr>
</tbody>
</table>

Editing a Permission Scheme

To edit a permission scheme,

1. From the 'Admin Menu', click 'Permission Schemes'.
2. Click 'edit' next to the scheme you wish to change. The 'Edit Permission Scheme' page will be displayed.
3. On the 'Edit Permission Scheme' page, you can change the groups or users that are allowed individual permissions by clicking 'edit' next to the permission in question.
4. When you have finished editing, click the 'Save' button.

Screenshot: Edit a Permission Scheme
List of Crucible Permissions

The following permissions are available:

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
<th>Default Assignees</th>
</tr>
</thead>
<tbody>
<tr>
<td>'Edit'</td>
<td>Ability to edit a review's details and change the set of revisions being reviewed.</td>
<td>'Creator' 'Moderator'</td>
</tr>
<tr>
<td>Permission</td>
<td>Description</td>
<td>Users with this permission</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>'View'</td>
<td>Ability to view a review. (People without this permission will not know that the review exists.)</td>
<td>Anonymous users, All logged-in users, 'Creator', 'Author', 'Reviewer', 'Moderator'</td>
</tr>
<tr>
<td>'Abandon'</td>
<td>Ability to abandon (i.e. cancel) a review.</td>
<td>'Moderator', 'Creator'</td>
</tr>
<tr>
<td>'Re-Open'</td>
<td>Ability to re-open a closed or abandoned review.</td>
<td>'Creator', 'Moderator'</td>
</tr>
<tr>
<td>'Uncomplete'</td>
<td>Ability of a reviewer to change their individual review status from 'Complete' to 'Uncomplete'.</td>
<td>'Reviewer'</td>
</tr>
<tr>
<td>'Reject'</td>
<td>Ability to reject a review submitted for approval (i.e. prevent it from being issued to reviewers).</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>'Complete'</td>
<td>Ability of a reviewer to change their individual review status to 'Complete'.</td>
<td>'Reviewer'</td>
</tr>
<tr>
<td>'Comment'</td>
<td>Ability to add or remove a comment to or from a review.</td>
<td>'Creator', 'Author', 'Reviewer', 'Moderator'</td>
</tr>
<tr>
<td>'Approve'</td>
<td>Ability to approve a review (i.e. issue it to the reviewers).</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>'Submit'</td>
<td>Ability to submit a review for approval (i.e. request that the review be issued to the reviewers).</td>
<td>'Creator', 'Author'</td>
</tr>
<tr>
<td>'Close'</td>
<td>Ability to close a review once it has been summarised.</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>'Delete'</td>
<td>Ability to delete a review.</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>'Summarise'</td>
<td>Ability to summarise a review. (Normally this would be done after all reviewers have completed their review.)</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>'Create'</td>
<td>Ability to create a review.</td>
<td>All logged-in users, 'Creator', 'Moderator'</td>
</tr>
<tr>
<td>'Recover'</td>
<td>Ability to resurrect an abandoned (i.e. cancelled) review.</td>
<td>'Creator', 'Moderator'</td>
</tr>
</tbody>
</table>

Further Reading

For more information on permissions schemes in Crucible, see the following documentation pages:

- Agile Permissions Schemes in Crucible
- Associating a Permission Scheme with a Project

**Agile Permissions Schemes in Crucible**

This page contains information about using and editing Agile permission schemes in Crucible.

**Understanding the Agile Permissions Scheme**

Agile development teams may not want to use the default Crucible permission schemes that require one person to approve or summarise reviews. Crucible ships with a pre-defined Agile permission scheme. By Agile, we mean permission schemes that have no moderator and very liberal permissions, suited to Agile or self-organising teams.

To use the Agile permissions scheme when creating a project, simply select 'agile' from the drop down list under 'Project Permissions Scheme' on the 'Edit Project' screen.

**Considerations**

⚠️ If you began your installation of Crucible with Crucible 2.1 or later, then this permission scheme will appear in the list of permission schemes in the administration menu.

⚠️ If you have upgraded from an earlier version of Crucible (pre Crucible 2.0), then the Agile permission may not appear by default. However, you can still create the equivalent by disabling the moderator when creating projects, allowing freer access to summarising, closing and generally
associating to Crucible reviews.

If you disable the moderator role on the Edit Project screen, then Crucible will check the current permission scheme. If the current permission scheme requires a moderator, a warning will be shown and you will be prompted to create a new permission scheme which will be called 'Agile' (or Agile-X if the name Agile already exists, where X is a number appended to the scheme name). The new permissions scheme will not require a moderator to carry out any actions.

Personally: Warnings on Permission Schemes that Require a Moderator

Associating a Permission Scheme with a Project

This page explains how to associate a permission scheme with a Crucible project and show details of the default permission schemes included with Crucible.

On this page:

- Associating a Permission Scheme with a Crucible Project
- Overview of the Permission Schemes Bundled with Crucible
  - Default Permission Scheme Settings
  - Agile Permission Scheme Settings
- Related Links

Associating a Permission Scheme with a Crucible Project

To associate a permission scheme with a project,

1. From the 'Admin Menu', click 'Project List'.
2. The 'Projects List' page will be displayed. Find the project you wish to associate with your permission scheme, and click its 'Edit' link.
3. The 'Edit Project' page will be displayed.
4. Under the heading 'Project Permissions Scheme', click the 'Permission Scheme' drop-down list to select your permission scheme.
5. You will be shown a list of the schemes that have been created in Crucible. You can create a new permission scheme if necessary.
6. Click the 'Save' button.

Overview of the Permission Schemes Bundled with Crucible

Crucible comes with two permission schemes, 'Default' and 'Agile'. The following tables show the default settings in detail; note that these can be easily edited by admin users to suit your needs.

Default Permission Scheme Settings

This table shows the various permissions and which user groups have them by default.
The default permission scheme has changed since Crucible 1.6.

**Agile Permission Scheme Settings**

This table shows the various permissions and which user groups have them by default.

<table>
<thead>
<tr>
<th>Permission</th>
<th>Anonymous</th>
<th>All Logged In</th>
<th>Individuals</th>
<th>Groups</th>
<th>Review Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abandon</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Author, Moderator</td>
</tr>
<tr>
<td>Approve</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Author, Moderator</td>
</tr>
<tr>
<td>Close</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Reviewer, Creator, Author, Moderator</td>
</tr>
<tr>
<td>Comment</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Reviewer, Creator, Author, Moderator</td>
</tr>
<tr>
<td>Complete</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Reviewer</td>
</tr>
<tr>
<td>Create</td>
<td>false</td>
<td>true</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Author, Moderator</td>
</tr>
<tr>
<td>Edit Review Details</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Moderator</td>
</tr>
<tr>
<td>Recover</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Moderator</td>
</tr>
<tr>
<td>Reject</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Author, Moderator</td>
</tr>
<tr>
<td>Re-Open</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Moderator</td>
</tr>
<tr>
<td>Submit</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Author, Moderator</td>
</tr>
<tr>
<td>Summarize</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Creator, Author, Moderator</td>
</tr>
<tr>
<td>Uncomplete</td>
<td>false</td>
<td>false</td>
<td>None</td>
<td>None</td>
<td>Reviewer</td>
</tr>
<tr>
<td>View</td>
<td>false</td>
<td>true</td>
<td>None</td>
<td>None</td>
<td>Reviewer, Creator, Author, Moderator</td>
</tr>
</tbody>
</table>

**Related Links**

- Creating a Permission Scheme

**Crucible and FishEye**

This page gives an overview of the joint installation of Crucible and FishEye. Both Crucible and FishEye are Atlassian products.
FishEye allows you to extract information from your source code repository and display it in sophisticated reports. Crucible allows you to request, perform and manage code reviews. Both of these products can run in isolation. However if you are using CVS, Subversion or Perforce you can significantly enhance your Crucible experience by also using FishEye.

Your Crucible installation package includes the files required for FishEye. If you use FishEye and Crucible together, they run as one instance.

Purchasing and Installing Crucible/FishEye

- If you install Crucible, there is no need to do a separate installation of FishEye.
- Upgrading Crucible to also use FishEye requires only a simple licence change in the admin screens.
- When upgrading to Crucible when you have an existing FishEye installation, you can either keep the original FishEye installation or install Crucible and FishEye as a fresh install. Refer to the guide on upgrading from FishEye to Crucible.

FISHEYE_HOME and FISHEYE_INST

Throughout the Crucible documentation, references are made to FISHEYE_HOME, which refers to the location of the FishEye application. Because most Crucible users also run FishEye, we use a single value for the sake of simplicity.

Crucible also makes use of this FishEye environment variable:

- FISHEYE_INST – the location of the FishEye data.

Refer to the FishEye documentation for more about the environment variables and how they are used in the FishEye installation.

Detailed Documentation

You can find more information in:

- Crucible Installation Guide
- FishEye Installation Guide

Customising Email Notifications

Email notifications in Crucible can be customised to change their formatting, by editing template files. This page contains instructions for this process.

Editing Crucible Email Templates

Template files for Crucible are stored in the FISHEYE_HOME/templates/ folder.

For Crucible, the set of templates is for plain-text email only. Note that these templates do not support embedding full diffs into notifications. They are only for changing the appearance and order of certain content inside the messages.

If you edit the templates of an operational Crucible instance, you may disrupt notifications that are being sent at that time. To avoid this, shut Crucible down during template editing.

Editing the Subject Line

1. Open the 'crucible-notification-subject.ftl' template file from FISHEYE_HOME/templates/ in a text editor.
2. Type in your new text for the email subject, ensuring that all of the content is contained within line 1 of the template. 'crucible-notification-subject.ftl' is used as the subject template for all Crucible email notifications.
3. Save and close the file.
4. Restarting Crucible will activate the new templates.

Editing the Header

Header information will be included at the beginning of the email body text.

1. Open the 'crucible-notification-header.ftl' template file from FISHEYE_HOME/templates/ in a text editor.
2. Add your new header content. 'crucible-notification-header.ftl' is used as the header template for all Crucible email notifications.
3. Save and close the file.
4. Restarting Crucible will activate the new templates.
**Editing the Footer**

Footer information will be included at the end of the email body text.

1. Open the `crucible-notification-footer.ftl` template file from FISHEYE_HOME/templates/ in a text editor. `crucible-notification-footer.ftl` is used as the footer template for all Crucible email notifications.
2. Add your new footer content.
3. Save and close the file.
4. Restarting Crucible will activate the new templates.

After an edit, the change to the email template will take place immediately. No restart is required.

⚠️ Try and avoid editing the live template file, as Crucible may try to use it while you are editing. This could have unpredictable results. Instead, back up the template file (it's wise to keep original versions of all these files), edit a copy you have made, then overwrite the 'live' template once you have finished.

**Advanced Editing of Crucible Email Templates**

The email notification templates use the **Freemarker** format. Freemarker is a general templating engine enabling automated content.

If you are familiar with Freemarker, more advanced customisations can be made to the email notification templates. However, you make such adjustments at your own risk.

Note: In Crucible, email notifications are limited to plain-text format only.

**Crucible Email Template File List**

The following template files for Crucible notification are stored in the FISHEYE_HOME/templates/ folder.

<table>
<thead>
<tr>
<th>Template filename</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>crucible-notification-subject.ftl</td>
<td>Subject template</td>
</tr>
<tr>
<td>crucible-notification-header.ftl</td>
<td>Header template</td>
</tr>
<tr>
<td>crucible-notification-footer.ftl</td>
<td>Footer template</td>
</tr>
<tr>
<td>state-closed-notification.ftl</td>
<td>State Closed template</td>
</tr>
<tr>
<td>all-completed-notification.ftl</td>
<td>All Completed template</td>
</tr>
<tr>
<td>state-changed-notification.ftl</td>
<td>State Changed template</td>
</tr>
<tr>
<td>completed-notification.ftl</td>
<td>Completed template</td>
</tr>
<tr>
<td>general-notification.ftl</td>
<td>General notification template</td>
</tr>
<tr>
<td>uncompleted-notification.ftl</td>
<td>'Uncompleted' template</td>
</tr>
<tr>
<td>all-no-longer-completed-notification.ftl</td>
<td>All-No-Longer-Completed template</td>
</tr>
<tr>
<td>comment-notification.ftl</td>
<td>Comment template</td>
</tr>
<tr>
<td>reply-notification.ftl</td>
<td>Reply template</td>
</tr>
<tr>
<td>review-precis-plain.ftl</td>
<td>Precis template</td>
</tr>
</tbody>
</table>

See also [Customising FishEye Email Notifications](#).

**Freemarker Data Model for Email Templates**

**Customising Crucible email templates with Freemarker**

See the [Freemarker documentation](#) for instructions on Freemarker syntax. Use the templates that ship with Crucible as a guide to the properties available on each object.

Specific email types will have extra data associated with them, and this data will be available in that particular template (but not in others).
**Example**

The syntax to access the data-model, using the data model object 'link' as an example, place this code into the email at the desired position.

```java
${notification.link}
```

**Customising the Defect Classifications**

This page explains how to customise defects and their classifications in Crucible.

**On this page:**
- Defects in Crucible Comments
- Changing Classification Settings
- Default Crucible Classifications
  - Ranking
  - Classification

**Defects in Crucible Comments**

Defects are comments made by reviewers that indicate a problem in a review. Defects can be classified by rank and type, custom classifications can also be defined. The default classifications are shown in the screenshot below.

**Screenshot: The List of Defect Classifications**

![Screenshot of a list of defect classifications](image)

**Changing Classification Settings**

To change the default classifications:

1. Open the Crucible Admin screen. The 'Admin Menu' opens.
2. Click 'Customize Crucible Defect Classifications' under 'Global Settings' in the Admin Menu.

⚠️ Only Crucible Admin users have access to this screen.

Any changes made within 'Customize crucible defect classifications' will only affect reviews created after the setting is changed.

**Default Crucible Classifications**

There are two default defect classifications that are preset in Crucible; ranking and classification. These settings (and their sub-categories) can be edited or removed; other custom classifications can be added.
**Ranking**

This classification can be set to 'Major' or 'Minor', indicating the importance of the defect.

**Classification**

This setting helps to define the nature of the defect in particular detail. This classification can be set to one of the options in the following table; the meaning of these is detailed in the table below.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing</td>
<td>The defect applies to code or information that is missing (absent).</td>
</tr>
<tr>
<td>Extra (superfluous)</td>
<td>The defect applies to code or information that should be removed.</td>
</tr>
<tr>
<td>Ambiguous</td>
<td>The defect applies to code or information that is not clear or easy to understand.</td>
</tr>
<tr>
<td>Inconsistent</td>
<td>The defect applies to code or information that is applied in several different ways.</td>
</tr>
<tr>
<td>Improvement desirable</td>
<td>The defect applies to code or information that needs to be revised.</td>
</tr>
<tr>
<td>Not conforming to standards</td>
<td>The defect applies to code or information that breaks established conventions.</td>
</tr>
<tr>
<td>Risk-prone</td>
<td>The defect applies to code or information that takes unacceptable risks.</td>
</tr>
<tr>
<td>Factually incorrect</td>
<td>The defect applies to code or information that is wrong.</td>
</tr>
<tr>
<td>Not implementable</td>
<td>The defect applies to code or information that may be impossible to create.</td>
</tr>
<tr>
<td>Editorial</td>
<td>The defect applies to code or information where the classification as a defect may be subject to personal opinion.</td>
</tr>
</tbody>
</table>

*Screenshot: Editing Defect Classifications in Crucible*
Customising the Welcome Message

To customise the welcome message which is shown when Crucible opens, access the administration page and click ‘Customize Front Page’ under ‘Global Settings’ on the left navigation bar.

The ‘Customize Front Page Messages’ page opens.

On this page, you can provide your own custom text for the Crucible welcome message that is displayed to users when they first log in. You can also provide custom Support text, providing the contact details of your own support organisation, which also appears on the opening page.

You can enter text into the boxes provided for either message and click the small ‘Save Welcome Message’ or ‘Save Support Message’ button to save it, or enter text for both messages and click ‘Save All’. The changes are made immediately.

Screenshot: Crucible Customize Welcome and Support Messages
Using HTML

The content in the welcome screen can be arranged using basic HTML tables, image references or anchor tags such as the following:

```
<a href="http://www.atlassian.com">Link to Atlassian Home Page</a>
```

Restoring the default messages

To revert to the default Welcome or Support messages, simply delete all text shown in the text box and click the corresponding "Save" button.

Manually editing the opening screen

You can also directly edit the XML file that contains the welcome and support messages. This file is called `config.xml`, located in your installation folder.

To do this, simply add the following XML tags to `config.xml`:

```
<content>
  <front-page-message>Example welcome message here</front-page-message>
  <support-message>Example support message here</support-message>
</content>
```

JIRA Integration in Crucible

Atlassian's JIRA is an issue tracking application, which can be used to manage projects and associated work. JIRA integration with Crucible can be set up by using the bundled Unified Application Links plugin.

On this page:

- Before You Begin
- Integrating your JIRA Server with your Crucible Server

Before You Begin

Ensure that you configure your JIRA instance to enable sub-tasks and allow Remote API access. We also recommend that you enable unassigned issues.
Integrating your JIRA Server with your Crucible Server

To integrate your JIRA server with your Crucible server, you will need to do the following:

1. Create an application link between JIRA and Crucible. See Adding an Application Link.
2. Configure authentication for the new application link. See Configuring Authentication for an Application Link. Note that if you are using Trusted Applications authentication you must add /secure and /browse to the list of ‘URL Paths to Allow’ in the JIRA Trusted Application Details configuration.
3. Create entity links for the mappings between your Crucible repositories and JIRA projects, as desired. See Adding an Entity Link.

The Application Links Quick Start Guide provides an example of setting up an application link.

Obtaining Subtask Values for Crucible Configuration

Obtaining the Subtask ID Values

This value is required (along with the Subtask Resolution ID and Subtask Resolution Action ID) to enable creating issues from a Crucible comment. This is the subtask type that will be created when you create a JIRA subtask in Crucible.

To set this up in Crucible, carry out the following steps.

1. Enable sub-tasks on your JIRA instance. See the JIRA documentation.
2. Return to the Crucible Administration screen and then click ‘Application Links’ under the ‘Global Settings’ sub-menu on the left navigation bar.
3. Click ‘JIRA settings’ next to the application link to the desired JIRA server. The ‘Update JIRA Server’ screen will be displayed.
4. Click ‘Test’. The field for ‘Subtask Type ID’ will change to a drop-down menu, showing the available subtask types. Choose the correct one. The field for ‘Subtask Resolution’ will also turn into a drop-down menu. Select the desired item from this menu as well.
5. Click ‘Save’ to save your Crucible configuration settings.

6. Open your JIRA instance and go to ‘Administration’ > ‘Workflows’. The ‘Workflows’ screen opens. By default, the ‘JIRA’ workflow is shown on screen in a table.
7. Click the ‘Steps’ link in the far right table cell. The ‘View Workflow Steps — JIRA’ page opens.
8. The ‘Subtask Resolution Action ID’ is in the ‘Open’ row, under the ‘Transitions’ column. Look at the link in that cell named ‘Resolve Issue’. The ID number is shown in brackets next to that heading ‘Resolve Issue’ (shown in the screenshot below as 5).
9. Enter the number into the field in Crucible.
10. Save your Crucible configuration settings.
11. Your Crucible JIRA integration should now be complete.

If you decide to use Trusted Applications for authentication with your JIRA server, activity streams and subtasks created from review defects will be generated using the currently logged in user. However JIRA project mapping and issue key linking (including the associated 'hovering' content) will be retrieved using the user specified on the 'Update JIRA Server' configuration page shown above.

**Related Topics**

- The JIRA documentation on Integrating JIRA with FishEye, which enables you to view Crucible data from within JIRA.
- The FishEye documentation on JIRA Integration in FishEye, which enables you to view JIRA data from within FishEye.

**Migrating to an External Database**

This page contains information about migrating Crucible from its default embedded HSQL database to an external database. Advantages of using a database other than the embedded HSQL database include:

- **Improved Protection Against Data Loss**: The Crucible built-in database, running HSQLDB is somewhat susceptible to data loss during system crashes. External databases are generally more resistant to data loss during a system crash.
- **Performance & Scalability**: if you have many users on your Crucible instance, running the database on the same server as FishEye may slow it down. When using the embedded database, the database will always be hosted and run on the same server as Crucible.
- **Data Stored in the Crucible Database**: The Crucible database stores all information besides the cache for repository scans. This means all reviews, comments, review states, user data and user preferences information.

**On this page:**

- Supported Databases
- Support for Other Databases

**Supported Databases**
Crucible and FishEye offer alternatives to the built-in HSQLDB database for storing its relational data. At the time of writing, MySQL Enterprise Server and PostgreSQL are supported (see Supported Platforms for version numbers). Please see the following pages for instructions on migrating to the relevant database:

- Migrating to MySQL Enterprise Server
- Migrating to PostgreSQL

Support for Other Databases

Crucible and FishEye currently ship with support for MySQL Enterprise Server and PostgreSQL as external databases (see Supported Platforms for version numbers).

If you are looking for support for Oracle or Microsoft SQL Server, please vote for the issues below. Your vote will help us prioritise them.

- Request Oracle Support: CRUC-1489
- Request MS-SQL Support: CRUC-1407

If you are using another database product that you would like to see supported, please create a JIRA issue for it under the Crucible project.

Migrating to MySQL Enterprise Server

This page contains instruction on how to migrate Crucible from the built-in HSQLDB database to MySQL Enterprise Server.

On this page:

1. Install MySQL
2. Install the JDBC Drivers
3. Create a UTF-8 Database
4. Set the Server Characterset to UTF-8
5. Verify Database Character Encoding Information
6. Create a Database User
7. Configure the Database for FishEye/Crucible
8. Start the Migration Process

Notes

1. Install MySQL

Install MySQL Enterprise Server and follow the steps below.

- Please ensure you install a version of MySQL that is supported. Check the Supported Platform Matrix for exact versions.

2. Install the JDBC Drivers

The JDBC drivers for MySQL Enterprise Server are bundled with FishEye. Skip to step 3 if this meets your needs. If you want to install a specific, different version of the bundled JDBC driver, download the MySQL Enterprise Server JDBC driver JAR file from the download website and copy the JAR to your FISHEYE_INST/lib directory (create the lib/ directory if it doesn’t already exist). Move the existing JDBC JAR file to another location (and back it up). Restart FishEye or Crucible to have it pick up the new driver.

3. Create a UTF-8 Database

Run the commands below to create a UTF-8 database. Note that the DB must be case-sensitive, i.e. it must use the utf8_bin collation sequence.

```
CREATE DATABASE crucible CHARACTER SET utf8 COLLATE utf8_bin;
```

4. Set the Server Characterset to UTF-8

You will need to set the Server Characterset to utf8. This can be done by adding the following in my.ini for Windows or my.cnf for other OS. It has to be declared in the Server section, which is the section after [mysqld]:

```
[mysqld]
default-character-set=utf8
```
Also set the value here:

```
[mysql]
default-character-set=utf8
```

5. Verify Database Character Encoding Information

Use the `status` command to verify database character encoding information:

**Screenshot: Using the MySQL Enterprise Server Status Command**

```sql
mysql> status;
```

```
+------------------+
| Server characterset | utf8 |
| Db characterset     | latin1 |
| Conn. characterset  | latin1 |
+------------------+
```

6. Create a Database User

Create a user that can log in from the host that Crucible or FishEye is running on and make sure that the user has full access to the newly created database. In particular, the user should be allowed to create and drop tables, indexes and other constraints.

For instance, when Crucible and MySQL Enterprise Server run on the same machine (accessible through `localhost`), issue the following commands (replacing `username` and `password` with the appropriate values):

```sql
mysql> grant all on crucible.* to 'username'@'localhost' identified by 'password';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)
```

7. Configure the Database for FishEye/Crucible

With the database prepared, navigate to the 'Database Configuration' section in the admin interface, select MySQL Enterprise Server from the drop down and fill out the database URL, username and password.

Then click 'Test Connection' to verify that Crucible or FishEye can log in to the database:

**Screenshot: Testing the Connection**
8. Start the Migration Process

Click ‘Save & Migrate Data’ to start the migration process.

During the migration process (which will take several minutes, depending on the size of your database and network throughput), the product will be inaccessible to users and external API clients. Users will see a maintenance screen that informs them of the process. Should the migration fail for any reason, Crucible will not switch to the new database and report on the encountered problems. Because the destination database may now contain some, but not yet all data, drop all tables, indexes and constraints before attempting a new migration.
Notes

Truncation of Database Fields

If you are using Crucible 1.6 or earlier with an HSQL database, or have an HSQL database which contains data created by Crucible 1.6 or earlier, truncation of some database fields may occur when migrating to MySQL. You will be notified of field truncations by messages in the migration log, displayed during the migration. The following scenarios may occur:

- **A database key is truncated:** In this case, the migration will fail. This typically occurs for the Revision ID field where ClearCase has been used with Crucible, as ClearCase can create long revision names. If you experience this problem, please contact support for assistance.
  
  **Error Log message:** "The value of column <table name.column name>, has a length of <length> which is greater than the maximum allowed length for this column of <max size of column>. As this column is part of a unique index, the migration cannot be completed."

- **A non-key field is truncated:** In this case, the migration will not fail, however the field will be truncated. This typically occurs for the Review Name field where the review has been created from a commit message. If you experience this problem, you may wish to update the field via Crucible.
  
  **Error Log message:** "Truncating value of column <table name.column name> from '<old value>' to '<value>' because its length of <length of old value> is greater than the maximum allowed length for this column of <max size of column>."

Migrating to PostgreSQL

This page contains instruction on how to migrate Crucible from the built-in HSQLDB database to PostgreSQL.

On this page:

- 1. Install PostgreSQL
- 2. Install the JDBC Drivers
- 3. Create a Database User
- 4. Create a UTF-8 Database
- 5. Set Access Permissions
- Notes

1. Install PostgreSQL

Install PostgreSQL and follow the steps below.

2. Install the JDBC Drivers

The JDBC drivers for PostgreSQL are bundled with FishEye. Skip to step 2 if this meets your needs. If you want to install a specific, different version of the bundled JDBC driver, download the Postgresql JDBC driver JAR file from the PostgreSQL website and copy the JAR to your FISHEYE_INST/lib directory (create the lib directory if it doesn't already exist). Move the existing JDBC JAR file to another location (and back it up). Restart FishEye or Crucible to have it pick up the new driver.

3. Create a Database User

Create a new database user (replacing 'username' and 'password' with the appropriate values):

```
$ psql
> create user username password 'password';
```

4. Create a UTF-8 Database

Create a UTF-8 database and make the newly created user the owner:

```
> create database crucible ENCODING 'UTF-8' OWNER username;
```

5. Set Access Permissions

Make sure the user has full access to the database:

```
> grant all on database crucible to username;
```

During the migration process (which will take several minutes, depending on the size of your database and network throughput), the product will be inaccessible to users and external API clients. Users will see a maintenance screen that informs them of the process. Should the migration fail...
for any reason, Crucible will not switch to the new database and report on the encountered problems. Because the destination database may now contain some, but not yet all data, drop all tables, indexes and constraints before attempting a new migration.

Notes

Truncation of Database Fields

If you are using Crucible 1.6 or earlier with an HSQL database, or have an HSQL database which contains data created by Crucible 1.6 or earlier, truncation of some database fields may occur when migrating to PostgreSQL. You will be notified of field truncations by messages in the migration log, displayed during the migration. The following scenarios may occur:

- **A database key is truncated:** In this case, the migration will fail. This typically occurs for the Revision ID field where ClearCase has been used with Crucible, as ClearCase can create long revision names. If you experience this problem, please contact support for assistance.
  
  *Error Log message:* "The value of column `<table name.column name>`, has a length of `<length>` which is greater than the maximum allowed length for this column of `<max size of column>`. As this column is part of a unique index, the migration cannot be completed."

- **A non-key field is truncated:** In this case, the migration will not fail, however the field will be truncated. This typically occurs for the Review Name field where the review has been created from a commit message. If you experience this problem, you may wish to update the field via Crucible.
  
  *Error Log message:* "Truncating value of column `<table name.column name>` from `<old value>` to `<value>` because its length of `<length of old value>` is greater than the maximum allowed length for this column of `<max size of column>`.

Creating a User

Some user management functions are identical between Crucible and FishEye, Crucible's sister application. See more user management documentation in the FishEye documentation.

There are two types of user accounts:

- **'Built-in' user accounts** — these are stored in the application's local database.
- **'External' user accounts** — these are stored in an external directory (e.g. LDAP), if any are configured. See Configuring External Authentication Sources.

**Note regarding external directories:**

- New users can only be added if they already exist in the external directory. Your external directory will not be modified.
- If you have enabled 'auto-add' for your external directory, users who don't exist locally will be automatically added the first time they log in.

To add a new user,

1. Click 'Users' on the 'Admin Menu'.
2. The 'User Browser' screen will be displayed (see screenshot below). Click the 'Add User' button at the bottom of the screen.
3. The 'Add new user' screen will be displayed.
4. In the 'Username' field, type the user's login name. You can use the following characters:
   - letters and numbers
   - hyphen ('-')
   - underscore ('_')
   - 'at' sign ('@')
5. In the 'Display name' field, type the user's display-name.
6. (Optional) In the 'Email' field, type the user's email address. This address is where the user will receive notifications.
7. In the 'Auth Type' field, select either 'Built-in' or the name of the appropriate external directory where the user will be stored.
8. (For built-in users only) In the 'Password' and 'Confirm Password' fields, type the user's password.
9. Click the 'Add' button.

Screenshot: User Browser
Trusted Applications

A 'trusted application' is an application that Crucible will allow to access specific functions in Crucible, on behalf of any user — without the user logging in to Crucible. You can set up trusted apps authentication between FishEye, Crucible, JIRA (3.12 and later) and Confluence (2.7 and later) servers.

Application links with trusted apps authentication (as well as other types of authentication) can be set up by using the bundled Unified Application Links plugin.

For further instructions, please see the Application Links Documentation, particularly the Application Links Quick Start Guide.

Related Topics

Application Links Documentation
JIRA Integration in Crucible

Setting up Users and Security

User management and security settings are covered in the FishEye documentation.

Other Related Security Resources

Page: Severity Levels for Security Issues
Page: Security Patch Policy
Page: Configuring User Managed Mappings
Page: Creating a Permission Scheme
Page: Setting up Users and Security
Page: Security Advisory Publishing Policy
Page: How to Report a Security Issue

Enabling Access Logging in Crucible

Stop Fisheye/Crucible then create the file FISHEYE_HOME/content/WEB-INF/jetty-web.xml with the following content:

```
<property name="accessLog" file="FISHEYE_HOME/var/log/fisheye-access.log.yyy_mm_dd"/>
```

Restart Fisheye/Crucible and that will create an access log in FISHEYE_HOME/var/log/fisheye-access.log.yyy_mm_dd format (e.g.
fisheye-access.log.2010_03_17). If you want to change the path to your FISHEYE_INST directory, change the default="./var/log" to the path to the log folder in FISHEYE_INST.

The logs are written in NCSA format:

```
```

Please refer to the Jetty documentation for more configuration options.

**Configuring User Managed Mappings**

In Crucible, Administrators can control whether users can use the Author Mapping setting to map their own Crucible usernames to repository committer accounts or not. By default, the setting allows users to set their own mappings.

If you wish to lock down the mappings for security or audit reasons, this setting lets you restrict all management of mappings to Crucible administrators only.

To do this, click ‘Administration’ in the footer of the Crucible interface and then ‘Security’ in the left navigation bar. The ‘Authentication Settings’ page opens. You can click to set User Managed Mappings ‘On’ or ‘Off’. The setting is applied immediately.

**Screenshot: User Managed Mappings**

![Screenshot: User Managed Mappings](image)

**Crucible Installation and Upgrade Guide**

- Crucible Installation Guide
  - Installing Crucible
  - Configuring Crucible
  - Configuring Repositories
  - Best Practices for Crucible Configuration
- Crucible Release Notes
  - Crucible 2.5 Release Notes
  - Crucible 2.4 Release Notes
  - Crucible 2.3 Release Notes
  - Crucible 2.2 Release Notes
  - Crucible 2.1 Release Notes
  - Crucible 2.0 Release Notes
  - Crucible 2.0 Beta Release Notes
  - Crucible 1.6 Release Notes
  - Crucible 1.5 Release Notes
  - Crucible 1.2 Release Notes
  - Crucible 1.1 Release Notes
  - Crucible Release Summary
Crucible Installation Guide

This guide explains how to get Crucible installed and running as easily as possible. Many references are made to the FishEye documentation.

This document will refer to the location where you have extracted Crucible (a directory) as /FISHEYE_HOME/.

Refer to our explanation of how Crucible works with FishEye.

Evaluating Crucible for the first time? See the Crucible 101 page.

Installing Crucible

This page contains instructions for the initial installation of Crucible.

On this page:

- Installing the Crucible Binary Files
- Setting up a Repository for use with Stand-alone Crucible
- Setting up a Repository for use with FishEye and Crucible
- Next: Configuration

Installing the Crucible Binary Files

Follow these steps to install Crucible:

1. Download the Crucible zip file and extract it. This document assumes you have extracted your Crucible zip file into a directory called /FISHEYE_HOME/.
2. Ensure you have installed an appropriate Java runtime - see System Requirements. Ensure that java is in the PATH, or that the JAVA_HOME environment variable is set.

Setting up a Repository for use with Stand-alone Crucible

When accessing repositories from stand-alone Crucible, the SCM client interface will access the repositories on demand. This requires that you configure the plugin.

For complete instructions, see Configuring Repositories.

Setting up a Repository for use with FishEye and Crucible

- If you intend to use Crucible and FishEye with Subversion, please ensure you read the System Requirements, Subversion client setup, and granting permission to FishEye to scan your repository.
- If you intend to use Crucible and FishEye with Git, please ensure you read the System Requirements and Git Client setup.
- If you intend to use Crucible and FishEye with Perforce, please ensure you read the System Requirements and Perforce Client setup.
- If you intend to use Crucible and FishEye with CVS, please ensure you read the System Requirements and CVS Client setup.

Next: Configuration

To go on with configuration, see the page configuring Crucible.

Configuring Crucible

On this page:
Running Crucible

This document assumes you have extracted your Crucible zip file into a directory called /FISHEYE_HOME/.

To run Crucible for the first time, simply do the following:

- On Windows:
  ```
  C:\> cd FISHEYE_HOME\bin
  C:\FISHEYE_HOME\bin> run.bat
  ```

- On Unix-based systems:
  ```
  $ cd /FISHEYE_HOME/bin
  $ ./run.sh
  ```

Once started, Crucible will run its own HTTP web server on port 8060. You can access Crucible immediately by going to http://HOSTNAME:8060/ in a browser.

By default, Crucible will listen on port 8060 for HTTP requests. It also listens on 127.0.0.1:8059 as a control port. You can configure both of these in the Administration screens, or by editing /FISHEYE_HOME/config.xml and restarting Crucible.

Supplying Administration Password and License Key

The first time you access the Crucible web server (http://HOSTNAME:8060/) you will see a screen like this, and here you will be asked for two things:

1. An administrator password. This password controls access to the Administration screens.
2. A license key. Please note your server ID. You can then get a Crucible evaluation license key here.

Accessing the Administration Pages

Once you have set up an administrator password (as described above), you can access the Administration screens at http://HOSTNAME:8060/admin/.

One of your first steps will be to set up access to a source-control repository, or an alternative form of code storage such as the Local File System or Atlassian Confluence.

The instructions for configuring your repositories are different, depending on your Crucible setup:

- If you are running Crucible with Atlassian's FishEye, you can manage your repositories using FishEye. See the FishEye documentation for more information.
- If you are running Crucible standalone, you still can manage your repositories using native repository access (bundled with Crucible). See the FishEye documentation for more information. Please also see the What happens if I decide to stop using FishEye with Crucible? page for important information about light FishEye.
- If you are running Crucible standalone and are using "lightSCM" plugins to connect to your repositories, like the Crucible Subversion SCM plugin, you need to follow the instructions in the pages listed below. Please also see the What happens if I decide to stop using FishEye with Crucible? page for important information about lightSCM plugins.
  - Setting Up a Git Repository in Stand-Alone Crucible
  - Setting Up a Perforce Repository in Stand-Alone Crucible
  - Setting Up a Subversion Repository in Stand-Alone Crucible

The following topics describe how to configure repositories that are not managed by FishEye. These instructions apply to all three setups described above:

- Enabling Reviews from the Server File System in Crucible
Setting Up Reviewing of Confluence Pages in Crucible

Setting Up Users

On initial setup of Crucible, there are no users. Adding user accounts is done via the Administration screens or by configuring Crucible/FishEye to use external authentication.

To add users:

1. Open the FishEye Administration screens at http://HOSTNAME:8060/admin/.
2. Click 'Users/Security' under 'Global Settings' in the 'Admin Menu'.

Read more details about the different ways of creating users.

Setting Up SMTP

Crucible can email each review participant on a range of changes. Each user can then set up their own preferences. This is described in the User Profile guide.

First, you must set up the SMTP Server.

Using Crucible

You can access Crucible immediately by going to http://HOSTNAME:8060/ in a browser

Or you can go directly into the Crucible homepage at http://HOSTNAME:8060/cru

Stopping Crucible

To stop the Crucible server:

- On Windows:
  ```
  C:\> cd FISHEYE_HOME\bin
  C:\FISHEYE_HOME\bin> stop.bat
  ```

- On Unix-based systems:
  ```
  $ cd /FISHEYE_HOME-bin
  $ ./stop.sh
  ```

Information on FishEye integration

If you want to know more about how Crucible and FishEye interact, refer to our explanation of how Crucible works with FishEye.

Configuring Repositories

The instructions for configuring your repositories are different, depending on your Crucible setup:

- If you are running Crucible with Atlassian's FishEye, you can manage your repositories using FishEye. See the FishEye documentation for more information.
- If you are running Crucible standalone, you still can manage your repositories using native repository access (bundled with Crucible). See the FishEye documentation for more information. Please also see the What happens if I decide to stop using FishEye with Crucible? page for important information about light FishEye.
- If you are running Crucible standalone and are using "lightSCM" plugins to connect to your repositories, like the Crucible Subversion SCM plugin, you need to follow the instructions in the pages listed below. Please also see the What happens if I decide to stop using FishEye with Crucible? page for important information about lightSCM plugins.
  - Setting Up a Git Repository in Stand-Alone Crucible
  - Setting Up a Perforce Repository in Stand-Alone Crucible
  - Setting Up a Subversion Repository in Stand-Alone Crucible

The following topics describe how to configure repositories that are not managed by FishEye. These instructions apply to all three setups described above:

- Enabling Reviews from the Server File System in Crucible
- Setting Up Reviewing of Confluence Pages in Crucible
Enabling Reviews from the Server File System in Crucible

To set up the File System as a Code Repository in stand-alone Crucible,

1. Start Crucible then open the 'Admin' menu.
2. Under the 'System Settings' heading, click 'Plugins' in the left-hand navigation bar.
3. The 'Plugins' screen opens.
4. Next to 'crucible-filesystem-scm-plugin', click 'Enable'.
5. New options appear next to 'File System SCM': 'Disable' and 'Configure'. Click 'Configure'.
6. The 'Configure Plugin' screen opens. Click 'Add Repository'.
7. The 'Add Repository' screen opens. Fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>What to enter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Choose a unique name for the repository.</td>
</tr>
<tr>
<td>Base Path</td>
<td>Choose the lowest level of directory that Crucible will access.</td>
</tr>
</tbody>
</table>

8. Click 'Save'. The view will return to the list of repositories.
9. The server file system is now set up as a code repository for Crucible. You will be able to select files from it when creating reviews, with the ability to browse the files and directories on the hard drive.

Setting Up a Git Repository in Stand-Alone Crucible

This page contains instructions on how to configure the Crucible Git SCM plugin to access Git repositories.

**Crucible SCM plugins superseded by Native Repository Access**

Crucible now ships with native repository access, which allows you to connect to repositories without a working version of standalone FishEye. Crucible SCM plugins will still work, but we recommend that you stop using them in favour of native repository access. See What happens if I decide to stop using FishEye with Crucible? for instructions.

On this page:

- Usage
- Installation
- Configuring the plugin
- Feedback

**Usage**

The Git plugin is an early-access implementation of a Crucible SCM plugin for Git. It allows users to perform code reviews on a local Git repository (local to the Crucible server). The plugin does not 'pull' updates from a remote master repository. Synchronising with the master repository needs to be executed manually (via the command line), for the changes to appear in the plugin.

**Installation**

Firstly, download the plugin .JAR file to your local computer.

The plugin is installed by placing the .JAR file in the FISHEYE_INST/var/plugins/user directory of your Crucible install. Once installed, you
need to enable the plugin in the Crucible Admin interface. Detailed instructions on the plugin installation steps can be found at the Managing Plugins page.

The plugin requires the Git command to be available in the system path when starting Crucible.

**Configuring the plugin**

Once the plugin has been installed, under the ‘Administration’ - ‘Repository List’ option, there should be a ‘Plugin Repository List: Git’ entry. Select ‘Configure Plugin’, then ‘Add a repository’. The fields required are:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name for the repository eg. Project</td>
</tr>
<tr>
<td>Repository Path</td>
<td>The location of the local Git repository clone</td>
</tr>
</tbody>
</table>

Once configured, the Git repository can be selected as the review source when creating a new review, whereupon reviews can be created either using changesets or by selecting files in the repository view.

**Feedback**

If you have any feedback on this plugin and its operation, we would appreciate users posting feedback in the Crucible Forums.

For more information, see the Crucible Git Plugin page.

**Setting Up a Perforce Repository in Stand-Alone Crucible**

This page contains instructions on how to configure the Crucible Perforce SCM plugin to access Perforce repositories.

⚠️ **Crucible SCM plugins superseded by Native Repository Access**

Crucible now ships with native repository access, which allows you to connect to repositories without a working version of standalone FishEye. Crucible SCM plugins will still work, but we recommend that you stop using them in favour of native repository access. See What happens if I decide to stop using FishEye with Crucible? for instructions.

On this page:

- Setting Up a Perforce Repository in Stand-Alone Crucible
- Notes

**Setting Up a Perforce Repository in Stand-Alone Crucible**

To set up Perforce in stand-alone Crucible,
1. Ensure that the Perforce executable file is on the system path, in the Crucible server's Environment Variables.
2. Start Crucible then open the 'Admin' menu by clicking the Administration link in the footer of the page.
3. Under the 'Repository Settings' heading, click 'Repository List' in the left-hand navigation bar.
4. The 'Repository List' screen opens.
5. Find the Perforce repository plugin and click its Configure Plugin link.
6. The 'Configure Plugin' screen opens. Click 'Add Repository'.
7. The 'Add Repository' screen opens. Fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>What to enter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Choose a unique name for the repository.</td>
</tr>
<tr>
<td>Repository Server</td>
<td>Enter the base URL and port for the repository, for example: example.com:666.</td>
</tr>
<tr>
<td>Repository Path</td>
<td>Add the path to your Perforce repository. For example: //depot/code/example/main.</td>
</tr>
<tr>
<td>Perforce Username</td>
<td>Enter the username of the Perforce account that Crucible will use. (optional) Note that this account should only have read-only access to the repository.</td>
</tr>
<tr>
<td>Perforce Password</td>
<td>Enter the password of the Perforce account that Crucible will use. (optional)</td>
</tr>
</tbody>
</table>

8. Click 'Save'. The view will return to the list of repositories.
9. Your Perforce repository is now set up for Crucible. You will be able to select changesets from it when creating reviews.

**Notes**

- There is no 'initial scanning' required in this process, as stand-alone Crucible's access to Perforce is strictly on-demand. Data is not indexed, hence there is no scanning.
- Crucible executes the Perforce command-line tool to enable this functionality.

**Setting Up a Repository via FishEye**

**Setting Up a Repository in Crucible via FishEye**

To use FishEye to access the source control repositories CVS, Subversion or Perforce for Crucible, see the FishEye documentation for how to add a repository.

- ClearCase
- CVS
- Git
- Mercurial
- Perforce
- Subversion
Building index and cache
FishEye needs to build an index and cache of the contents of your repository, so some information will not appear in FishEye until this is complete. This may take some time to complete, depending on the size of the repositories.

We recommend you access the repository with a user that has only read access to the repository.

Setting Up a Subversion Repository in Stand-Alone Crucible
This page contains instructions on how to configure the Crucible Subversion SCM plugin to access Subversion repositories.

Crucible SCM plugins superseded by Native Repository Access
Crucible now ships with native repository access, which allows you to connect to repositories without a working version of standalone FishEye. Crucible SCM plugins will still work, but we recommend that you stop using them in favour of native repository access. See What happens if I decide to stop using FishEye with Crucible? for instructions.

On this page:
- Setting Up a Subversion Repository in Stand-Alone Crucible
- Finding your Repository Root.

Setting Up a Subversion Repository in Stand-Alone Crucible
To set up Subversion in stand-alone Crucible,
1. Start Crucible then open the 'Admin' menu by clicking the Administration link in the footer of the page.
2. Under the 'Repository Settings' heading, click 'Repository List' in the left-hand navigation bar.
3. The 'Repository List' screen opens.
4. Find the SVN repository plugin and click its Configure Plugin link.
5. The 'Configure Plugin' screen opens. Click 'Add Repository'.
6. The 'Add Repository' screen opens. Fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>What to enter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Choose a unique name for the repository.</td>
</tr>
<tr>
<td>Repository Root</td>
<td>Enter the repository root URL for the repository. If you are not sure what the repository root is, please see the instructions below under &quot;Finding your Repository Root&quot;.</td>
</tr>
<tr>
<td>Repository Path</td>
<td>Add the path on the base URL where your repository. For example, if you used the root URL above, and the full path to your Subversion instance is &quot;<a href="http://svn.example.com/svn5">http://svn.example.com/svn5</a>&quot;, you would enter 'svn5' into this field.</td>
</tr>
<tr>
<td>SVN Username</td>
<td>Enter the username of the Subversion account that Crucible will use.</td>
</tr>
<tr>
<td>SVN Password</td>
<td>Enter the password of the Subversion account that Crucible will use.</td>
</tr>
<tr>
<td>Save</td>
<td></td>
</tr>
</tbody>
</table>

7. Click 'Save'. The view will return to the list of repositories.
8. Your Subversion repository is now set up for Crucible. You will be able to select changesets from it when creating reviews.

There is no 'initial scanning' required in this process, as stand-alone Crucible's access to Subversion is strictly on-demand. Data is not cached, hence scanning is not required.

Finding your Repository Root.

Run the following command:

```svn info SVN_URL```

Where SVN_URL is the complete URL of the repository you want to add.

You will get something like the following:
Next to "Repository Root" is the URL you should define as your repository root. The path will be whatever is remaining.

**Setting Up Reviewing of Confluence Pages in Crucible**

You need to first install the Confluence Crucible Plugin in the Confluence instance (see instructions on installing Confluence plugins), and you must set up a trust relationship so that your Crucible instance trusts your Confluence instance.

**To set up Confluence as a Code Repository in Crucible,**

1. Start Crucible then open the 'Admin' menu by clicking the Administration link in the footer of the page.
2. Under the 'Repository Settings' heading, click 'Repository List' in the left-hand navigation bar.
3. The 'Repository List' screen opens.
4. Find the Confluence repository plugin and click its Configure Plugin link.
5. The 'Configure Plugin' screen opens. Click 'Add Repository'.
6. The 'Add Repository' screen opens. Fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>What to enter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Choose a unique name for the repository.</td>
</tr>
<tr>
<td>URL</td>
<td>Enter the URL of your Confluence instance.</td>
</tr>
<tr>
<td>Space Key</td>
<td>You may optionally enter a space key here to restrict Crucible's view to that key only. If there are many spaces in your Confluence instance you will find that this improves performance. You can set up several Confluence repositories in Crucible, each using the same Confluence instance but covering a different Space.</td>
</tr>
</tbody>
</table>

7. Click 'Save'. The view will return to the list of repositories.
8. Now, access your Confluence instance. Open the 'Confluence Administration Console', then select 'Trusted Applications'. The Confluence 'Trusted Applications Details' dialog opens.
9. In the 'Trusted Applications Details' dialog, enter the URL of your Crucible instance into the 'Name' field and click 'Send Request'. The 'Application Alias' will be automatically retrieved from Crucible. Save your changes.
10. Confluence is now set up as a code repository for Crucible. You will be able to select your Confluence from the list of repositories, then select files from the Confluence wiki and add them to reviews.

**Best Practices for Crucible Configuration**

1. Set up a separate FISHEYE_INST folder location on the same system for Crucible's data.
This will allow for easy upgrades of the core program and neatly separated data backup.

2. Run Crucible on a dedicated machine, accessing its data on the local file system.

This is the best environment for swift Crucible performance. Avoid running Crucible in a virtual environment.

3. Do not give Crucible projects the same key as your JIRA projects.

When naming projects, take care to ensure that the key you assign to them is not the same as any of your JIRA projects. The reason for this is, if one of your Crucible projects has the same key as one of your projects in JIRA, then all links with that key will lead back to Crucible, rather than leading to JIRA, removing the ability to navigate between the two applications.

To avoid this, name your Crucible project keys differently. For example, you could place the following text at the beginning of each project key: CR– to distinguish it. So, for this case, if you have an existing JIRA key of 'RHUBARB', you would create a Crucible key called 'CR–RHUBARB' so that they do not conflict.

4. Do not use the built-in HSQLDB database for production use.

The Crucible built-in database, running HSQLDB is somewhat susceptible to data loss during system crashes. We recommend that you do not use HSQLDB for production systems. External databases are generally more resistant to data loss during a system crash and are more suited for production use.

At this time, Crucible supports the following external databases:

- MySQL Enterprise Server 5.0.21 onwards and MySQL Community Server 5.0.21 onwards (see the Crucible Database documentation).
- PostGreSQL 8.x onwards (see the Crucible Database documentation).

Crucible Release Notes

Crucible 2.5 has now been released. Read the Release Notes.

Crucible Release Notes and Changelogs

- Security Advisories
  - FishEye and Crucible Security Advisory 2011-01-12
  - Crucible Security Advisory 2010-06-16
  - Crucible Security Advisory 2010-05-04
- Crucible Release Summary
- Crucible 2.5 Release Notes
  - Crucible 2.5 Upgrade Guide
  - Crucible 2.5 Changelog
- Crucible 2.4 Release Notes
  - Crucible 2.4 Upgrade Guide
  - Crucible 2.4 Changelog
- Crucible 2.3 Release Notes
  - Crucible 2.3 Upgrade Guide
  - Crucible 2.3 Changelog
- Crucible 2.2 Release Notes
  - Crucible 2.2 Upgrade Guide
  - Crucible 2.2 Changelog
- Crucible 2.1 Release Notes
  - Crucible 2.1 Upgrade Guide
  - Crucible 2.1 Changelog
- Crucible 2.0 Release Notes
  - Crucible 2.0 Upgrade Guide
  - Crucible 2.0 Changelog
- Crucible 2.0 Beta Release Notes
  - JIRA Integration in Crucible 2.0 Beta
  - Crucible 2.0 Beta Upgrade Notes
  - Crucible 2.0 Beta Reviewer's Guide
- Crucible 1.6 Release Notes
  - Crucible 1.6 Upgrade Guide
  - Crucible 1.6 Changelog
- Crucible 1.5 Release Notes
  - Crucible 1.5 Upgrade Guide
  - Crucible 1.5 Changelog
- Crucible 1.2 Release Notes
  - Crucible 1.2 Upgrade Guide
Installation

You can download Crucible from here. Information on installing Crucible can be found here.

If upgrading from a previous version, please follow the Upgrade Guide.

- As of version 1.0, Crucible now requires a JVM version 1.5 or later. Previously, 1.4+ was required.
- Crucible 1.1.4 includes FishEye 1.3.8.
- Upgrading from 1.0.4 (or earlier) will force a complete re-index of P4 repositories.

Crucible 2.5 Release Notes

8 February 2011

With great pleasure, Atlassian presents Crucible 2.5, now with support for Oracle and redesigned Activity Stream.

Highlights of this Release:

- Oracle Support (beta)
- Redesigned Activity Stream
- Improved Header
- Comment Notification Batching
- And Even More Improvements

Responding to your Feedback:

🌟 Over 150 votes satisfied

Thank you for all your issues and votes. Keep logging issues to help us keep improving!

Read the release notices for important information about this release.

Highlights of Crucible 2.5

1

Oracle Support (beta)

We have been working very hard on adding support for the Oracle DBMS for Crucible 2.5 as this has been a very popular feature request with more that 115 votes.

We have been using Crucible with Oracle over the past month ourselves, so we believe we are almost there. We would love for you to give Crucible a try on Oracle DBMS and let us know if you run into any problems by creating an issue.

2

Redesigned Activity Stream

We have spent a considerable amount of time improving the FishEye activity stream:
- Cleaner visual design
- Larger avatars
- Improved scanability
- De-cluttering of UI elements, showing actions on hover
- Space-saving condensed mode so you can see more changesets on the same page

<table>
<thead>
<tr>
<th>Activity</th>
<th>Reviews</th>
<th>Favourites</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Commits</td>
<td>Reviews</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Issues</td>
</tr>
<tr>
<td>Today</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sharkie</td>
<td>9 mins ago</td>
<td></td>
</tr>
<tr>
<td>Anna Buttfield</td>
<td>17 mins ago</td>
<td></td>
</tr>
<tr>
<td>Sharkie</td>
<td>18 mins ago</td>
<td></td>
</tr>
<tr>
<td>Anna Buttfield</td>
<td>19 mins ago</td>
<td></td>
</tr>
<tr>
<td>Michael Studman</td>
<td>5 hrs ago</td>
<td></td>
</tr>
</tbody>
</table>

**Improved Header**

Crucible 2.5 tracks locations that you have recently visited, and provides quick and easy access to navigate to these resources. The five most recently viewed projects, reviews, users and repositories (if FishEye is installed) are available from the header drop down links as shown:
Comment Notification Batching

Email notifications from Crucible can be frequent, particularly on reviews with many participants and lots of comments. We won’t say spam, but it can certainly feel “chatty!” Crucible 2.5 remedies this problem with improved batching of emails. All comments are grouped together and sent in a user-specified interval. They’re easier to read and will keep your inbox happy. It works for draft comments and edited comments, too, so fixing a typo will no longer send out multiple notifications!

Example:

There are 2 new comments in the following thread:

Geoff Crain on 02 Feb 2011, 12:01
Is this really how this is supposed to work? You should fix this.

****** New
***************************************************************************************************
Seb Ruiz on 02 Feb 2011, 13:31
Geoff is right. But you’ll have to fix it on a separate branch.

****** New
***********************************************************************************************
Joe Xie on 02 Feb 2011, 13:32
Fixed.. change is here: http://localhost:6060/crucible/changelog/FE?cs=e962e59bccda

And Even More Improvements

• The Universal Plugin Manager
• Improved submit time dialog to JIRA:

![Review Complete dialog]

Visit our issue tracker to see the full list of improvements and bug fixes in FishEye and Crucible for this release.

Release Notices

Crucible 2.5 Changelog

This page contains information about the Crucible 2.5 minor releases. FishEye license holders should also check the FishEye 2.5 Changelog.

⚠️ Please read the Crucible 2.5 Upgrade Guide before upgrading to any of the minor releases below.

On this page:

- From 2.5.2 to 2.5.3
- From 2.5.1 to 2.5.2
- From 2.5.0 to 2.5.1

From 2.5.2 to 2.5.3

28 March 2011

This is a bug fix release. The complete list of issues is below.

<table>
<thead>
<tr>
<th>JIRA Issues (10 issues)</th>
<th>Type</th>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
</table>
### Crucible 2.4 Documentation

| Key     | Type               | Summary                                                                 | Assignee                | Reporter                | Priority  | Status  | Resolution | Created  | Updated  |
|---------|--------------------|-------------------------------------------------------------------------|                        |                        |          |         |            |          |          |
| CRUC-5668 | Branch Order in branch selector drop down should be sorted better | Seb Ruiz [Atlassian] Brydie McCoy [Atlassian]                                 | Closed                     | Fixed                     | Mar 08, 2011 | Mar 21, 2011 |

### From 2.5.1 to 2.5.2

**8 March 2011**

This is a bug fix release. The complete list of issues is below.

#### JIRA Issues (23 issues)

<p>| Key  | Type               | Summary                                                                 | Assignee                | Reporter                | Priority  | Status  | Resolution | Created  | Updated  |
|------|--------------------|-------------------------------------------------------------------------|                        |                        |          |         |            |          |          |
| CRUC-5645 | Files that are added/moved should take up the whole width of the pane, even if side by side view is selected | Geoff Crain [Atlassian] Joe Xie [atlassian]                               | Closed                     | Fixed                     | Feb 24, 2011 | Mar 01, 2011 |
| CRUC-5635 | Defect classification screens show id | Seb Ruiz Seb Ruiz                                                          | Fixed                     | Feb 21, 2011 | Feb 21, 2011 |</p>
<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-5612</td>
<td>Make Oracle multidb tests support testing of upgrades</td>
<td>Michael Studman</td>
<td>Michael Studman</td>
<td></td>
<td>Fixed</td>
<td>Closed</td>
<td>Feb 09, 2011</td>
<td>Feb 16, 2011</td>
</tr>
<tr>
<td>CRUC-5609</td>
<td>Review rss feed provides no way to navigate to review in thunderbird</td>
<td>Seb Ruiz</td>
<td>Brydie McCoy</td>
<td></td>
<td>Fixed</td>
<td>Closed</td>
<td>Feb 08, 2011</td>
<td>Feb 09, 2011</td>
</tr>
<tr>
<td>CRUC-5608</td>
<td>Cannot Edit Per-Repository Authentication Settings</td>
<td>Jason Hinch</td>
<td>Dylan Hansen</td>
<td></td>
<td>Fixed</td>
<td>Closed</td>
<td>Feb 08, 2011</td>
<td>Feb 09, 2011</td>
</tr>
<tr>
<td>CRUC-5599</td>
<td>Upgrade scripts do not always create the same schema as scripts to create a new database</td>
<td>Michael Studman</td>
<td>Michael Studman</td>
<td></td>
<td>Fixed</td>
<td>Closed</td>
<td>Feb 07, 2011</td>
<td>Feb 16, 2011</td>
</tr>
<tr>
<td>CRUC-5597</td>
<td>The &quot;Tabs&quot; on the statistics panel of the &quot;Project View&quot; do not work in JIRA 4.2.3 and JIRA 4.3</td>
<td>Joe Xie</td>
<td>Brendan Bain</td>
<td></td>
<td>Fixed</td>
<td>Closed</td>
<td>Feb 06, 2011</td>
<td>Feb 13, 2011</td>
</tr>
</tbody>
</table>

From 2.5.0 to 2.5.1

10 February 2011

This is a bug fix release. The complete list of issues is below.

**JIRA Issues (17 issues)**
Crucible 2.5 Upgrade Guide

Below are some important notes on upgrading to Crucible 2.5. For details of the new features and improvements in this release, please read the Crucible 2.5 Release Notes.

On this page:
- Upgrade Notes
- Upgrade Procedure
- Checking for Known Issues and Troubleshooting the Crucible Upgrade

**Upgrade Notes**

- **Crucible 2.5:**
  - **⚠️** This is only relevant when Crucible is run in conjunction with FishEye.
  
  This release will trigger an automatic upgrade of the metadata index the first time Crucible is started. Crucible is still usable while this task runs in the background, however the FishEye search functionality will not contain all results. A warning will appear in the UI until the upgrade has complete. Please also see How do I avoid long reindex times when I upgrade?

**Upgrade Procedure**
Before you begin

- Test your upgrades in your test environment before rolling into production.
- Back up your **entire** Crucible instance (see Backing Up and Restoring Crucible Data), i.e.
  - If you are backing up your Crucible instance via the Admin interface, tick all of the 'Include' checkboxes (e.g. plugins, templates, uploads, SQL database, etc).
  - If you are backing up your Crucible instance using the command-line interface, do not use any exclusion options.

If you are already running a version of Crucible, please follow these instructions on Upgrading to a New Version of Crucible.

**Checking for Known Issues and Troubleshooting the Crucible Upgrade**

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- **Check for known issues.** Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the [Crucible 2.5 Known Issues](https://confluence.atlassian.com/kwbase/known-issues) in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- **Did you encounter a problem during the Crucible upgrade?** Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

  - If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

**RELATED TOPICS**

- [Crucible 2.5 Release Notes](https://confluence.atlassian.com/kwbase/2.5-release-notes)
- [Crucible 2.4 Release Notes](https://confluence.atlassian.com/kwbase/2.4-release-notes)

**Crucible 2.4 Release Notes**

20 October 2010

With great pleasure, Atlassian presents the all-inclusive yet self-sufficient **Crucible 2.4**.

**Highlights of this Release:**

- Easier Application Linking
- Native Repository Access
- Starter Licenses
- Adding Changesets to Reviews Simplified
- User Interface Improvements
- Snippets Tweaks
- And Even More Improvements

**Responding to your Feedback:**

- Over 80 votes satisfied

**Highlights of Crucible 2.4**

- Thank you for all your issues and votes. [Keep logging issues](https://confluence.atlassian.com/kwbase/keep-logging-issues) to help us keep improving!
- Read the [release notices](https://confluence.atlassian.com/kwbase/release-notices) for important information about this release.
Easier Application Linking

Crucible now includes a brand new version of the Application Links plugin. You can use this plugin to easily link your Crucible instance to other applications, like a JIRA server or another Crucible instance. You can choose between the Trusted Applications protocol, OAuth or basic HTTP authentication. Linking two applications allows you to share information and access one application's functions from within the other. For example, if you linked your Crucible instance with a JIRA server, you could view JIRA issues in your Crucible activity stream or view the reviews associated with an issue/project in JIRA.

Native Repository Access

If you are using Crucible without Atlassian's FishEye, you can now connect to your Subversion, Perforce, CVS, Git, Mercurial or ClearCase repositories seamlessly without using plugins. You will notice a suite of improvements including:

- improved performance for day-to-day operations,
- commits displayed in all activity streams,
- the ability to search and browse for files during reviews, and
- repository administration via the Administration Console.

Upgrading to FishEye is also a breeze, if you choose to migrate to a full FishEye license in the future.

Starter Licenses

Crucible has joined the starter license party! Get started with code review on a full-featured perpetual license at a fraction of the cost. Best of all, all proceeds from our starter licenses go to charity.
Adding Changesets to Reviews Simplified

It's even easier to create reviews in this release, as we've simplified the process of adding changesets to reviews. You now only have to worry about which changeset to add to a review, rather than browse all the revisions. Revisions are added as iterations, by default. You can still remove revisions later, if you want.

User Interface Improvements

We're continuing our ongoing work to improve the Crucible user interface. This release includes a brand new Crucible inbox and a redesigned header. We've also improved the toolbars on a number of screens including the dashboard, as well as replaced dropdowns throughout the application with autocomplete controls.
Snippets Tweaks

The popular snippet reviews introduced in Crucible 2.3 have been further improved in this release. We've added inline editing of snippet titles as well as syntax highlighting, for existing snippets.

More....

And Even More Improvements

Visit our issue tracker to see the full list of improvements and bug fixes in FishEye and Crucible for this release.
Release Notices

- **Upgrading from a previous version of Crucible.** Upgrading Crucible should be fairly straightforward. We strongly recommend that you back up Crucible before upgrading. Please refer to the Crucible 2.4 Upgrade Guide for further essential information about your upgrade.

- **Known Issues.** Please check the important technical advisories on the front page of the Knowledge Base for information about any known issues for this release.

Crucible 2.4 Changelog

This page contains information about the Crucible 2.4 minor releases. FishEye license holders should also check the FishEye 2.4 Changelog.

Please read the Crucible 2.4 Upgrade Guide before upgrading to any of the minor releases below.

On this page:

- From 2.4.4 to 2.4.5
- From 2.4.3 to 2.4.4
- From 2.4.2 to 2.4.3
- From 2.4.1 to 2.4.2
- From 2.4.0 to 2.4.1

### From 2.4.4 to 2.4.5

**3 February 2011**

This is a bug fix release. The complete list of issues is below.

<table>
<thead>
<tr>
<th>JIRA Issues (6 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>---</td>
</tr>
</tbody>
</table>

### From 2.4.3 to 2.4.4

**11 January 2011**

This is a bug fix release. The complete list of issues is below.

The release includes fixes for a number of security issues. Please see the Security Advisory for more information.

<table>
<thead>
<tr>
<th>JIRA Issues (36 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>CRUC-5131</td>
</tr>
<tr>
<td>CRUC-4951</td>
</tr>
<tr>
<td>CRUC-4904</td>
</tr>
</tbody>
</table>
Crucible 2.4 Documentation

This is a bug fix release. The complete list of issues is below.

### JIRA Issues (23 issues)

<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CRUC-5008</td>
<td>File history review links are misstyled</td>
<td>Seb Ruiz</td>
<td>Seb Ruiz</td>
<td></td>
<td>Closed</td>
<td>Fixed</td>
<td>Nov 24, 2010</td>
<td>Nov 2 2010</td>
</tr>
<tr>
<td></td>
<td>CRUC-5003</td>
<td>Crucible and FishEye dashboard pages have missing or partial</td>
<td>Craig Sharkie</td>
<td>Craig Sharkie</td>
<td></td>
<td>Closed</td>
<td>Fixed</td>
<td>Nov 23, 2010</td>
<td>Nov 2 2010</td>
</tr>
<tr>
<td></td>
<td>CRUC-4998</td>
<td>ChangesetTooltipAction.findChangeSet() eagerly creates cruRevisions</td>
<td>Seb Ruiz</td>
<td>Brendan Humphreys</td>
<td></td>
<td>Closed</td>
<td>Fixed</td>
<td>Nov 22, 2010</td>
<td>Nov 2 2010</td>
</tr>
<tr>
<td></td>
<td>CRUC-4996</td>
<td>Map .json files to javascript syntax highlighting</td>
<td>Seb Ruiz</td>
<td>Seb Ruiz</td>
<td></td>
<td>Closed</td>
<td>Fixed</td>
<td>Nov 22, 2010</td>
<td>Nov 2 2010</td>
</tr>
<tr>
<td></td>
<td>CRUC-4991</td>
<td>Manifest implementation has large memory footprint</td>
<td>Anna Buttfield</td>
<td>Anna Buttfield</td>
<td></td>
<td>Closed</td>
<td>Fixed</td>
<td>Nov 21, 2010</td>
<td>Nov 2 2010</td>
</tr>
<tr>
<td></td>
<td>CRUC-4987</td>
<td>Git Indexing Issues</td>
<td>Conor MacNeill</td>
<td>Conor MacNeill</td>
<td></td>
<td>Closed</td>
<td>Fixed</td>
<td>Nov 18, 2010</td>
<td>Nov 2 2010</td>
</tr>
</tbody>
</table>

From 2.4.2 to 2.4.3

26 November 2010

This is a bug fix release. The complete list of issues is below.
### From 2.4.1 to 2.4.2

**11 November 2010**

This is a bug fix release. The complete list of issues is below.

#### JIRA Issues (63 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-ID</td>
<td>Description</td>
<td>Assignee(s)</td>
<td>Status</td>
<td>Resolution Date</td>
<td>Created Date</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>
</tr>
<tr>
<td>CRUC-4858</td>
<td>FishEye dialogue sends user to URL escaped URL</td>
<td>Michael Heemskerk [Atlassian]</td>
<td>Fixed</td>
<td>Sep 12, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4857</td>
<td>Watch email contains broken link when there is no context path</td>
<td>Michael Heemskerk [Atlassian]</td>
<td>Fixed</td>
<td>Jul 14, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4850</td>
<td>Upgrade to JIRA Blockers 1.6.9</td>
<td>Tim Pettersen [Atlassian]</td>
<td>Fixed</td>
<td>Nov 09, 2010</td>
<td>Nov 0, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4845</td>
<td>Map xslt files for xml syntax highlighting</td>
<td>Seb Ruiz [Atlassian]</td>
<td>Fixed</td>
<td>Nov 08, 2010</td>
<td>Nov 0, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4829</td>
<td>Javascript error occurs when clicking a folder when you have never selected a branch in the branch filter</td>
<td>Geoff Crain [Atlassian]</td>
<td>Fixed</td>
<td>Nov 05, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4799</td>
<td>Too many objects are kept in memory during changest indexing, resulting in an OutOfMemoryException</td>
<td>Michael Heemskerk [Atlassian]</td>
<td>Fixed</td>
<td>Nov 04, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4774</td>
<td>plain text rendering of wiki text in the activity stream for changeset comments is wrong</td>
<td>Tom Davies [Atlassian]</td>
<td>Fixed</td>
<td>Nov 02, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4742</td>
<td>Excludes cause indexing to fail for Git repositories</td>
<td>Unassigned</td>
<td>Fixed</td>
<td>Nov 01, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4710</td>
<td>addGeneralComment fails when called outside a HTTP request. e.g. a plugin commit listener</td>
<td>Joe Xie [atlassian]</td>
<td>Fixed</td>
<td>Oct 27, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4700</td>
<td>Activity tab for all repositories is gone in 2.4</td>
<td>Seb Ruiz [Atlassian]</td>
<td>Fixed</td>
<td>Oct 27, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4699</td>
<td>svn does not fully support the branch selector</td>
<td>Geoff Crain [Atlassian]</td>
<td>Fixed</td>
<td>Oct 26, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4698</td>
<td>ClearCase indexing misses revisions or indexes revisions multiple times if the server clocks of the ClearCase server and the FishEye servers are out of synch or in different timezones</td>
<td>Michael Heemskerk [Atlassian]</td>
<td>Fixed</td>
<td>Oct 26, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4690</td>
<td>No Description for Repository LDAP Settings</td>
<td>Seb Ruiz [Atlassian]</td>
<td>Fixed</td>
<td>Oct 26, 2010</td>
<td>Nov 0, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4681</td>
<td>Show the branch a changeset was committed to in watch emails</td>
<td>Seb Ruiz [Atlassian]</td>
<td>Fixed</td>
<td>Oct 25, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4679</td>
<td>ClearCase: 'Add' and 'Test Connection' buttons in admin don't do anything when both 'VOB To Include' and 'Include Pattern' and/or 'Exclude Pattern' is specified</td>
<td>Pierre-Etienne Poirot [Atlassian]</td>
<td>Fixed</td>
<td>Oct 25, 2010</td>
<td>Nov 1, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4675</td>
<td>ClearCase: nothing is indexed when branch excludes is non-empty and branch branch in the branch filter</td>
<td>Michael Heemskerk [Atlassian]</td>
<td>Fixed</td>
<td>Oct 25, 2010</td>
<td>Nov 0, 2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-ID</td>
<td>Description</td>
<td>Assigned To</td>
<td>Fixed By</td>
<td>Status</td>
<td>Resolved</td>
<td>Created</td>
<td>Updated</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
<td>----------</td>
<td>---------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>CRUC-4671</td>
<td>Path segments in Include/Exclude cannot be parsed in 2.4.0</td>
<td>Nick Pellow</td>
<td>Dylan Hansen</td>
<td>Closed</td>
<td>Oct 24, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4688</td>
<td>ClearCase: project exclude rules aren’t being stored</td>
<td>Michael Heemskerk</td>
<td>Michael Heemskerk</td>
<td>Fixed</td>
<td>Oct 24, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4667</td>
<td>ClearCase: specifying include or exclude rules for projects and vobs results in an invalid config.xml file, causing errors on restart</td>
<td>Michael Heemskerk</td>
<td>Michael Heemskerk</td>
<td>Closed</td>
<td>Oct 24, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4663</td>
<td>ClearCase: performance of incremental indexing has decreased in 2.4.0 (Base CC only)</td>
<td>Michael Heemskerk</td>
<td>Michael Heemskerk</td>
<td>Fixed</td>
<td>Oct 21, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4659</td>
<td>ClearCaseRepositoryCache.getChangeSet throws IllegalStateException when the changeset does not exist. The cache for other SCM types don’t</td>
<td>Michael Heemskerk</td>
<td>Michael Heemskerk</td>
<td>Closed</td>
<td>Oct 21, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4657</td>
<td>&quot;IllegalStateException: Indexer is stopped&quot; on 2.4 Upgrade</td>
<td>Seb Ruiz</td>
<td>Dylan Hansen</td>
<td>Closed</td>
<td>Oct 21, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4656</td>
<td>&quot;Edit Details&quot;, &quot;Summarize&quot; and &quot;Tools&quot; menu missing under &quot;Reviews&quot; tab in Opera in 2.4</td>
<td>Craig Sharkie</td>
<td>Gurleen Anand</td>
<td>Fixed</td>
<td>Oct 21, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4646</td>
<td>Quick file links in activity stream disappear if browser isn’t wide enough</td>
<td>Craig Sharkie</td>
<td>Nicolas Venegas</td>
<td>Fixed</td>
<td>Oct 20, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4636</td>
<td>Make more of GWT admin deep linkable</td>
<td>Michael Studman</td>
<td>Michael Studman</td>
<td>Fixed</td>
<td>Oct 19, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4623</td>
<td>Git: Use of file content-hash to map to file revision is insufficient</td>
<td>Michael Heemskerk</td>
<td>Michael Heemskerk</td>
<td>Fixed</td>
<td>Oct 18, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4614</td>
<td>Annotation links from file history don’t preserve wayback spec</td>
<td>Tom Davies</td>
<td>Seb Ruiz</td>
<td>Closed</td>
<td>Oct 18, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4613</td>
<td>Links to comments in the same review don’t work</td>
<td>Tom Davies</td>
<td>Seb Ruiz</td>
<td>Fixed</td>
<td>Oct 18, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4612</td>
<td>infinite redirects to login via dialog iframes</td>
<td>Tom Davies</td>
<td>Geoff Crain</td>
<td>Fixed</td>
<td>Jun 20, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4597</td>
<td>Clear filter button goes to changeset</td>
<td>Tom Davies</td>
<td>Chris Lam</td>
<td>Closed</td>
<td>Oct 15, 2010</td>
<td>Nov 2nd 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4585</td>
<td>Crucible Inbox: Summarize state review with incomplete reviewers shows in no filter</td>
<td>Tom Davies</td>
<td>Matthew Watson</td>
<td>Closed</td>
<td>Oct 14, 2010</td>
<td>Nov 2nd 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4580</td>
<td>Change the logging level of valid ‘error’ messages</td>
<td>Michael Heemskerk</td>
<td>Ross Rowe</td>
<td>Fixed</td>
<td>Jun 2nd 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4568</td>
<td>Committer name bolds when you mouseover it somewhere else</td>
<td>Craig Sharkie</td>
<td>Chris Lam</td>
<td>Fixed</td>
<td>Oct 13, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4547</td>
<td>Join Review is duplicated</td>
<td>Craig Sharkie</td>
<td>Seb Ruiz</td>
<td>Fixed</td>
<td>Oct 12, 2010</td>
<td>Nov 1st 2010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-4541</td>
<td>Sometimes can’t edit objectives</td>
<td>Adam Ahmed</td>
<td>Seb Ruiz</td>
<td>Fixed</td>
<td>Oct 12, 2010</td>
<td>Nov 2nd 2010</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Crucible 2.4 Documentation

| CRUC-4450 | ClearCase: CPU usage has increased significantly between 2.3.6 and 2.4-M6 | Michael Heemskerk [Atlassian] | Michael Heemskerk [Atlassian] | Fixed | Oct 04, 2010 | Nov 01, 2010 |
| CRUC-4415 | GWT admin for ClearCase is slow because cleartool commands are being executed too often and too early | Michael Studman [Atlassian] | Michael Heemskerk [Atlassian] | Fixed | Sep 27, 2010 | Nov 01, 2010 |
| CRUC-4411 | Reload of admin page causes CPU max out | Unassigned | Michael Heemskerk [Atlassian] | Fixed | Sep 27, 2010 | Nov 02, 2010 |

From 2.4.0 to 2.4.1

27 October 2010

This is a bug fix release. The complete list of issues is below.

<table>
<thead>
<tr>
<th>JIRA Issues (20 issues)</th>
<th>Type</th>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
</table>

This is a bug fix release. The complete list of issues is below.
## Crucible 2.4 Upgrade Guide

Below are some important notes on upgrading to Crucible 2.4. For details of the new features and improvements in this release, please read the Crucible 2.4 Release Notes.

### On this page:
- XSS vulnerability in FishEye/Crucible Code Macro
- XSS vulnerability in FishEye Repository Configuration
- XSS vulnerability in Crucible’s Linked JIRA Issue
- NullPointerExcepton on p4 describe
- Perforce patch reviews throw errors in 2.4.0
- Permisel element cannot be serialized to JSON for REST requests
- fix api breakage from 2.3.x to 2.4.0
- Perforce case-sensitivity issues on FE 2.4 Upgrade
- Perforce branch specification doesn't work
- NullPointerException when indexing perforce repositories on upgrade to 2.4
- NullPointerException in SvnCache.getFileRevision
- Invocation of git blame fails for files that contain spaces
- Whenever viewing the users tab, if the user has an email address for their login name, Crucible errors out
- Nasty error message when trying to start repo
- NPE when scanning git repo trying to read commit date:
- SVN branch selector enabled with aix loaded directories
- Testing a Connection in the Edit Repo Dialog, doesn't use the values from the form
- [IE7] - Horizontal Scrollbar on File History page
- Changeset link changes server url
- Changeset hover of CVS broken

### JIRA Issues

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-4673</td>
<td>fix api breakage from 2.3.x to 2.4.0</td>
<td>Joe Xie [atlassian]</td>
<td>Joe Xie [atlassian]</td>
<td>Fixed</td>
<td>Oct 24, 2010</td>
<td>Jan 05, 2011</td>
</tr>
<tr>
<td>CRUC-4649</td>
<td>Whenever viewing the users tab, if the user has an email address for their login name, Crucible errors out</td>
<td>Jason Hinch [atlassian]</td>
<td>Dylan Hansen [Atlassian]</td>
<td>Fixed</td>
<td>Oct 20, 2010</td>
<td>Nov 15, 2010</td>
</tr>
</tbody>
</table>
Upgrade Notes

**Crucible 2.4:**
- If you are using a Mercurial or Git repository with Crucible, it will be automatically re-indexed when you upgrade to Crucible 2.4. This is because we have upgraded the database schema for these SCMs.
- If you are using a lightSCM plugin to connect to a repository (i.e. you are not using FishEye), we recommend that you disable the lightSCM plugin in favour of the new native repository access functionality that is bundled with this Crucible release. Atlassian's lightSCM plugins (not lightSCM itself) are being deprecated in this release. Please see What happens if I decide to stop using FishEye with Crucible? for more information.

**Crucible 2.4.1:**
- If you use FishEye/Crucible with Perforce and previously upgraded to FishEye/Crucible 2.4.0, you must reindex all of your Perforce repositories after upgrading to FishEye/Crucible 2.4.1. Note, FishEye/Crucible will not automatically reindex your Perforce repositories, you will need to start the reindex manually.

**Crucible 2.4.2:**
- If you are using a ClearCase repository with Crucible, it will be automatically re-indexed when you upgrade to Crucible 2.4. This is because we have upgraded the database schema for this SCM.

Upgrade Procedure

⚠️ Upgrade a test environment first
As always, please test your upgrades in your test environment before rolling into production.

If you are already running a version of Crucible, please follow these instructions on Upgrading to a New Version of Crucible.

Checking for Known Issues and Troubleshooting the Crucible Upgrade

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- **Check for known issues.** Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the Crucible 2.4 Known Issues in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- **Did you encounter a problem during the Crucible upgrade?** Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

RELATED TOPICS

Crucible 2.4 Release Notes

Crucible 2.3 Release Notes

26 May 2010

ℹ️ For details on minor releases since Crucible 2.3, see the Crucible Changelog.

Atlassian presents Crucible 2.3

Crucible 2.3 is focused on the all-new lightweight Snippet Reviews and the innovative Review Coverage report, along with an enhanced installation wizard.

**Highlights of this release:**
- Snippet Reviews
- Changeset Discussions
- Mercurial SCM Alpha
- Review Coverage report
Highlights of Crucible 2.3

1 Snippet Reviews

Snippet Reviews are a new feature of Crucible, providing ultra-lightweight, ad-hoc code reviews with zero configuration. Atlassian is listening, and some customers asked for reviews with no ceremony, permissions or red tape. Snippet Reviews are instant to create, require no SCM repository and can be reviewed by anyone.

Screenshot: Crucible Snippet Review

2 Changeset Discussions

When using Crucible with FishEye, you can now leave comments on a changeset. Your colleagues will be able to read your comments and respond to them, creating a threaded discussion.
See the documentation for more.

Mercurial SCM Alpha

When using Crucible with FishEye, the Mercurial SCM can now be used. This release adds alpha support for Mercurial repositories. Atlassian is providing early access to the functionality for our customers; there are still a few kinks to be worked out, but it provides full access to FishEye.
When using Crucible with FishEye, a new Review Coverage Report is now available. A new paradigm that shows you the percentage of code that has been peer-reviewed, this report lets you easily see what parts of your codebase haven't had many eyes scanning them for a sanity check. This forms another kind of quality check for your software projects, along with unit testing and code coverage analysis.

*Screenshot: Crucible Review Coverage Report*
Revamped Installation Process

Crucible's installation process has been given a thorough revision and a visual facelift. It's now smoother, faster and provides a better experience.

Screenshot: Crucible’s New Installation Screen

See the documentation for more.
Gadgets

Crucible 2.3 includes a cluster of handy gadgets. These allow you to see Crucible data in other locations such as the JIRA or Confluence Dashboards.

These gadgets include the following:

- **To Do Gadget**
  This gadget is a list of Crucible to-do items including reviews to do, comments to read or reviews to summarise.

- **Hassle Gadget**
  This gadget shows you who you are still waiting on, in other words which reviewers haven't completed your reviews.

- **Overdue Reviews Gadget**
  This gadget shows you reviews that are yet to be completed in the project, across all authors. This is useful for managers or team leads.

- **Review Coverage Gadget**
  This gadget shows you information from the innovative Review Coverage Report, showing how much of your codebase has been subjected to code review.

These gadgets are published by default, and can be configured to appear on your JIRA or Confluence Dashboards.

**Screenshot: The Hassle Gadget**

<table>
<thead>
<tr>
<th>reviewer</th>
<th>ID</th>
<th>Name</th>
<th>(1 unread comment)</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR-1</td>
<td>FE-1234: fix Widget stamping</td>
<td>in 6 days</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR-1</td>
<td>FE-1234: fix Widget stamping</td>
<td>in 6 days</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR-4</td>
<td>FE-4567: Disentangle states</td>
<td>in 10 hours</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

See the [documentation](#) for more.

Plus numerous improvements and bug fixes

Visit our issue tracker to see the full list of improvements and bug fixes.

**Crucible 2.3 Changelog**

This page contains information about the Crucible 2.3 minor releases. FishEye license holders should also check the [FishEye 2.3 Changelog](#).

> Please read the Crucible 2.3 Upgrade Guide before upgrading to any of the minor releases below.

**On this page:**
- From 2.3.7 to 2.3.8
- From 2.3.6 to 2.3.7
- From 2.3.5 to 2.3.6
From 2.3.7 to 2.3.8

12 January 2011

This is a bug fix release that only addresses security issues. Please see the Security Advisory for more information.

The complete list of issues follows below.

<table>
<thead>
<tr>
<th>JIRA Issues (7 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-5345</td>
</tr>
<tr>
<td>CRUC-5309</td>
</tr>
<tr>
<td>CRUC-5313</td>
</tr>
<tr>
<td>CRUC-5307</td>
</tr>
<tr>
<td>CRUC-5308</td>
</tr>
<tr>
<td>CRUC-5311</td>
</tr>
<tr>
<td>CRUC-5312</td>
</tr>
</tbody>
</table>

From 2.3.6 to 2.3.7

20 October 2010

This release is to support the FishEye 2.3.7 release only. FishEye 2.3.7 fixes a FishEye security vulnerability. If you are using Crucible without FishEye, you are not affected by this security vulnerability.

From 2.3.5 to 2.3.6

25 August 2010

This is a bug fix release. The complete list of issues follows below.

<table>
<thead>
<tr>
<th>JIRA Issues (7 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-3887</td>
</tr>
<tr>
<td>CRUC-3758</td>
</tr>
<tr>
<td>CRUC-3874</td>
</tr>
<tr>
<td>CRUC-856</td>
</tr>
<tr>
<td>CRUC-3890</td>
</tr>
<tr>
<td>CRUC-3726</td>
</tr>
<tr>
<td>CRUC-4052</td>
</tr>
</tbody>
</table>

From 2.3.4 to 2.3.5

26 July 2010

This is a bug fix release. The complete list of issues follows below.

<table>
<thead>
<tr>
<th>JIRA Issues (1 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
From 2.3.3 to 2.3.4

13 July 2010

This is a bug fix release. The complete list of issues follows below.

<table>
<thead>
<tr>
<th>JIRA Issues (5 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>CRUC-3706</td>
</tr>
<tr>
<td>CRUC-3701</td>
</tr>
<tr>
<td>CRUC-3645</td>
</tr>
<tr>
<td>CRUC-3740</td>
</tr>
<tr>
<td>CRUC-3700</td>
</tr>
</tbody>
</table>

From 2.3.2 to 2.3.3

16th June 2010

This is a bug fix release that addresses security issues. Please see the Security Advisory for more information.

<table>
<thead>
<tr>
<th>JIRA Issues (3 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>CRUC-3635</td>
</tr>
<tr>
<td>CRUC-3289</td>
</tr>
<tr>
<td>CRUC-3630</td>
</tr>
</tbody>
</table>

From 2.3.1 to 2.3.2

3rd June 2010

This is a bug fix release. The complete list of issues follows below.

<table>
<thead>
<tr>
<th>JIRA Issues (9 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>CRUC-3613</td>
</tr>
<tr>
<td>CRUC-3610</td>
</tr>
<tr>
<td>CRUC-3609</td>
</tr>
<tr>
<td>CRUC-3581</td>
</tr>
<tr>
<td>CRUC-3580</td>
</tr>
<tr>
<td>CRUC-3568</td>
</tr>
<tr>
<td>CRUC-3553</td>
</tr>
<tr>
<td>CRUC-3547</td>
</tr>
<tr>
<td>CRUC-3499</td>
</tr>
</tbody>
</table>

From 2.3.0 to 2.3.1

27th May 2010

This is a bug fix release, addressing an issue that occurs when upgrading Crucible with an expired license.
Complete list of issues follows below.

<table>
<thead>
<tr>
<th>JIRA Issues (3 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-3596</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Startup is broken for upgrades with licenses that have expired maintenance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-3595</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest login method is ambiguous for CAPTCHA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-3584</td>
<td></td>
<td></td>
</tr>
<tr>
<td>When changeset comments are disabled for a repo, the changeset pane tells you to log in, even though you are already logged in.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Crucible 2.3 Upgrade Guide**

Below are some important notes on upgrading to **Crucible 2.3**. For details of the new features and improvements in this release, please read the **Crucible 2.3 Release Notes** and **Crucible 2.3 Changelog**.

**On this page:**

- Upgrade Notes
- Upgrade Procedure
- Checking for Known Issues and Troubleshooting the Crucible Upgrade

**Upgrade Notes**

- If you use FishEye with ClearCase and you upgrade to FishEye/Crucible 2.3.4 or later from a version prior to 2.3.4, FishEye/Crucible will reindex all ClearCase repositories.

**Upgrade Procedure**

**Upgrade a test environment first**

As always, please test your upgrades in your test environment before rolling into production.

If you are already running a version of Crucible, please follow these instructions on **Upgrading to a New Version of Crucible**.

**Checking for Known Issues and Troubleshooting the Crucible Upgrade**

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- **Check for known issues.** Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the **Crucible 2.3 Known Issues** in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- **Did you encounter a problem during the Crucible upgrade?** Please refer to the guide to **troubleshooting upgrades** in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a **support ticket** and one of our support engineers will help you.

**RELATED TOPICS**

- **Crucible 2.3 Release Notes**
- **Crucible 2.3 Changelog**

**Crucible 2.2 Release Notes**

**Crucible 2.5 has now been released. Read the Release Notes.**

18 February 2010
Atlassian presents Crucible 2.2

Crucible 2.2 is focused on improving the user experience, with its innovative pre-commit support, wizard-like review creation and JIRA time tracking integration.

Highlights of this release:

- Smart Pre-Commit (Patch) Support
- 'No Moderator' Reviews
- Wizard-Like Review Creation
- Integrated Timetracking Between Crucible and JIRA
- Edit Mode for Reviews
- Plus numerous improvements and bug fixes

Thank you for your interest in Crucible 2.2.

See the documentation on Upgrading to this version.

Installing Crucible 2.2

Download Crucible 2.2 now. If upgrading from a previous version, please follow the Upgrade Guide.

Highlights of Crucible 2.2

1

Smart Pre-Commit (Patch) Support

Previously, when viewing a patch in FishEye you could only see three lines of code around your code diffs (three lines of context). Sometimes, three lines is not enough. From Crucible 2.2, when creating a review from a patch file you can anchor it to revisions in your FishEye repository. FishEye will then automatically fetch more lines of code from the repository (beyond the default of three) and add them to the diff view. This gives reviewers direct access to the file's history and enables full context diffs, improving the review experience.

Screenshot: Crucible Patch Anchoring
'No Moderator' Reviews

Crucible is a lightweight code review tool, so the Crucible developers are always working to make it lighter. In Crucible 2.2, they've removed the requirement for a moderator, or a single person as the judge on each review. Moderators are a part of the 1970's Fagan Inspection doctrine for code reviews (which Crucible fully supports), but may not be right for your code development processes in 2010. Now, reviews can be freely opened, closed, re-opened, summarised, joined, quit or abandoned by any of the participants involved. These free-form reviews especially suit Agile or self-organising teams.

Screenshot: 'No Moderator' Reviews

See the documentation for more.
Wizard-Like Review Creation

Creating a review now follows a simple, wizard-like process where you make your content selections from a series of relevant sub-menus, allowing you to jump back and forth between steps, and edit any details right up until the review is launched.

Screenshot: Wizard-Like Review Creation

Integrated Timetracking Between Crucible and JIRA

Crucible already has time tracking as an inline feature, but now you can also submit time worked in Crucible to your JIRA issues. When you hover your mouse over the time tracking control in the Crucible top navigation bar, a special window appears that allows you to instantly fire off time estimates to JIRA, with a single click.

Screenshot: Crucible Time Tracking
With the new Edit Mode feature, you can now easily remove content from a review that has been started. You simply click the 'Edit Review' button in the left nav to launch Edit Mode. In Edit mode, you can quickly click red cross icons to remove files from the review. A single click returns you to regular Crucible functions, so you can more easily tune the content inside your reviews. Another button opens a dialog for rapidly adding more content to the review.

---

**Edit Mode for Reviews**

With the new Edit Mode feature, you can now easily remove content from a review that has been started. You simply click the 'Edit Review' button in the left nav to launch Edit Mode. In Edit mode, you can quickly click red cross icons to remove files from the review. A single click returns you to regular Crucible functions, so you can more easily tune the content inside your reviews. Another button opens a dialog for rapidly adding more content to the review.
Plus numerous improvements and bug fixes
Visit our issue tracker to see the full list of improvements and bug fixes.

**Crucible 2.2 Changelog**

This page contains information about the Crucible 2.2 minor releases. FishEye license holders should also check the FishEye 2.2 Changelog.

Please read the Crucible 2.2 Upgrade Guide before upgrading to any of the minor releases below.

On this page:

- From 2.2.1 to 2.2.3
- From 2.2.0 to 2.2.1

**From 2.2.1 to 2.2.3**

**4th May 2010**

This release addresses critical security issues. Atlassian strongly recommends that you upgrade to the latest version. Please read the Security Advisory for details.

This is a security and bug fix release, addressing the issues listed below. See the Security Advisory for specific information about the fixes and patches which are available.

<table>
<thead>
<tr>
<th>JIRA Issues (7 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
</tbody>
</table>

See the documentation for more.
Crucible version 2.2.2 was an internal release.

From 2.2.0 to 2.2.1

9th March 2010

This is a bugfix release, addressing the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (15 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-3232 Files in Review keep going missing</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>CRUC-3202 Add option to restrict user mappings to the admin UI</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3199 Clear filter doesn’t work</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3187 Cant set due time in a review</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3143 &quot;org.hibernate.HibernateException: Could not acquire unique-row lock, timeout&quot; exceptions under load</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3141 Excessive memory usage caused by some quicknav queries</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3139 Password recovery email link is broken</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3116 Issue keys and review keys are not rendered properly inside wiki tables</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3113 Filerevision linecount is incorrect for new files if it is merged from outside the fe repo</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3105 Optimize rendering of side-by-side segments</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3101 Add loading spinners to side-by-side diff</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-3079 Change appearance of side-by-side diff permalinks</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-2983 Update the JIRA FishEye plugin to comply with the new JIRA 4.1 View Issue look and feel</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>
**Crucible 2.2 Upgrade Guide**

Below are some important notes on upgrading to **Crucible 2.2**. For details of the new features and improvements in this release, please read the **Crucible 2.2 Release Notes** and **Crucible 2.2 Changelog**.

**On this page:**
- Upgrade Procedure
- Checking for Known Issues and Troubleshooting the Crucible Upgrade

**Upgrade Procedure**

**⚠️ Upgrade a test environment first**
As always, please test your upgrades in your test environment before rolling into production.

If you are already running a version of Crucible, please follow these instructions on **Upgrading to a New Version of Crucible**.

**Checking for Known Issues and Troubleshooting the Crucible Upgrade**

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- **Check for known issues.** Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the **Crucible 2.2 Known Issues** in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- **Did you encounter a problem during the Crucible upgrade?** Please refer to the guide to **troubleshooting upgrades** in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a **support ticket** and one of our support engineers will help you.

**RELATED TOPICS**

- **Crucible 2.2 Release Notes**
- **Crucible 2.2 Changelog**

**Crucible 2.1 Release Notes**

**⚠️ Crucible 2.5 has now been released. Read the Release Notes.**

**12 November 2009**

**ℹ️ For details on minor releases since Crucible 2.1, see the Crucible Changelog.**

**Atlassian presents Crucible 2.1**

**Crucible 2.1** adds Wiki Markup rendering, a new review history dialog, new review blockers report, and runs significantly faster.

**Highlights of this release:**

- Wiki Markup Rendering
Crucible 2.4 Documentation

- Progress Tracking
- Usability and Productivity Updates
- Streamlined JIRA Integration
- Review Time Tracking
- Review History Dialog
- "Blockers" Reports
- Threaded Comments
- Plugin Developer Tools
- Plus numerous improvements and bug fixes

Thank you for your interest in Crucible 2.1.

See the documentation on Upgrading to this version.

<table>
<thead>
<tr>
<th>Installing Crucible 2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can now download Crucible 2.1 from <a href="#">here</a>. If upgrading from a previous version, please follow the Upgrade Guide.</td>
</tr>
</tbody>
</table>

**Highlights of Crucible 2.1**

1. **Wiki Markup Rendering**

The Wiki Markup language that’s used in Confluence and JIRA can now be rendered by Crucible. Review comments, review descriptions and commit messages will now be shown rendering Wiki Markup code, allowing insertion of images, diagrams and text formatting. See the [documentation](#) for more.

*Screenshot: Wiki Markup Rendering in Crucible*

2. **Progress Tracking**

While reviewing, Crucible will now automatically remember which files you’ve read and show this as a percentage in the Review Details panel. You no longer have to explicitly click a checkbox to mark a file as reviewed. If you’ve only skimmed the page and plan to revisit it, you can select that a file’s status be left unread.

*Screenshot: Crucible Progress Tracking*
Usability and Productivity Updates

- **User Interface Update**
  The user interface has also been improved, consolidating items from the left navigation bar into the centre, freeing up space for the directory tree and other menus.

- **Floating File Mastheads**
  When viewing a review, you now have a floating masthead that contains information about the file in context. This gives you access to more meta-data about the file, you'll also have more control of the file and how it interacts with its peers. Additionally, more relevant data can be kept in focus.

- **Inline Editing for Review Details**
  Sometimes you just want to tweak a review's title, objective or summary – you can now edit these inline. The title will give you a cue that its editable by turning yellow when you hover over it. Click it to start editing and save your changes. The other areas have an edit link for you to use, come with a preview and are Wiki Markup ready as you'd expect.

- **Default Review Objectives**
  You can pre-populate reviews with default text so that you can avoid manually entering the same objectives for many reviews, where the goals for each one are similar.

- **JIRA auto-linker**
  When you create a review now, the title and objectives are scanned, looking for a JIRA reference. And when we find one, that JIRA is automatically associated with the Review.

- **Performance**
  Performance was also enhanced. The team focused on the main review page, the users page and FishEye pages that display large changesets.

- **Simplified navigation**
  In reviews, you can now choose to navigate at comment, defect or file level. You can easily jump between these items using a new, universal navigation control.
Streamlined JIRA Integration

When you create a review, the title and objectives are now parsed in the hunt for a JIRA reference. When we find one, that JIRA issue is automatically associated with the review. And if we’ve been a little over eager in sourcing a reference, simply click to remove it. Creating JIRA issues from within Crucible has also been enhanced, with an ‘Assignee’ drop-down being added:

Review Time Tracking

Crucible now has time tracking for review participants. When you’ve got a review open in your browser, Crucible will track the time you have spent on that particular review. You can also click to change the amount of time recorded. Totals are displayed in the Review Details panel.
Review History Dialog

The new History Dialog differentiates old and new states of your interactions with a review. The result is that now you get richer information about those interactions and more control. You can sort the information by date, actor, or action. This information can also be displayed in the new timeline mode.

Additionally, you can get access to the entire Review history through the CSV download link in the upper right corner, allowing for easy data import into a spreadsheet or other application.

Screenshot: Crucible Review History Dialog

Screenshot: Timeline Mode in the Crucible Review History Dialog
"Blockers" Reports

Every now and then, someone in the team can become a bottleneck in the review process. The new review blockers report helps by identifying team members that have a lot of reviews waiting in their inbox. Additionally, the JIRA blockers report accepts a JIRA feed, which you can find under the 'View' drop-down on your JIRA Issue Navigator page. Add the URL to the report and you get an insight to which issues and participants you need to follow up. Both these reports are plugins that come bundled with Crucible.
Threaded Comments

Crucible comments now support fully threaded discussions. Threads can also be collapsed.
Plugin Developer Tools

This release adds Atlassian Plugin SDK support to Crucible and FishEye, simplifying build management for plugin developers. The developer documentation for FishEye and Crucible has been co-located into a new documentation space as well.

Screenshot: Plugin Developer Tools for Crucible

Plus numerous improvements and bug fixes

Visit our issue tracker to see the full list of improvements and bug fixes.

Crucible 2.1 Changelog

This page contains information about the Crucible 2.1 minor releases. FishEye license holders should also check the FishEye 2.1 Changelog.

⚠️ Please read the Crucible 2.1 Upgrade Guide before upgrading to any of the minor releases below.

On this page:

- From 2.1.3 to 2.1.4
- From 2.1.2 to 2.1.3
- From 2.1.1 to 2.1.2
- From 2.1.0 to 2.1.1

From 2.1.3 to 2.1.4

27th January 2010

This is a bugfix release, addressing the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (3 issues)</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-2994</td>
<td>CLONE -Cannot create notes:// hyperlinks using WIKI renderer</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-2973</td>
<td>relax url validation in the linker</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>
From 2.1.2 to 2.1.3

13th January 2010

This is a bugfix release, addressing a number of ClearCase bugs and performance problems.

The following issues are addressed by this release:

<table>
<thead>
<tr>
<th>JIRA Issues (12 issues)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
<td><strong>Summary</strong></td>
<td></td>
</tr>
<tr>
<td>CRUC-2802</td>
<td>NPE when accessing to the people tab</td>
<td></td>
</tr>
<tr>
<td>CRUC-2759</td>
<td>FisheyePluginManagerFactory does not export com.atlassian.event</td>
<td></td>
</tr>
<tr>
<td>CRUC-2739</td>
<td>comment nav doesn't work if there is only 1 comment and it is already selected</td>
<td></td>
</tr>
<tr>
<td>CRUC-2710</td>
<td>NPE when setting up instance and only entering a fisheye license</td>
<td></td>
</tr>
<tr>
<td>CRUC-2706</td>
<td>Upgrading to 2.1 doesn't work with MySQL 4.x</td>
<td></td>
</tr>
<tr>
<td>CRUC-2697</td>
<td>Ghosting appears when sticky header is pushed above review toolbar</td>
<td></td>
</tr>
<tr>
<td>CRUC-2689</td>
<td>the todate= param doesn't work in changelog when clicked from the commit activity graph</td>
<td></td>
</tr>
<tr>
<td>CRUC-2278</td>
<td>Comments not showing if they are on out of context lines</td>
<td></td>
</tr>
<tr>
<td>CRUC-2216</td>
<td>Permission error when creating sub task from Crucible comment</td>
<td></td>
</tr>
<tr>
<td>CRUC-2166</td>
<td>the go-to-changeset input textbox cannot be clicked</td>
<td></td>
</tr>
<tr>
<td>CRUC-2129</td>
<td>Go to changeset field lacks feedback if cs doesn't exist</td>
<td></td>
</tr>
<tr>
<td>CRUC-1917</td>
<td>File link in the review is broken</td>
<td></td>
</tr>
</tbody>
</table>

From 2.1.1 to 2.1.2

19th November 2009

This is a bugfix release.

The following issues are addressed by this release:

<table>
<thead>
<tr>
<th>JIRA Issues (5 issues)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
<td><strong>Summary</strong></td>
<td></td>
</tr>
<tr>
<td>CRUC-2738</td>
<td>Getting java.lang.NoClassDefFoundError: Could not initialize class com.cenqua.fisheye.web.themer BriefCheckinCommentTag</td>
<td></td>
</tr>
<tr>
<td>CRUC-2713</td>
<td>Caused by: org.osgi.framework.BundleException: Unresolved constraint in bundle 32: package; (&amp;(package=org.apache.commons.collections)(version&gt;=3.2.0)((version&gt;=4.0.0)))</td>
<td></td>
</tr>
<tr>
<td>CRUC-2620</td>
<td>HSQLDB floods its sql log with AUTOCOMMIT TRUE/FALSE</td>
<td></td>
</tr>
<tr>
<td>CRUC-2421</td>
<td>(IE7): scrolling in multi frx view puts cpu at 99% for a while</td>
<td></td>
</tr>
<tr>
<td>CRUC-2412</td>
<td>frx menus overlap slider when window narrow</td>
<td></td>
</tr>
</tbody>
</table>
From 2.1.0 to 2.1.1

17th November 2009

This is a bugfix release.

The following issues are addressed by this release:

<table>
<thead>
<tr>
<th>JIRA Issues (24 issues)</th>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CRUC-2707</td>
<td>Paging of commit-only activity stream can be very slow</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2705</td>
<td>Document Supported Database server versions</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2704</td>
<td>Upgrading to 2.1 doesn't work with Postgresql 7.x</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2703</td>
<td>Changeset Navigation in manage files is broken</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2702</td>
<td>Plugins can't be enabled because of apache commons version constraints</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2700</td>
<td>Don't try to create a revisions drop down on the review contents tab when the revision is in a light scm repo which doesn't implement DirectoryBrowser</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2698</td>
<td>Lucene Index for Review State Changes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2645</td>
<td>something wrong with python syntax highlighting</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2191</td>
<td>Document supported database versions</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2118</td>
<td>Defect properties on comments are being lost</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2095</td>
<td>Activity stream says &quot;finished&quot; instead of &quot;uncompleted&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-2085</td>
<td>FE PERF - fixing the people page. Based on above perf data. Goal, all cases &lt; 10s, ideally &lt; 2s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1818</td>
<td>Double slashes appended incorrectly at the end of the SVN URL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1687</td>
<td>upgrade to latest jquery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1673</td>
<td>GenericJDBCException thrown on atlaseye</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1603</td>
<td>Due date display is too exact</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1601</td>
<td>find out the css differences between the review dir tree and the existing content dir tree</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1599</td>
<td>display settings for reviews arent intuitive and arent sticky</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1597</td>
<td>Need Performance Benchmarks for Crucible and Fisheye</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRUC-1449</td>
<td>Largish review beachballs on resize and when expanding FRXs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Crucible 2.1 Upgrade Guide

Below are some important notes on upgrading to Crucible 2.1. For details of the new features and improvements in this release, please read the Crucible 2.1 Release Notes and Crucible 2.1 Changelog.

On this page:
- Upgrade Procedure
- Checking for Known Issues and Troubleshooting the Crucible Upgrade

Upgrade Procedure

Upgrade a test environment first
As always, please test your upgrades in your test environment before rolling into production.

If you are already running a version of Crucible, please follow these instructions on Upgrading to a New Version of Crucible.

Checking for Known Issues and Troubleshooting the Crucible Upgrade

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- Check for known issues. Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the Crucible 2.1 Known Issues in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- Did you encounter a problem during the Crucible upgrade? Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

RELATED TOPICS

Crucible 2.1 Release Notes
Crucible 2.1 Changelog

Crucible 2.0 Release Notes

Crucible 2.5 has now been released. Read the Release Notes.

30 June 2009

Atlassian presents Crucible 2.0

Crucible 2.0 adds the all-new Iterative Reviews feature, enhanced JIRA integration and a brand new user interface.

Highlights of this release:
- Iterative Reviews
- New User Interface
- Read/Unread comments
Highlights of Crucible 2.0

Iterative Reviews

Crucible now allows you to review several revisions of a file within one review, seamlessly switching between them. Comments are linked and relative to a specific revision. This allows you to review every change that has occurred on a code file within a given period of time and hence visualise its evolution in the context of the review.

Screenshot: Iterative Reviews
New User Interface

Taking on board wide-ranging feedback from customers, the Crucible team has completely revamped the user interface of the product, adding more views on your work and allowing you to access controls from multiple locations and allowing for different work styles. Files in review are arranged in a tree for easy navigation. New viewing modes for reviews support full screen view, side-by-side diff view and single or multiple file views.

Screenshot: Reviewed files

Read/Unread comments

As you move around a review, Crucible keeps track of which comments you’ve seen and marks them as read. When you see a comment that you want to come back to, check the ‘leave as unread’ box so you don’t forget it. Unlike an email thread, new comments are rarely at the bottom. That’s why its especially useful to filter and highlight just the new comments when you come back to a review that’s underway.

Screenshot: Unread comments
Enhanced JIRA Integration

Crucible now has better JIRA integration, allowing you to see regular JIRA updates in your Crucible dashboard and create JIRA sub-tasks inline, without leaving the Crucible interface. You can still click on issue names to visit the JIRA instance they belong to, also. See instructions for JIRA configuration.

Screenshot: Enhanced JIRA Integration
Keyboard Shortcuts

Crucible now has keyboard shortcuts for navigating around your reviews efficiently. No more repetitive strain injury from that mouse wheel.

Screenshot: Keyboard shortcuts

Review Activity

All the activity that happens in Crucible is available as an activity stream. Streams can be accessed per project as well as a personal stream that includes the activity from people, projects, reviews and even comments you favorite or are involved in.

Screenshot: Project review activity stream
External Databases and Backup

Crucible now supports MySQL and Postgress in addition to the embeded HSQL database. Backup and restore capabilities have also been greatly enhanced.

Plus numerous improvements and bug fixes

Visit our issue tracker to see the full list of improvements and bug fixes.

Crucible 2.0 Changelog

This page contains information about the Crucible 2.0 minor releases. FishEye license holders should also check the FishEye 2.0 Changelog.

Please read the Crucible 2.0 Upgrade Guide before upgrading to any of the minor releases below.

On this page:
- From 2.0.5 to 2.0.6
- From 2.0.4 to 2.0.5
- From 2.0.3 to 2.0.4
- From 2.0.2 to 2.0.3
- From 2.0.1 to 2.0.2
- From 2.0 to 2.0.1
• From 2.0 Beta3 to 2.0
• From 2.0 Beta2 to 2.0 Beta3
• From 1.6.6 to 2.0 Beta2

From 2.0.5 to 2.0.6

8th October 2009

This is a bugfix release.

This release fixes a bug that affected Crucible-only installations.

From 2.0.4 to 2.0.5

6th October 2009

This is a bugfix and improvement release.

The following issues are addressed by this release:

<table>
<thead>
<tr>
<th>JIRA Issues (9 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-2317</td>
</tr>
<tr>
<td>CRUC-2295</td>
</tr>
<tr>
<td>CRUC-2214</td>
</tr>
<tr>
<td>CRUC-2203</td>
</tr>
<tr>
<td>CRUC-2135</td>
</tr>
<tr>
<td>CRUC-1856</td>
</tr>
<tr>
<td>CRUC-1742</td>
</tr>
<tr>
<td>CRUC-1734</td>
</tr>
<tr>
<td>CRUC-269</td>
</tr>
</tbody>
</table>

From 2.0.3 to 2.0.4

8th September 2009

This is a bugfix and improvement release.

• Crucible Light SCM Issues with Deleted Files: See the JIRA issue for details.

The following issues are addressed by this release:

<table>
<thead>
<tr>
<th>JIRA Issues (14 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-2160</td>
</tr>
<tr>
<td>CRUC-2128</td>
</tr>
<tr>
<td>CRUC-2112</td>
</tr>
<tr>
<td>CRUC-2105</td>
</tr>
<tr>
<td>CRUC-2104</td>
</tr>
<tr>
<td>CRUC-2101</td>
</tr>
</tbody>
</table>
CRUC-1952 Iterative reviews with uploaded files is broken
CRUC-1919 NonUniqueResultException: query did not return a unique result
CRUC-1907 Warn before dropping database in restore
CRUC-1402 NPE When using eclipse plugin
CRUC-1256 Crucible does not handle incomplete record data (via REST API) properly
CRUC-859 NPE when loading full history for file in lightSVN
CRUC-794 SVN LightSCM plugin tries to get revision which don't exist when copies are made
CRUC-761 Add warning that if user does not enter fisheye license they will not have access to fisheye capabilities

From 2.0.2 to 2.0.3

18th August 2009
This is a bugfix release which includes the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (14 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-2051</td>
</tr>
<tr>
<td>CRUC-2050</td>
</tr>
<tr>
<td>CRUC-2015</td>
</tr>
<tr>
<td>CRUC-1940</td>
</tr>
<tr>
<td>CRUC-1927</td>
</tr>
<tr>
<td>CRUC-1911</td>
</tr>
<tr>
<td>CRUC-1906</td>
</tr>
<tr>
<td>CRUC-1899</td>
</tr>
<tr>
<td>CRUC-1845</td>
</tr>
<tr>
<td>CRUC-1833</td>
</tr>
<tr>
<td>CRUC-1828</td>
</tr>
<tr>
<td>CRUC-1814</td>
</tr>
<tr>
<td>CRUC-1094</td>
</tr>
<tr>
<td>CRUC-854</td>
</tr>
</tbody>
</table>

From 2.0.1 to 2.0.2

24th July 2009
This is a bugfix release which includes the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (15 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
</tbody>
</table>


Crucible 2.4 Documentation

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-1886</td>
<td>UTF-8 encoding of forms not being handled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1831</td>
<td>various misc wording improvements in Crucible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1791</td>
<td>when creating a review from a changeset in the activity stream, don't add the revision number to the objectives</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1781</td>
<td>[Security Vulnerability] - REST API is working even if Remote API is OFF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1780</td>
<td>Invalid filter parameter could lead to security issues</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1779</td>
<td>stored paths can be invalid when creating reviews at the same time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1771</td>
<td>Switching between single file and multi-file should be hoisted to the tool bar</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1770</td>
<td>Switching to fullscreen should be hoisted to an icon on the tool bar</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1747</td>
<td>NPE in atlaseye for lauchlan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1746</td>
<td>Adding revisions iteratively breaks in litesvn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1739</td>
<td>change 'summarise' to 'summarize' on review age chart</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1729</td>
<td>bad wording of comment activity tool menu 'add favourites'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1720</td>
<td>Error logging in if the project or repository added as a favourite has been deleted</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1715</td>
<td>links to review inline comments don't work properly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1384</td>
<td>Review comments and replies added via REST API do not trigger e-mail notifications</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**From 2.0 to 2.0.1**

**14th July 2009**

This is a bugfix release which includes the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (29 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-1743</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1716</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1703</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1701</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1691</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-1690</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CRUC-1686 Editing Permission Schemes throws javax.el.PropertyNotFoundException  
Closed

CRUC-1683 Non-empty folder doesn't have comment count in tree  
Closed

CRUC-1682 Opening a review with a 2mb patch causes the review to take 1 minute to open  
Closed

CRUC-1676 Empty folders are incorrectly ordered  
Closed

CRUC-1672 "Create Review" on browse page does not add-to-review, but always creates a new review  
Closed

CRUC-1671 Empty folder doesn't have folder icon in tree  
Closed

CRUC-1670 FRXs with the same path are collapsed in the navigation tree  
Closed

CRUC-1669 Patch reviews shouldn't allow deletion of frxrevisions  
Closed

CRUC-1664 Backup admin page includes all items when de-selecting all  
Closed

CRUC-1657 draft comments are shown as unread from the reviews list  
Closed

CRUC-1655 inline comments need revisiting in side-by-side diff mode  
Closed

CRUC-1654 Restore user to previous position in review when reloading from a review update  
Closed

CRUC-1653 Show "reviews to do" table when closing a review  
Closed

CRUC-1650 edit revisions allows you to try and add a revision already in the review, then complains  
Closed

CRUC-1638 terminology - should use "unreviewed" or "not reviewed"  
Closed

CRUC-1636 inconsistent state names  
Closed

CRUC-1630 Grey out menu items when not applicable  
Closed

CRUC-1619 due dates aren't updated when new revisions are added  
Closed

CRUC-1606 Sort indicator on "Age" column of reviews dashboard points in the wrong direction  
Closed

CRUC-1598 Add linked reviews to the review sidebar.  
Closed

CRUC-1579 Editing a User in crucible standalone will render an sterisk next to their name  
Closed

CRUC-1567 Comment hovers only on text in IE8  
Closed

CRUC-1443 Safari issue - Tools dropdown causes 'invisible scrollbars'  
Closed

### From 2.0 Beta3 to 2.0

30th June 2009
Full list of issues in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (200 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-1749 XSS issue in filter parameter</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1661 File marked as unreviewed for defect author</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1656 RC3 - NPE when trying to create first project on empty CRU install</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1651 clicking [add] branch in SVN admin symbolic screen causes page-pop and 404</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1649 toggling Display Preferences &gt; Show Source in a review does nothing</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1644 Some display preferences don't update when clicked</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1641 error mapping JIRA servers</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1623 Admin: Default reviewers are not re-saved</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1616 Tune polling period.</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1615 Reset the delay when actively using the review</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1612 Move the warning to a Gmail-style overlay</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1611 Commenting doesn't work in Safari 4</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1609 CAC documentation for iterative review info from CRUC-1536</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1481 Warning dialogs and polling when new stuff added since last refresh</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1445 STUPID: old style error page when trying to access fisheye pages</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1442 Crucible Standalone - Logout screen has FishEye icon &amp; link</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1413 Adding all changesets for new files with multiple commits should show file as &quot;Added&quot;, not &quot;Modified&quot;</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1412 &quot;Earlier changeset&quot; and &quot;Later changeset&quot; links should be grayed out when earliest/latest already listed.</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1408 Images in review blend in to crucible</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1404 Review throws 500 error NPE</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1403 Rename &quot;crucibledb&quot; to &quot;sql&quot;</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1399 Default Reviewer for project is not exposed through the SPI</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1398 overdue but review has not been started. Setting a due date in the past causes this.</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>CRUC-1393</td>
<td>Refactor Email Comments Page into a Dialog</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1392</td>
<td>fix remove file behaviour of current content panel</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1391</td>
<td>Use newDirTreeBox in the files pane</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1390</td>
<td>Fix Uploads</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1388</td>
<td>Missing css files on Send Email page / Layout needs love</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1386</td>
<td>fix suggest reviewer ui</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1383</td>
<td>suggest reviewers - turn blame calculation back on</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1382</td>
<td>defect metrics charts are broken</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1381</td>
<td>.png file not detected as binary</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1380</td>
<td>Email Review button from the dashboard doesn't work / Email review in toils menu gives a 404 - permaid is undefined</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1379</td>
<td>clicking HELP opens help twice / Help in Tools menu will open a new window and change your window</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1378</td>
<td>page load error &quot;The requested URL /atlaseye/static/kidi0j/2static/script/fe/fisheye-profile.js was not found on this server.&quot;</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1377</td>
<td>wrong review actions (comments.txt and Email Review) are visible</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1376</td>
<td>we need to handle multiple errors in a smarter way</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1375</td>
<td>There is no jira quick link in the edit review dialog</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1374</td>
<td>Review info popup on the dashboard shouldn't show when clicking on a link such as the review permaId</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1373</td>
<td>Page load error &quot;The requested URL /atlaseye/static/kidi0j/2static/images/sprite_arrows.png was not found on this server&quot;</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1371</td>
<td>External dialog 'View ...' links need to redirect to review and trigger filter</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1370</td>
<td>Show current content panel in manage files</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1369</td>
<td>make the side filters like the fisheye tabs</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1368</td>
<td>bring back the custom filter</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1367</td>
<td>make the pages consistent</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1366</td>
<td>Add warnings for incomplete FRXs and unresolved JIRAs to dialogs</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1365</td>
<td>Move review summary form into a dialog.</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1364</td>
<td>Move creation-time review suggestions into a dialog</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1363</td>
<td>Convert project selection screen to a dialog</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1362</td>
<td>Backup broke Crucible Blitz</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1361</td>
<td>General Issues</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1360</td>
<td>Manage Files</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1359</td>
<td>Edit review form</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1358</td>
<td>Interstitials/dialogs</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1357</td>
<td>Dashboard Fixes</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1356</td>
<td>Remove driver url parameter from Restore and use new DBInfo properties</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1354</td>
<td>Refactor programatic frx filtering to not invoke click events</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1351</td>
<td>Add path and base filename to backup UI</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1345</td>
<td>Extend QuartzManager to support persistent configuration data per job</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1344</td>
<td>Rest Api returns the wrong URL for download Patch</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1332</td>
<td>Going to full context doesn't work</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1330</td>
<td>Put custom quartz job scheduling in a separate tab</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1329</td>
<td>Dynamically update general comment counts</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1328</td>
<td>Add general comments to the tree</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1327</td>
<td>Add the BackupItem selection to the scheduled backup</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1325</td>
<td>Add file type css place holders</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1324</td>
<td>FishEye only indicate Crucible review for a file with latest revision</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1319</td>
<td>Update doco for admin backup page</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1317</td>
<td>Dynamically update comment counts</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1316</td>
<td>Add comment counts to the frx tree</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1305</td>
<td>Springify Backup Manager</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1303</td>
<td>Write unit test</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1294</td>
<td>Check start/stop no longer needed in restore</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1293</td>
<td>Render maintenance page for non-admin pages when a backup is being made</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1291</td>
<td>Add &quot;Change Diff&quot; dropdown to manipulate both fromRev and toRev independently</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1290</td>
<td>Collapse common directories</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1289</td>
<td>Apply tree html structure and css styling to list</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1288</td>
<td>Treeify FRX list pane</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1286</td>
<td>add other filters</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1285</td>
<td>add filter - with comments</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1284</td>
<td>add UI for dropdown and expand/collapse</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1283</td>
<td>Drop-down filter for frx tree</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1275</td>
<td>Convert stored dates to long</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1274</td>
<td>Create FRX anchors and adjust the browser location as you navigate</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1273</td>
<td>FRXs should start expanded and maintain position when loading frxs above</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1272</td>
<td>Told that other participants have drafts, but also that all other participants have zero drafts</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1271</td>
<td>Move edit review box into AJS.Dialog</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1270</td>
<td>Integrate the summarise and close pages</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1254</td>
<td>Rearrange content</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1253</td>
<td>Review meta information sub pane</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1252</td>
<td>Move manage files tabs into AJS.Dialog</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1251</td>
<td>scroll frx pane when clicking on frx list item</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1250</td>
<td>create floating frx info subpane which updates on scroll</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1248</td>
<td>Modify overview header</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1247</td>
<td>3 Pane UI</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1242</td>
<td>Delete the old backup mechanism</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-ID</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>CRUC-1241</td>
<td>Rewire the quartz job</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1235</td>
<td>Revision slider hardening/tweaking</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1233</td>
<td>Wrap option</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1232</td>
<td>Delete comment refactor</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1215</td>
<td>Quartz backup job must invoke the new xml-based backup</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1208</td>
<td>reorganise contents of static directory</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1206</td>
<td>add new preference for new frxrevisions</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1205</td>
<td>add new notification for adding frxRevisions</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1201</td>
<td>displaying frx does not take into account showAsDiff flag</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1179</td>
<td>Sort FRX - alpha server side sort</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1174</td>
<td>Add banner warning &quot;you have unresolved jiras&quot; to page</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1173</td>
<td>Add resolve all JIRAs button &amp; action to summarize page</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1171</td>
<td>XML-RPC bollocks</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1170</td>
<td>Delete a defect's linked JIRA subtask when deleting the defect</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1169</td>
<td>ui &amp; action</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1168</td>
<td>XML-RPC bollocks</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1167</td>
<td>Resolve a defect's linked JIRA subtask from Crucible</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1166</td>
<td>XML-RPC bollocks</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1165</td>
<td>UI display of jira link and status + post error creating</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1164</td>
<td>schema</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1163</td>
<td>Automatically add defects as subtasks to JIRA &amp; link to them</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1162</td>
<td>guessing + jira validation</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1161</td>
<td>edit details ui + actions</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1160</td>
<td>schema</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1159</td>
<td>Link a review to a single JIRA and enable auto defect creation</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1158</td>
<td>just do it</td>
<td>✅</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------------------------------------------------</td>
<td>----</td>
</tr>
<tr>
<td>CRUC-1157</td>
<td>Configure a JIRA instance for use by Crucible</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1156</td>
<td>admin UI + create review</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1155</td>
<td>schema</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1154</td>
<td>Configure a default review duration for a project in week days so the due date can be pre-populated</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1153</td>
<td>highlighting overdue</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1152</td>
<td>sorting in the lists</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1151</td>
<td>Order any list of reviews by due date &amp; highlight any that are overdue</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1150</td>
<td>due date field in UI</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1149</td>
<td>schema</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1148</td>
<td>Set or change a due date on a review</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1143</td>
<td>just do it</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1142</td>
<td>Combine diff and annotation views</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1141</td>
<td>just do it</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1140</td>
<td>show the reviews you can't update without buttons</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1139</td>
<td>Author colours</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1138</td>
<td>Invisible markers (between lines)</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1137</td>
<td>Mouseover</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1136</td>
<td>Improve comment marker behaviour (tetris bar)</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1134</td>
<td>replace yahoo shite if we can</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1133</td>
<td>event binding</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1132</td>
<td>error handling</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1131</td>
<td>namespace &amp; multiple files</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1130</td>
<td>remove prototype</td>
<td>✅</td>
</tr>
<tr>
<td>CRUC-1129</td>
<td>client side js model</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1128</td>
<td>refactor commentator.js and friends</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1127</td>
<td>just do it</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1126</td>
<td>handle modifier keypress</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1125</td>
<td>just do it</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1124</td>
<td>Popup a keyboard shortcut help overlay when I press '?'</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1123</td>
<td>add user pref</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1122</td>
<td>add complete status resetter to the defect notification</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1121</td>
<td>Reset my complete status when new defects are added to a review</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1119</td>
<td>Tweak comment scroll tracker behaviour</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1118</td>
<td>Improve revision slider behaviour</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1117</td>
<td>add unread comment counts per revision</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1114</td>
<td>fix rendering/script issue in IE</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1113</td>
<td>make the slider handles move in unison</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1108</td>
<td>Mark comment read when it hits top of the viewport</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1105</td>
<td>adding to review doesn't work properly when you add the first file after a branch</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1097</td>
<td>Remove the use of half closed TCP connections</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1096</td>
<td>Change SPI to support multiple revisions per FRX</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1092</td>
<td>Show user's lastname and firstname when selecting a reviewer</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1089</td>
<td>Crucible Project should move from Global Settings to own Section</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1083</td>
<td>render author + participant results</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1062</td>
<td>api changes for multi-revisions and revision comments</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1060</td>
<td>Comment-counts displayed are not taking replies into consideration</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1057</td>
<td>Expand Review Summary Email</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1048</td>
<td>Change the admin backup page to use the new backup/restore implementation</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-1044</td>
<td>When upgrading from fisheye to crucible, the UI does not allow you to enter the Crucible license</td>
<td>![Status]</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>CRUC-977</td>
<td>issues comments added on changed and unchanged code don't show up none or less context</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-971</td>
<td>CRU Defect–&gt; JIRA Subtask creation</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-940</td>
<td>Comments made on a line not in the diff is not shown</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-927</td>
<td>Add a warning in the product that re-sync in large LDAP servers can cause performance issue</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-922</td>
<td>Admin screens with config.xml write and create/migrate workflow</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-921</td>
<td>Set up maintenance Mode</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-920</td>
<td>set up tests to compare upgrades of each database type</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-918</td>
<td>Write tests to determine/compare expected results from database tables</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-915</td>
<td>Usage Documentation for web-based backup/restore feature</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-912</td>
<td>Integration Tests</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-910</td>
<td>Admin Backup Page</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-909</td>
<td>Backup Status Page</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-907</td>
<td>Online Backup/Restore</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-882</td>
<td>REST API return other people's drafts</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-861</td>
<td>Allow soft wrap on diff and annotated display</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-860</td>
<td>Autosave has issues</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-858</td>
<td>&quot;Lines are missing&quot; divider graphics are no longer there</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-845</td>
<td>Describe this in the CAC docs.</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-825</td>
<td>remove abandon button from edit details section</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-824</td>
<td>If we haven't already we should automatically create a backup of our CRUDB (and other configuration files) before upgrading</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-823</td>
<td>Add an Expand Unchecked Files option</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-813</td>
<td>Side-by-side diff mode is too wide</td>
<td>![Status]</td>
</tr>
<tr>
<td>CRUC-770</td>
<td>Support file upload via REST</td>
<td>![Status]</td>
</tr>
</tbody>
</table>
CRUC-745  Setup page doesn't warn of invalid fe licence when valid cru licence is provided  
CRUC-741  Change moderator via REST  
CRUC-734  "Side-by-side" Diffs option does not update upon refreshing the Firefox browser  
CRUC-603  Support Anonymous Users Properly  
CRUC-581  Include Light SCM Repos in information returned by REST API  
CRUC-537  Summary Report & Searching by File  

From 2.0 Beta2 to 2.0 Beta3

5th June 2009

Full list of issues in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (35 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-1524</td>
</tr>
<tr>
<td>CRUC-1515</td>
</tr>
<tr>
<td>CRUC-1513</td>
</tr>
<tr>
<td>CRUC-1512</td>
</tr>
<tr>
<td>CRUC-1508</td>
</tr>
<tr>
<td>CRUC-1507</td>
</tr>
<tr>
<td>CRUC-1506</td>
</tr>
<tr>
<td>CRUC-1505</td>
</tr>
<tr>
<td>CRUC-1504</td>
</tr>
<tr>
<td>CRUC-1503</td>
</tr>
<tr>
<td>CRUC-1501</td>
</tr>
<tr>
<td>CRUC-1499</td>
</tr>
<tr>
<td>CRUC-1496</td>
</tr>
<tr>
<td>CRUC-1493</td>
</tr>
<tr>
<td>CRUC-1492</td>
</tr>
<tr>
<td>CRUC-1491</td>
</tr>
<tr>
<td>CRUC-1487</td>
</tr>
<tr>
<td>CRUC-1486</td>
</tr>
<tr>
<td>CRUC-1484</td>
</tr>
<tr>
<td>CRUC-1480</td>
</tr>
<tr>
<td>CRUC-1477</td>
</tr>
<tr>
<td>CRUC-1475</td>
</tr>
<tr>
<td>CRUC-1474</td>
</tr>
</tbody>
</table>
From 1.6.6 to 2.0 Beta2

Full list of issues in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (13 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-1488</td>
</tr>
<tr>
<td>CRUC-1478</td>
</tr>
<tr>
<td>CRUC-1476</td>
</tr>
<tr>
<td>CRUC-1465</td>
</tr>
<tr>
<td>CRUC-1464</td>
</tr>
<tr>
<td>CRUC-1463</td>
</tr>
<tr>
<td>CRUC-1462</td>
</tr>
<tr>
<td>CRUC-1458</td>
</tr>
<tr>
<td>CRUC-1454</td>
</tr>
<tr>
<td>CRUC-1439</td>
</tr>
<tr>
<td>CRUC-1427</td>
</tr>
<tr>
<td>CRUC-1396</td>
</tr>
<tr>
<td>CRUC-1116</td>
</tr>
</tbody>
</table>

Crucible 2.0 Upgrade Guide

Below are some important notes on upgrading to Crucible 2.0. For details of the new features and improvements in this release, please read the Crucible 2.0 Release Notes and Crucible 2.0 Changelog.

On this page:

- Upgrade Notes
- Supported Browsers
- MySQL Database Issues
- Problems with Crucible Freezing Unexpectedly
- Upgrade Procedure
- Checking for Known Issues and Troubleshooting the Crucible Upgrade
Upgrade Notes

Supported Browsers

Crucible 2.0 now supports the following browsers:

- Safari 3 (or later)
- FireFox 3 (or later)
- Internet Explorer 7 (or later)

⚠️ Please note, Internet Explorer 6 is no longer supported.

MySQL Database Issues

When migrating your database to MySQL, you may encounter problems with very long comments in MySQL.

Problems with Crucible Freezing Unexpectedly

A known issue may cause Crucible 2.0 to freeze unexpectedly.

Upgrade Procedure

⚠️ Upgrade a test environment first
As always, please test your upgrades in your test environment before rolling into production.

If you are already running a version of Crucible, please follow these instructions on Upgrading to a New Version of Crucible.

Checking for Known Issues and Troubleshooting the Crucible Upgrade

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- Check for known issues. Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the Crucible 2.0 Known Issues in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- Did you encounter a problem during the Crucible upgrade? Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

RELATED TOPICS

Crucible 2.0 Release Notes
Crucible 2.0 Changelog

Crucible 2.0 Beta Release Notes

⚠️ Crucible 2.5 has now been released. Read the Release Notes.

Crucible 2.0 Beta is a public development release leading up to Crucible 2.0. For all production use and testing of Crucible, please use the latest official release.

⚠️ This page refers to an updated version of the Beta (Beta 3). We strongly recommend all beta users upgrade to this release.

⚠️ Do not use in production.
Beta releases should not be used in production environments.
Please also take note of the following information:

- Beta releases are not safe — Beta releases are snapshots of the ongoing Crucible development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, may change or be removed before the next full release.
  - FireFox 3 and Safari are the only browsers supported.

5 June 2009

Atlassian presents Crucible 2.0 Beta

Crucible 2.0 adds the all-new Iterative Reviews feature, enhanced JIRA integration and a brand new user interface.

Highlights of this release:

- Iterative Reviews
- Enhanced JIRA Integration
- Stars Feature
- Unique Avatars
- People View
- New User Interface
- Plus numerous improvements and bug fixes

Thank you for your interest in Crucible 2.0 Beta.

You can now download the Crucible 2.0 Beta from here. See the documentation on Upgrading to this version.

Highlights of Crucible 2.0 Beta

Iterative Reviews

Crucible now allows you to review several revisions of a file within one review, seamlessly switching between them. Comments are persistent and relative to a specific revision. This allows you to review every change that has occurred on a code file within a given period of time and hence visualise its evolution in the context of the review.

Screenshot: Iterative Reviews
Enhanced JIRA Integration

Crucible now has better JIRA integration, allowing you to see regular JIRA updates in your Crucible dashboard and create JIRA sub-tasks inline, without leaving the Crucible interface. You can still click on issue names to visit the JIRA instance they belong to, also. See instructions for JIRA configuration.

Screenshot: Enhanced JIRA Integration
Stars Feature

Crucible now allows you to create a list of “Starred” favourite items that includes reviews, JIRA issues, colleagues and more. These favourites can be viewed together in aggregate as a stream of active work you are doing.

Screenshot: Stars

Unique Avatars

Crucible will now generate a unique “Charlie” image that will be used as your avatar in the system. These avatars create a new visual linkage to the personalities working on various items and are used as a visual shorthand to show user involvement on items in menu screens and dialogs.

Screenshot: Unique Avatars
You can now view detailed charts and activity statistics people who use your Crucible instance. You can compare number of reviews complete and other metrics charted over time.

<table>
<thead>
<tr>
<th>Due</th>
<th>Reviewers</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 days ago</td>
<td>🧵</td>
</tr>
<tr>
<td>11 days ago</td>
<td>🧵</td>
</tr>
<tr>
<td>10 days ago</td>
<td>🧵</td>
</tr>
<tr>
<td>3 days ago</td>
<td>🧵</td>
</tr>
<tr>
<td>4 hours ago</td>
<td>🧵</td>
</tr>
<tr>
<td>3 hours ago</td>
<td>🧵</td>
</tr>
<tr>
<td>79 minutes ago</td>
<td>🧵</td>
</tr>
</tbody>
</table>

People View

You can now view detailed charts and activity statistics people who use your Crucible instance. You can compare number of reviews complete and other metrics charted over time.

Screenshot: People View

---

<table>
<thead>
<tr>
<th>User</th>
<th>Latest Activity</th>
<th>Reviews</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geoff Crain</td>
<td>🧵 on 29 May 2009</td>
<td>516</td>
</tr>
<tr>
<td>Enk van Zijst</td>
<td>🧵 on 29 May 2009</td>
<td>459</td>
</tr>
<tr>
<td>Craig Sharkie</td>
<td>🧵 on 29 May 2009</td>
<td>71</td>
</tr>
<tr>
<td>Conor MacNeill</td>
<td>🧵 on 29 May 2009</td>
<td>461</td>
</tr>
<tr>
<td>Nick Fellow</td>
<td>🧵 on 29 May 2009</td>
<td>195</td>
</tr>
<tr>
<td>Sebastian Ruiz</td>
<td>🧵 on 29 May 2009</td>
<td>375</td>
</tr>
<tr>
<td>Joe Xie</td>
<td>🧵 on 29 May 2009</td>
<td>231</td>
</tr>
<tr>
<td>Peter McNeil</td>
<td>🧵 on 29 May 2009</td>
<td>517</td>
</tr>
</tbody>
</table>
New User Interface

Taking on board wide-ranging feedback from customers, the Crucible team has completely revamped the user interface of the product, adding more views on your work and allowing you to access controls from multiple locations and allowing for different work styles. Files in review are arranged in a tree for easy navigation. New viewing modes for reviews support full screen view, side-by-side diff view and single or multiple file views.

Screenshot: New User Interface

Plus numerous improvements and bug fixes

Visit our issue tracker to see the full list of improvements and bug fixes between Beta 2 and Beta 3. We strongly recommend all beta users upgrade to the latest beta release.

See the Beta Reviewer's Guide for a list of known issues and guidance on the beta experience.

Crucible 2.0 Beta Upgrade Notes

Crucible 2.0 Beta is a public development release leading up to Crucible 2.0. For all production use and testing of Crucible, please use the latest official release.

⚠️ Do not use in production.
Beta releases should not be used in production environments.
Please also take note of the following information:

- Beta releases are not safe — Beta releases are snapshots of the ongoing Crucible development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
  - There will be an upgrade path from the 2.0 Beta to the final release.

This page contains instructions on how to upgrade your Crucible instance to the Crucible 2.0 Beta.

Read about how your Crucible installation works with FishEye.

**Before you Start**

- Before upgrading you should always read the Release Notes for the version you are upgrading to, as well as any versions you are skipping.
- **We strongly recommend you make a backup of your data before upgrading Crucible.** Simply make a copy of your crucible_install_dir/var/data/ directory.
- Download the Crucible zip file.

**Upgrade Procedure**

**Method 1 - Using a FISHEYE_INST Directory**

If you have Crucible configured to use a FISHEYE_INST directory, then simply:

1. Shutdown your existing fisheye server
2. Make a backup of your FISHEYE_INST directory
3. Extract the new Crucible version to a directory.
4. Leave your FISHEYE_INST environment variable set to its existing location.
5. Start Crucible from the new installation.

Read more about the FISHEYE_INST environment variable.

**Method 2 - Without a FISHEYE_INST Directory**

If you are not using FISHEYE_INST, you will need to copy some files from your old Crucible installation to your new one.

1. Extract the new Crucible archive into a directory such as /NEW_FISHEYE/.
2. Delete the /NEW_FISHEYE/var directory.
3. Shut down the old Crucible instance if it is running.
4. Copy /OLD_FISHEYE/config.xml to /NEW_FISHEYE/.
5. Copy (or move) the /OLD_FISHEYE/var directory to /NEW_FISHEYE/var.
6. If you have a Cenqua-issued Crucible license, copy all /OLD_FISHEYE/*.license files to /NEW_FISHEYE/. (Atlassian-issued licenses are included within config.xml.)
7. Follow any version-specific instructions found in the Release Notes.

**Method 3 - Without a FISHEYE_INST Directory, but would like to set one up**

1. Shut down the old FishEye instance if it is running.
2. Set up the FISHEYE_INST environment variable, then create the FISHEYE_INST directory on your filesystem.
3. Copy the /OLD_FISHEYE/config.xml to /FISHEYE_INST.
4. Copy the /OLD_FISHEYE/var directory to /FISHEYE_INST.
5. Download Crucible.
6. Extract the new Crucible archive into a directory such as /NEW_FISHEYE/.
7. Start Crucible from the new installation by running NEW_FISHEYE/bin/run.sh. (Use run.bat on Windows.)
8. Follow the initial configuration steps outlined below.
9. If your configuration is not automatically picked up and you cannot see your existing repositories, check your Administration > Sys-Info page, where you will see information about FISHEYE_HOME and FISHEYE_INST. Check your FISHEYE_INST is pointing to the right directory.

**Crucible 2.0 Beta Reviewer’s Guide**

**Crucible 2.0 Beta** is a public development release leading up to Crucible 2.0. For all production use and testing of Crucible, please use the latest official release.
Do not use in production.
Beta releases should not be used in production environments.

Please also take note of the following information:

- Beta releases are not safe — Beta releases are snapshots of the ongoing Crucible development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.
  - FireFox 3 and Safari are the only browsers supported.

Thank you for your interest in the Crucible 2.0 Beta. This page contains some direction on what is ready for testing, what the known issues are and how you can submit feedback.

**Known Issues**

This is a list of known issues with the Crucible 2.0 Beta; please do not raise requests related to these as solutions for them are already under way.

- REST API updates are not fully functional; you can't access various features from the IDE Connectors or other technology that depends on REST. The features affected include marking comments as read, Iterative Reviews and Activity Streams. Also, JSON support is alpha at this stage.
- JIRA sub-tasks integration will not work on JIRA instances where the JIRA General Setting "allow unassigned issues" is set to 'OFF'.
- Sorting functionality is only partially functional; when you are trying to sort content — such as review lists on the reviews page, content on the dashboard and project dashboard — the sorting of content by name, date and so on only applies to the current page of results. To mitigate this, narrow down your searches so that you only have one page of results. This will be revised to have server-side sorting.
- You can't create reviews with files or changesets from multiple repositories.

**Features Ready For Testing**

The following features in the Crucible 2.0 Beta are relatively hardened and using these thoroughly will help contribute to the final product.

- Iterative Reviews; create a review on multiple revisions of a changeset, to see the history and evolution of the code in-line.
- Activity Streams; see review activity, code commits and JIRA activity (when properly configured) on the Dashboard.
- External Database Support; You can now store Crucible's internal data in a MySQL or PostgreSQL database, as an alternative to the built-in HSQLDB.
- Crucible can assist you in updating reviews by suggesting changesets that are likely related to your reviews.
- Reviews can now have a Due Date. Reviews that are overdue will show up in red on the reviewer's dashboards to minimise the chance of reviews getting stale.
- Stars; add colleagues, reviews and files to your favourites list, then view updates related to them as a feed.
- Charlietars; the automatically generated Crucible avatars should work smoothly. Also, you can sign up to Globally Recognised Avatars (http://www.gravatar.com) to upload a profile image and use that instead of the Charlie image.
- Scheduled Backups; you can now easily set Crucible to backup data periodically using a simple calendar function in the user interface.

**Submitting feedback**

To submit feedback on the Crucible 2.0 Beta, please use the Crucible Forums.

**JIRA Integration in Crucible 2.0 Beta**

**Crucible 2.0 Beta** is a public development release leading up to **Crucible 2.0**. For all production use and testing of Crucible, please use the latest official release.

Do not use in production.
Beta releases should not be used in production environments.
Please also take note of the following information:

- Beta releases are not safe — Beta releases are snapshots of the ongoing Crucible development process. As such:
  - While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
  - Features in development releases may be incomplete, or may change or be removed before the next full release.

This page contains instructions for setting up JIRA integration in Crucible.

- JIRA is Atlassian’s issue tracking product, which can be used to manage projects and associated work.

- Before you begin: Ensure that you configure your JIRA instance to enable sub-tasks, enable unassigned issues and allow Remote API access. The instructions on this page have been tested with JIRA 3.13.4.

On this page:

- Opening the Administration Screen for JIRA Integration
- Adding a New JIRA Server
  - Obtaining the Subtask Type ID
  - Obtaining the Subtask Resolution ID and Subtask Resolution Action ID
- Obtaining the Subtask ID
- Editing Default JIRA Server Mappings
- Operations on Existing Servers
  - Edit settings for an existing JIRA server
  - Edit mappings for an existing JIRA server
  - Delete an existing JIRA server

JIRA issues can be viewed in the main Dashboard view in Crucible. This requires you to enter details on the required JIRA server(s) via the Crucible administration screens.

Opening the Administration Screen for JIRA Integration

To set up JIRA integration, open the Administration screen and then click ‘JIRA Servers’ under the ‘Global Settings’ sub-menu on the left navigation bar. The ‘View JIRA Servers’ administration page opens.

Screenshot: The View JIRA Servers Page

On the View JIRA Servers page, you can carry out a number of operations as listed on this page.

Adding a New JIRA Server

To add a new JIRA server from the View JIRA Servers page, click ‘Add JIRA Server’.

The ‘Add JIRA Server’ page opens.

Screenshot: The Add JIRA Server Page
A number of fields and options must be filled out or selected on this page. See the table below for information on each field.

<table>
<thead>
<tr>
<th>Option</th>
<th>Type</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text Field</td>
<td>A descriptive name for the JIRA server.</td>
<td>Yes</td>
</tr>
<tr>
<td>URL</td>
<td>Text Field</td>
<td>The Internet address of the JIRA server.</td>
<td>Yes</td>
</tr>
<tr>
<td>Subtask Type ID</td>
<td>Number</td>
<td>This is required to enable creating issues from a Crucible comment.</td>
<td>No</td>
</tr>
<tr>
<td>Subtask Resolution Action ID</td>
<td>Number</td>
<td>This is required to enable creating issues from a Crucible comment. See below for instructions on obtaining this number.</td>
<td>No</td>
</tr>
<tr>
<td>Subtask Resolution ID</td>
<td>Number</td>
<td>This is required to enable creating issues from a Crucible comment. See below for instructions on obtaining this number.</td>
<td>No</td>
</tr>
<tr>
<td>Allow Unassigned</td>
<td>True/False Button</td>
<td>Allow unassigned sub-tasks.</td>
<td>No</td>
</tr>
<tr>
<td>Username</td>
<td>Text Field</td>
<td>The username of an account on the JIRA instance (All activity that takes place will be attributed to this user, unless using the Trusted Application setting).</td>
<td>Yes</td>
</tr>
<tr>
<td>Password</td>
<td>Text Field</td>
<td>The password for the account on the JIRA instance.</td>
<td>Yes</td>
</tr>
<tr>
<td>Include in Activity Streams</td>
<td>Check Box</td>
<td>Allows JIRA information to appear on the Dashboard.</td>
<td>No</td>
</tr>
<tr>
<td>Authenticate as Trusted Application</td>
<td>Check Box</td>
<td>Allows the system to interface with JIRA and let users log on with their own accounts (and use their own accounts on the JIRA server. See complete FishEye documentation and complete JIRA documentation.</td>
<td>No</td>
</tr>
</tbody>
</table>

Once you’ve filled out the necessary fields, click 'Test' to ensure that your details are correct. If you have a positive message return from the test, click 'Save'.

**Obtaining the Subtask Type ID**
This value is required (along with the Subtask Resolution ID and Subtask Resolution Action ID) to enable creating issues from a Crucible comment. This is the subtask type that will be created when you create a JIRA subtask in Crucible.

To obtain this value, carry out the following steps.

1. Enable sub-tasks on your JIRA instance from the ‘JIRA Administration’ > ‘Sub-Tasks’ page. See the JIRA documentation for details on this step.
2. Return to the FishEye Administration screen and then click ‘JIRA Servers’ under the ‘Global Settings’ sub-menu on the left navigation bar. Click ‘Edit’ next to the JIRA server you have configured.
3. Your JIRA server's basic details should appear. Click ‘Test’ once again. The field for Subtask Type ID will change to a drop-down menu, showing the available subtask types. Choose the correct one.
4. Save your Crucible configuration settings.

Screenshot: Obtaining the Subtask Type ID

Obtaining the Subtask Resolution ID and Subtask Resolution Action ID

These values are required (along with the Subtask Type ID) to enable creating issues from a Crucible comment.

To obtain these values, carry out the following steps.

1. Open your JIRA instance and go to ‘Administration’ > ‘Workflows’. The ‘Workflows’ screen opens. By default, the ‘JIRA’ workflow is shown on screen in a table.
2. Click the ‘Steps’ link in the far right table cell. The ‘View Workflow Steps — JIRA’ page opens.
3. The ‘Subtask Resolution Action ID’ is in the ‘Open’ row, under the ‘Transitions’ column. Look at the link in that cell named ‘Resolve Issue’. The ID number is shown in brackets next to that heading ‘Resolve Issue’ (shown in the screenshot below as 5).
4. Save your Crucible configuration settings.
5. The ‘Subtask Resolution ID’ is the ‘Resolved ID’ on this page. The ID number is shown in brackets next to the heading ‘Resolved’ (shown in the screenshot below as ‘4’). Note it down and enter it into the Crucible configuration screen.
6. Save your Crucible configuration settings.

Screenshot: Obtaining the Subtask Resolution ID & Subtask Resolution Action ID
Editing Default JIRA Server Mappings

This setting enables the Crucible feature that shows JIRA information in a dynamic window when you hover the mouse over a JIRA issue key in Crucible. It will also turn every issue key into a hyperlink to that issue in Crucible.

To enable this feature, click 'Edit Default JIRA Server Mappings' from the View JIRA Servers page. The 'Map JIRA Project Default' page opens.

On this page, select the FishEye repositories or Crucible Projects that you wish to associate with all the JIRA servers you have configured for use in Crucible. You can click 'add all' to quickly include them all in this category. You can remove individual items by clicking the small 'X' marks.
Once you've finished, click 'Save'.

⚠ You should disable any existing Crucible linkers you have set up for JIRA, as they will override this feature and prevent the dynamic dialog box from appearing when you mouse over an issue.

**Operations on Existing Servers**

Once you have configured an existing JIRA server, there are three main operations you can carry out on it: 'Edit', 'Mappings' and 'Delete'. These options appear on the far right of the screen.

*Screenshot: Operations in the JIRA Servers Page*

<table>
<thead>
<tr>
<th>Name</th>
<th>Url</th>
<th>Subtask ID</th>
<th>Subtask Resolution Action ID</th>
<th>Subtask Resolution ID</th>
<th>Allow Unassigned</th>
<th>Activity Stream Provider</th>
<th>Trusted Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>jac</td>
<td><a href="https://jira.atlassian.com">https://jira.atlassian.com</a></td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

*Edit settings for an existing JIRA server*

When you click 'Edit', you can adjust any of the general settings you configured when you first added the server.

*Edit mappings for an existing JIRA server*

When you click 'Mappings', a page is loaded that is almost identical to the 'Default Mapping' screen, but allows you to choose mappings only for that specific JIRA server.

*Delete an existing JIRA server*

Clicking 'Delete' will remove the server from the list.

**Crucible 1.6 Release Notes**

⚠ Crucible 2.5 has now been released. Read the Release Notes.

23 September 2008

**Atlassian presents Crucible 1.6**

**Crucible release 1.6** makes it easier to review content that is not in FishEye. Furthermore, Crucible 1.6 can be deployed without FishEye for the first time. Through Crucible’s new 'Light SCM' plugins, you can include content in reviews that are not associated with FishEye or even a source control repository. For example, you can review pages directly from Confluence, files on any file system connected to the machine FishEye is running on, and Subversion repositories not connected to FishEye. Crucible now has better support for uploading files for pre-commit review, in addition to the existing support for patches.

**Highlights of this release:**

- Support for Non-FishEye Repositories
- Confluence Page Reviews
- Shared File System Repositories
- Enhanced Pre-commit Reviews & Image Support
- Multiple Admin Users
- Expanded API
- Plus numerous improvements and bug fixes

**DOWNLOAD latest version**

**Upgrading to Crucible 1.6**

You can now download Crucible from here. If upgrading from a previous version, please follow the Upgrade Guide.
Highlights of Crucible 1.6

1. Support for Non-FishEye Repositories

Crucible can now be deployed as a stand-alone application for the first time. With Crucible 1.6, you no longer need a FishEye license or even a source-control repository. Crucible's new Light SCM plugin infrastructure already supports Confluence, server file systems, and Subversion repository types. We will be adding Git and ClearCase in the near future. The Light SCM interface is public and the shipped plugins are open source. As a result, you can extend these plugins or even write your own — great news for plugin developers.

2. Confluence Page Reviews

Crucible 1.6 allows you to select Atlassian Confluence as a source of material for reviews. In this way, you can use Crucible to easily review the Wiki Markup of pages created in Confluence. Read more.

3. Shared File System Repositories

You can create a 'repository' for a local or remote directory on the server file system. Teams that are managing documents through a shared file system instead of a source control system can still benefit from peer reviews. Read more.
Enhanced Pre-commit Reviews & Image Support

In addition to Crucible's patch support, 1.6 enables any file to be uploaded for review. The new upload functionality enables two files to be uploaded and compared in the review window, with diff highlighting. Read more.

Crucible now supports before and after inline image previews.

(Screenshot: Crucible Review of Uploaded Files)
Multiple Admin Users

Crucible now allows the Administrator to grant other users administration privileges. Admin Users can be individually assigned or given privileges through local or remote directory groups. Read more.

Expanded API

The Crucible API now allows programmable review creation, along with a host of other additions. Read more.

Plus numerous improvements and bug fixes

<table>
<thead>
<tr>
<th>JIRA Issues (110 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-2348</td>
</tr>
<tr>
<td>CRUC-668</td>
</tr>
<tr>
<td>CRUC-664</td>
</tr>
<tr>
<td>CRUC-663</td>
</tr>
<tr>
<td>CRUC-662</td>
</tr>
<tr>
<td>CRUC-661</td>
</tr>
<tr>
<td>CRUC-659</td>
</tr>
<tr>
<td>CRUC-654</td>
</tr>
<tr>
<td>CRUC-647</td>
</tr>
<tr>
<td>CRUC-646</td>
</tr>
<tr>
<td>CRUC-645</td>
</tr>
<tr>
<td>CRUC-640</td>
</tr>
<tr>
<td>CRUC-639</td>
</tr>
<tr>
<td>CRUC-635</td>
</tr>
</tbody>
</table>
CRUC-634  ReviewData should not include repoName  Closed
CRUC-632  Help text not updated after saving a review...  Closed
CRUC-626  Debug Logging Documentation  Closed
CRUC-623  SvnChangeLogBrowser.listChanges() in LiteSVN fails when processing revisions that are above the repo's configured base path  Closed
CRUC-622  Improve performance of querying for review details  Closed
CRUC-615  Make sure everything has a help link  Closed
CRUC-610  Removing revisions from review that have draft comments fails without an explanation  Closed
CRUC-608  Show a list of LightSCM repositories in admin  Closed
CRUC-602  Improve the efficiency of checking whether a review has been completed by all reviewers  Closed
CRUC-600  REST API should return error when non-existent filter used  Closed
CRUC-598  Emails always include a "Summary" section  Closed
CRUC-597  Extend the Remote API to allows reviews to be deleted  Closed
CRUC-594  Review description preview on dashboard swallows new lines from the review description  Closed
CRUC-592  Filter names used in Cru do not match the menus  Closed
CRUC-591  renaming a confluence repository adds a new one instead of editing  Closed
CRUC-589  Change revisionData getAdded/Removed lines an Integer  Closed
CRUC-588  Search Subtab in Review->Manage Files throws PropertyNotFoundException for all searches  Closed
CRUC-586  Remove Delete option from project drop down  Closed
CRUC-579  Changing repository when certain review tabs are selected causes NPE/bad redirect  Closed
CRUC-574  Allowed Review Participants left blank means what?  Resolved
CRUC-566  lightscm package should not be "fisheye"  Closed
CRUC-565  need get bulk version of getRevisionData() in SCMRepository  Closed
CRUC-563  Toggle side-by-side and show per frx diff options on review page  Closed
CRUC-562  REST API: createReview does not set Moderator and author correctly  Closed
CRUC-561  anchors to some comment types don't work  Closed
| CRUC-560 | REST-API: provide commit type information - esp. deleted file status | Closed |
| CRUC-556 | Next/previous comment arrows do not jump between file and general comments | Closed |
| CRUC-554 | Upgrade to non-beta atlassian-plugins | Closed |
| CRUC-551 | Confluence Light SCM plugin dependencies belong in plugin | Closed |
| CRUC-549 | Adjust IFRAME size when the admin page is resized | Closed |
| CRUC-548 | REST API: Handle allowReviewersToJoin flag on review | Closed |
| CRUC-547 | Make Repository source check if the engine is available. | Closed |
| CRUC-545 | NPE Closing Review | Closed |
| CRUC-542 | Use content manager to retrieve detailed file revisions | Closed |
| CRUC-540 | make metrics-config.xsd available online | Closed |
| CRUC-539 | Serious: "Next comment" does not always work right | Closed |
| CRUC-526 | Space character in installation path to Crucible disables remote API | Closed |
| CRUC-519 | Added files display "no change" | Closed |
| CRUC-518 | Crucible does not seem to be incrementally updating the FishEye cache | Closed |
| CRUC-516 | REST API: Return detailed information on review in one call - items, comments, reviewers, available actions | Closed |
| CRUC-511 | Added files when stored show "No Change" | Closed |
| CRUC-507 | Confluence: implement change set paging | Closed |
| CRUC-505 | Add "Search for Review" functionality to Crucible remote API | Closed |
| CRUC-504 | Confluence: get revision comments | Closed |
| CRUC-499 | Create separate API jar | Closed |
| CRUC-496 | Document store revisions with review | Closed |
| CRUC-494 | Don't make source type visible in URLs or elsewhere | Closed |
| CRUC-488 | clicking summarise without clicking close makes the review 'disappear' from the workflow | Closed |
| CRUC-487 | Page title says "dead" for a review that has been abandoned | Closed |
| CRUC-484 | you can link a review to itself | Closed |
| CRUC-479 | A comment which was a defect, but is no longer, still shows the defect attributes | Closed |
| CRUC-478 | Ability to specify that Branches are not supported | Closed |
| CRUC-472 | Changing tab while editing review deletes details | Closed |
| CRUC-469 | Sort list of repositories alphabetically on "Add project" page | Closed |
| CRUC-461 | Line comments are sometimes rendered twice | Closed |
| CRUC-457 | show authorname next to revision in revision dropdowns | Closed |
| CRUC-456 | crucible doesn't create default permission scheme when creating a blank db | Closed |
| CRUC-455 | creating a review from a changeset with a non-existent repo keeps redirecting to the login screen | Closed |
| CRUC-453 | Allow adding a screenshot as an attachment to a review | Closed |
| CRUC-439 | Invite to review | Closed |
| CRUC-438 | Not sending email notifications after Crucible and Fisheye upgrade | Closed |
| CRUC-436 | create 'remove all revision' link for all tabs under 'manage files' | Closed |
| CRUC-433 | REST API: Get review list based on predefined and custom filters | Closed |
| CRUC-422 | Create Schema Upgrade to remove foreign key constraint as reqd. by delete project work | Closed |
| CRUC-419 | Typo in email "reviewers are now complete" | Closed |
| CRUC-407 | Minimal OSGi Infrastructure | Closed |
| CRUC-404 | Turn FE Off When only Crucible licence is present | Closed |
| CRUC-402 | Provide Plugin Authors with somewhere to store properties | Closed |
| CRUC-401 | Real Admin Users | Closed |
| CRUC-400 | Add Configuration UI to SVN Plugin, and Polish | Closed |
| CRUC-399 | Optionally Store Files for all FileRevisions | Closed |
| CRUC-397 | Stored FRX Create UI | Closed |
| CRUC-396 | Stored FRX Data Impl | Closed |
| CRUC-394 | Add Light SCM Revisions to Reviews | Closed |
| CRUC-392 | Filesystem Light SCM Plugin | Closed |
CRUC-391 Default Repository for a Project can be a Light SCM repo
CRUC-389 Light SCM Plugin instances appear in source/repository dropdowns
CRUC-388 Subversion LSCM plugin
CRUC-387 Confluence LSCM Plugin
CRUC-386 Modify File Browser to Use Light SCM Plugins
CRUC-385 Modify Changeset Browser to use Light SCM Plugins
CRUC-384 Create Light SCM Module Type
CRUC-381 Admin Pages for Plugins
CRUC-373 Add General Comment via REST API
CRUC-362 Return replies to comments via API
CRUC-361 Project RSS feed
CRUC-349 REST method /rest-service/reviews-v1/CR-1/comments returns HTTP 500 error
CRUC-303 Add REST method to retrieve all reviews which involve a given file
CRUC-302 Add REST methods to allow review creation
CRUC-297 Upload files for review (not patches)
CRUC-292 javax.xml.bind.JAXBException generated when invoking the getGeneralComments web service method
CRUC-244 Add revisions not diffs to a review
CRUC-227 Cannot delete projects
CRUC-186 Allow an abandoned review to be deleted
CRUC-150 Allow the email address in the from section of the notification emails to be the user's actual email address
CRUC-18 Load All Users from Crowd/LDAP/etc

Crucible 1.6 Changelog

This page contains information about the Crucible 1.6 minor releases. FishEye license holders should also check the FishEye 1.6 Changelog.

⚠️ Please read the Crucible 1.6 Upgrade Guide before upgrading to any of the minor releases below.

On this page:
10 February 2009

This release updates the supporting libraries for Crucible plugins. This change enables the use of the new Git Crucible plugin for performing code reviews against a Git repository.

The Git plugin is not currently bundled with Crucible but may be downloaded from the Atlassian Maven repository here: https://maven.atlassian.com/browse/com.atlassian.crucible.plugins/crucible-git-scm-plugin/1.0

The Git plugin should be considered an early access release. It allows reviews to be performed against a local Git repository clone. Note that the plugin does not update the cloned repository automatically. For more information on the Git plugin, please see the documentation.

We are very interested in any feedback users have on the Git Crucible plugin. Please post feedback in the Crucible forums.

Full list of issues fixed in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (2 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-1406</td>
</tr>
<tr>
<td>CRUC-900</td>
</tr>
</tbody>
</table>

22 December 2008

This release contains a number of improvements and bug fixes.

Full list of issues fixed in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (6 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-947</td>
</tr>
<tr>
<td>CRUC-938</td>
</tr>
<tr>
<td>CRUC-928</td>
</tr>
<tr>
<td>CRUC-886</td>
</tr>
<tr>
<td>CRUC-867</td>
</tr>
<tr>
<td>CRUC-683</td>
</tr>
</tbody>
</table>

20 November 2008

This release contains bug fixes and minor improvements, and includes the new plugin points developed for AtlasCamp 2008.

Full list of issues fixed in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (27 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-926</td>
</tr>
<tr>
<td>CRUC-862</td>
</tr>
<tr>
<td>CRUC-846</td>
</tr>
<tr>
<td>CRUC-844</td>
</tr>
<tr>
<td>CRUC-837</td>
</tr>
<tr>
<td>CRUC-835</td>
</tr>
<tr>
<td>CRUC-834</td>
</tr>
<tr>
<td>CRUC-832</td>
</tr>
<tr>
<td>CRUC-827</td>
</tr>
<tr>
<td>CRUC-826</td>
</tr>
<tr>
<td>CRUC-821</td>
</tr>
<tr>
<td>CRUC-820</td>
</tr>
<tr>
<td>CRUC-809</td>
</tr>
<tr>
<td>CRUC-802</td>
</tr>
<tr>
<td>CRUC-793</td>
</tr>
<tr>
<td>CRUC-788</td>
</tr>
<tr>
<td>CRUC-787</td>
</tr>
<tr>
<td>CRUC-778</td>
</tr>
<tr>
<td>CRUC-765</td>
</tr>
<tr>
<td>CRUC-753</td>
</tr>
<tr>
<td>CRUC-748</td>
</tr>
<tr>
<td>CRUC-727</td>
</tr>
<tr>
<td>CRUC-582</td>
</tr>
<tr>
<td>CRUC-568</td>
</tr>
<tr>
<td>CRUC-398</td>
</tr>
<tr>
<td>CRUC-328</td>
</tr>
</tbody>
</table>
From 1.6.2.1 to 1.6.3

5 November 2008

This release rolls together several improvements and bug fixes.

- Auto-save draft comments.
- Performance improvements when using Light SCM repositories.
- Bundle a Perforce Light SCM implementation.
- Various REST API improvements, including Conditional-Get support, improved error handling and revised review searching, which now allows any criteria to be omitted.
- JSON serialization has been added to the REST API, allowing the use of JSON in REST API calls. This feature is in an experimental state at present. Please report any issues discovered.

Please be aware of the upgrade notes regarding Light SCM repositories (this does not impact FishEye repositories):

- The configuration storage for the bundled File-system, Confluence and Subversion Light SCM plugins changed. Once you have upgraded to 1.6.3 you will need to re-add those repositories. Please read the Upgrade Notes in the Crucible 1.6 Upgrade Guide.
- The Light SCM plugin API was changed in this release. Light SCM plugins compiled against the old API will not work in this release of Crucible.

Full list of issues fixed in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (78 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>CRUC-785</td>
</tr>
<tr>
<td>CRUC-784</td>
</tr>
<tr>
<td>CRUC-777</td>
</tr>
<tr>
<td>CRUC-776</td>
</tr>
<tr>
<td>CRUC-775</td>
</tr>
<tr>
<td>CRUC-773</td>
</tr>
<tr>
<td>CRUC-772</td>
</tr>
<tr>
<td>CRUC-771</td>
</tr>
<tr>
<td>CRUC-769</td>
</tr>
<tr>
<td>CRUC-768</td>
</tr>
<tr>
<td>CRUC-767</td>
</tr>
<tr>
<td>CRUC-764</td>
</tr>
<tr>
<td>CRUC-763</td>
</tr>
<tr>
<td>CRUC-759</td>
</tr>
<tr>
<td>CRUC-758</td>
</tr>
<tr>
<td>CRUC-757</td>
</tr>
<tr>
<td>CRUC-752</td>
</tr>
<tr>
<td>CRUC-750</td>
</tr>
<tr>
<td>CRUC-749</td>
</tr>
<tr>
<td>CRUC-747</td>
</tr>
<tr>
<td>CRUC-744</td>
</tr>
<tr>
<td>CRUC-742</td>
</tr>
<tr>
<td>CRUC-739</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>CRUC-736</td>
</tr>
<tr>
<td>CRUC-735</td>
</tr>
<tr>
<td>CRUC-733</td>
</tr>
<tr>
<td>CRUC-729</td>
</tr>
<tr>
<td>CRUC-728</td>
</tr>
<tr>
<td>CRUC-725</td>
</tr>
<tr>
<td>CRUC-724</td>
</tr>
<tr>
<td>CRUC-723</td>
</tr>
<tr>
<td>CRUC-722</td>
</tr>
<tr>
<td>CRUC-713</td>
</tr>
<tr>
<td>CRUC-712</td>
</tr>
<tr>
<td>CRUC-711</td>
</tr>
<tr>
<td>CRUC-710</td>
</tr>
<tr>
<td>CRUC-709</td>
</tr>
<tr>
<td>CRUC-708</td>
</tr>
<tr>
<td>CRUC-706</td>
</tr>
<tr>
<td>CRUC-705</td>
</tr>
<tr>
<td>CRUC-702</td>
</tr>
<tr>
<td>CRUC-701</td>
</tr>
<tr>
<td>CRUC-700</td>
</tr>
<tr>
<td>CRUC-699</td>
</tr>
<tr>
<td>CRUC-698</td>
</tr>
<tr>
<td>CRUC-697</td>
</tr>
<tr>
<td>CRUC-696</td>
</tr>
<tr>
<td>CRUC-695</td>
</tr>
<tr>
<td>CRUC-694</td>
</tr>
<tr>
<td>CRUC-693</td>
</tr>
<tr>
<td>CRUC-691</td>
</tr>
<tr>
<td>CRUC-689</td>
</tr>
<tr>
<td>CRUC-688</td>
</tr>
<tr>
<td>CRUC-687</td>
</tr>
<tr>
<td>CRUC-680</td>
</tr>
<tr>
<td>CRUC-679</td>
</tr>
<tr>
<td>CRUC-678</td>
</tr>
<tr>
<td>CRUC-677</td>
</tr>
<tr>
<td>CRUC-671</td>
</tr>
<tr>
<td>CRUC-670</td>
</tr>
<tr>
<td>CRUC-667</td>
</tr>
<tr>
<td>CRUC-665</td>
</tr>
<tr>
<td>CRUC-658</td>
</tr>
<tr>
<td>CRUC-656</td>
</tr>
<tr>
<td>CRUC-655</td>
</tr>
<tr>
<td>CRUC-653</td>
</tr>
<tr>
<td>CRUC-627</td>
</tr>
<tr>
<td>CRUC-624</td>
</tr>
<tr>
<td>CRUC-536</td>
</tr>
<tr>
<td>CRUC-529</td>
</tr>
<tr>
<td>CRUC-509</td>
</tr>
<tr>
<td>CRUC-486</td>
</tr>
<tr>
<td>CRUC-475</td>
</tr>
<tr>
<td>CRUC-470</td>
</tr>
<tr>
<td>CRUC-395</td>
</tr>
<tr>
<td>CRUC-383</td>
</tr>
<tr>
<td>CRUC-380</td>
</tr>
<tr>
<td>CRUC-327</td>
</tr>
</tbody>
</table>

**From 1.6.2 to 1.6.2.1**

**24 October 2008**

This release fixes a problem in 1.6.2 when running Crucible on Windows. Due to a file-lock issue, the upgrade script in 1.6.2 could not start.

- CRUC-781 Upgrade from 1.6 to 1.6.2 fails for windows machines.

**From 1.6.1 to 1.6.2**

**21 October 2008**

This release fixes a bug in the way Crucible stores review data. This bug was introduced in Crucible 1.6.0. **We strongly recommend all 1.6.0, 1.6.0-beta and 1.6.1 users immediately upgrade to this release.**

If this bug occurs in your Crucible instance, you will find that review data created after that point will be corrupt. If you find that is the case please contact Crucible support for assistance.

- CRUC-743 Switch from CACHED tables mode back to memory table.

**From 1.6.0 to 1.6.1**

**24 September 2008**

This is a bug fix release.

- Crowd 1.3 users will need to upgrade to Crowd 1.4.4 or later due to an incompatibility with this version of Crucible.
- CRUC-673 NPE when viewing a review.
- CRUC-674 NPE when closing a review.
Crucible 2.4 Documentation

Crucible 1.6 Upgrade Guide

Below are some important notes on upgrading to Crucible 1.6. For details of the new features and improvements in this release, please read the Crucible 1.6 Release Notes and Crucible 1.6 Changelog.

On this page:

- Upgrade Notes
  - Crucible 1.6.3 Upgrade Notes
  - Upgrade Procedure
  - Checking for Known Issues and Troubleshooting the Crucible Upgrade

Upgrade Notes

Crucible 1.6.3 Upgrade Notes

- The configuration storage for the bundled File-system, Confluence and Subversion Light SCM plugins changed. Please follow the procedure below.
- The Light SCM plugin API was changed in this release. Light SCM plugins compiled against the old API will not work in this release of Crucible.

Crucible 1.6.3 Upgrade Procedure:

Due to a configuration change, you will need to delete and add your LightSCM repositories again:

1. Before you shut down Crucible, take a note of your Light SCM configuration. You can view this configuration in the 'Repository List' Administration page, in the 'LightSVN', 'Confluence' and 'File System' sections.
2. Follow the general instructions on upgrading Crucible.
3. While Crucible is shut down, delete the confluence, svn and filesystem config files in FISHEYE_INST/var/plugins/user.
4. Once Crucible has been restarted, add the Light SCM repositories from step 1 back in again.

Upgrade Procedure

Upgrade a test environment first
As always, please test your upgrades in your test environment before rolling into production.

If you are already running a version of Crucible, please follow these instructions on Upgrading to a New Version of Crucible.

Checking for Known Issues and Troubleshooting the Crucible Upgrade

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- Check for known issues. Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the Crucible Knowledge Base for known issues and follow the instructions to apply any necessary patches if necessary.

- Did you encounter a problem during the Crucible upgrade? Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

RELATED TOPICS

Crucible 1.6 Release Notes
Crucible 1.6 Changelog

Crucible 1.5 Release Notes

Crucible 2.5 has now been released. Read the Release Notes.

14 April 2008
Atlassian presents Crucible 1.5

Crucible release 1.5 brings new enhancements that make your code review activities quicker and easier. The all-new per-project page consolidates the display of work done on a particular goal or product, while filtered search for defects and comments provides rapid access to Crucible content that you need to see, now.

Highlights of this release:

- Project Dashboard
- Filtered comments & defects search, with statistical summary
- Customisable email templates
- Improvements to Crucible Plugin API beta
- Plus numerous improvements and bug-fixes

Upgrading to Crucible 1.5

You can now download Crucible from here. If upgrading from a previous version, please follow the Upgrade Guide.

Highlights of Crucible 1.5

1

Project Dashboard

Crucible 1.5 introduces the Project Dashboard, which allows you to see open reviews that belong to a given project, presented with additional project-related data and graphs.

Screenshot: Crucible Project Dashboard
Filtered comments & defects search, with statistical summary

Defects and comments are now searchable, easing the difficulty of finding a particular piece of work or revision (and its relevant comments). These search results now also show a very useful statistical summary. Also, a new defect metrics report is available.

Screenshot: Crucible Defect Metrics Report

Customisable email templates

You can now customise the content and appearance of email notifications that get sent to Crucible users. For example you can append a legal...
disclaimer, alter the subject line or provide custom header text for all messages.

4

**Improvements to Crucible Plugin API beta**

Now with REST support and the ability to upload patches, the Crucible Plugin API beta is for Crucible integrators who want to extend Crucible to interoperate with their enterprise infrastructure or processes.

**Plus numerous improvements and bug-fixes**

<table>
<thead>
<tr>
<th>JIRA Issues (38 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-344 Defect pie chart on comment search page includes non-defect comments.</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-332 create a better looking indication icon</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-324 Add Project Dropdown to Search Pages</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-323 Create 'Reviews Started' Pie Chart for Project Dashboard</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-318 Make Project Dashboard Charts cover 14 days instead of 7</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-314 sidebyside diff line numbers must be aligned to the top</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-307 Optimize chart queries</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-299 link to project page</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-290 Warnings during backup: File does not exist /data/crucible/inst/var/data/crudb/crucible.lck &amp; File does not exist /data/crucible/inst/var/data/crudb/crucible.log</td>
<td>⬇</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-289 permissions are not checked in rss feeds (but are in list html display)</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-285 &quot;project project&quot; in Permission Screen in admin</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-283 Escape HTML in Javascript properly</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-282 Crucible fails to upload patch file with file information that contain blank spaces between file path and timestamp</td>
<td>⬇</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-280 remove debug log from Process Notification</td>
<td>⬆</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-266 let the subject of the email be modifiable via the notification template</td>
<td>⬇</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-260 How to customize the email notification content</td>
<td>⬇</td>
<td>Resolved</td>
</tr>
<tr>
<td>CRUC-248 Configurable email subject line</td>
<td>⬇</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-247 Flag files which have been modified since a review was created</td>
<td>⬇</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-246 Parentheses should be valid URL characters</td>
<td>⬇</td>
<td>Closed</td>
</tr>
</tbody>
</table>
CRUC-242 Add RPC call to create review from patch

CRUC-241 Incorrect dates in RSS feeds

CRUC-240 Checking box in "manage files" -> "changesets" spins forever

CRUC-238 Perforce unit tests do not work on Windows platform

CRUC-232 Address all issues outline here

CRUC-230 Crucible allows circular review linking and can't delete a review link

CRUC-226 Title wrong for post-approval in manage files tab

CRUC-220 permalink for a comment does not work when summarize mode is active

CRUC-212 Create Project Dashboard page

CRUC-206 URL recognition in Crucible comment does not always match whole string

CRUC-201 Allow project to be specified on create review URL

CRUC-194 Documentation: restoring Crucible backup data

CRUC-184 When Creating Review From FishEye, Default to Right Project

CRUC-172 Create 'Getting started with Crucible' document as part of User Guide revisions

CRUC-162 From field on emails is incorrect or a least deceiving

CRUC-147 Add Permissions Checks to All Operations on Reviews

CRUC-128 email notification of review closure prints literal "null" for absent field value

CRUC-85 Sometimes we need to see differences in UNICODE files.

CRUC-21 change icon for "View History In Fisheye"

**Crucible 1.5 Changelog**

This page contains information about the Crucible 1.5 minor releases. FishEye license holders should also check the FishEye 1.5 Changelog.

⚠️ Please read the Crucible 1.5 Upgrade Guide before upgrading to any of the minor releases below.

**On this page:**

- From 1.5.3 to 1.5.4
- From 1.5.2 to 1.5.3
- From 1.5.1 to 1.5.2
- From 1.5.0 to 1.5.1

**From 1.5.3 to 1.5.4**
1 August 2008

This release contains minor improvements and bug fixes.

<table>
<thead>
<tr>
<th><strong>JIRA Issues</strong> (10 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>CRUC-638</td>
</tr>
<tr>
<td>CRUC-633</td>
</tr>
<tr>
<td>CRUC-538</td>
</tr>
<tr>
<td>CRUC-513</td>
</tr>
<tr>
<td>CRUC-503</td>
</tr>
<tr>
<td>CRUC-498</td>
</tr>
<tr>
<td>CRUC-454</td>
</tr>
<tr>
<td>CRUC-429</td>
</tr>
<tr>
<td>CRUC-417</td>
</tr>
<tr>
<td>CRUC-284</td>
</tr>
</tbody>
</table>

From 1.5.2 to 1.5.3

23 June 2008

This release contains bug fixes.

<table>
<thead>
<tr>
<th><strong>JIRA Issues</strong> (1 issue)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>CRUC-462</td>
</tr>
</tbody>
</table>

From 1.5.1 to 1.5.2

27 May 2008

This release contains bug fixes.

<table>
<thead>
<tr>
<th><strong>JIRA Issues</strong> (16 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>CRUC-434</td>
</tr>
<tr>
<td>CRUC-413</td>
</tr>
<tr>
<td>CRUC-406</td>
</tr>
<tr>
<td>CRUC-405</td>
</tr>
<tr>
<td>CRUC-376</td>
</tr>
<tr>
<td>CRUC-371</td>
</tr>
<tr>
<td>CRUC-370</td>
</tr>
<tr>
<td>CRUC-363</td>
</tr>
<tr>
<td>CRUC-358</td>
</tr>
</tbody>
</table>
From 1.5.0 to 1.5.1

24 April 2008

This release contains bug fixes.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-367</td>
<td>LOC exceptions filling Debug log to the tune of 3-6GB per day</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-364</td>
<td>Starting Crucible 1.5 Spits Lots of Errors Out</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-357</td>
<td>Sparklines have wrong content type</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-353</td>
<td>Crucible seems not to be able to handle uploaded git patches</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-350</td>
<td>javax.servlet.ServletException</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-333</td>
<td>In IE, &quot;view diff to latest&quot; partially obscured</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-304</td>
<td>Add REST method to retrieve review items for a given review</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-291</td>
<td>NullPointerException thrown when invoking the getAllRevisionComments web service method</td>
<td>📊</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-161</td>
<td>Create document stating when a customer purchases Crucible that they do not need to have the latest version of FishEye</td>
<td>📊</td>
<td>Closed</td>
</tr>
</tbody>
</table>

**Crucible 1.5 Upgrade Guide**

Below are some important notes on upgrading to Crucible 1.5. For details of the new features and improvements in this release, please read the Crucible 1.5 Release Notes and the Crucible 1.5 Changelog.

On this page:

- Upgrade Procedure
- Checking for Known Issues and Troubleshooting the Crucible Upgrade

**Upgrade Procedure**

⚠️ Upgrade a test environment first
As always, please test your upgrades in your test environment before rolling into production.
If you are already running a version of Crucible, please follow these instructions on Upgrading to a New Version of Crucible.

Checking for Known Issues and Troubleshooting the Crucible Upgrade

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- **Check for known issues.** Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the Crucible Knowledge Base for known issues and follow the instructions to apply any necessary patches if necessary.

- **Did you encounter a problem during the Crucible upgrade?** Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

RELATED TOPICS

Crucible 1.5 Release Notes
Crucible 1.5 Changelog

**Crucible 1.2 Release Notes**

⚠️ Crucible 2.5 has now been released. Read the Release Notes.

**December 5, 2007**

The Atlassian Crucible team is delighted to present Crucible 1.2. **Crucible release 1.2** brings you a host of popular new features. You can now group your reviews into projects (similar to JIRA projects) and authorise your users via project permission schemes.

New user management screens make the administrator's job a lot easier. The new built-in integration with Atlassian Crowd extends your authentication and authorisation capabilities. You can now include users and groups from one or more Crowd directories, and provide single sign-on (SSO) across Atlassian products plus any other applications that support SSO.

Crucible's integration with JIRA and FishEye is now closer than ever before. Read the details below.

**Highlights of this release:**

- Reviews grouped into projects
- Customisable permission schemes
- Plugin API
- Enhancements to user management
- JIRA integration
- Crucible 1.2 includes FishEye 1.4
- Plus over 20 improvements and bug-fixes

**Responding to your feedback:**

🌟 8 new feature requests/improvements implemented
🌟 9 votes satisfied

Your [votes and issues](http://jira.atlassian.com/browse/CRUC) help us keep improving our products, and are much appreciated.

<table>
<thead>
<tr>
<th>Upgrading to Crucible 1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can now download Crucible from here. If upgrading from a previous version, please follow the Upgrade Guide.</td>
</tr>
</tbody>
</table>

**Highlights of Crucible 1.2**
Reviews grouped into projects

- Crucible now supports projects — every review will belong to a project.
- Each project has a unique key (prefix), modelled on JIRA keys.
- You can add your own projects via the new administration screens.
- You can specify review defaults per project, such as the default users for each role and a default repository.
- And you can restrict the users/groups who can perform a particular role, e.g. only team leaders can be moderators.
- Each project has its own permission scheme (see below).

Customisable permission schemes

- A permission scheme is a set of actions which a user can perform (e.g. create a review, approve a review, etc).
- Each project can have its own custom permission scheme — or you can use the same scheme for multiple projects.
- The permission scheme for a review is determined by the review’s project.
Plugin API

- A new plugin Crucible programming interface (API), in **beta** for this release, supports the following functionality:
  - Create or modify reviews and comments.
  - Add files, patches, etc to reviews.
  - Invoke state transitions.
  - Add custom servlet handlers.
- More information.

Enhancements to user management

In Crucible 1.1.2, we introduced support for public signup (self-registration). Now in Crucible 1.2:

- Administrator can make the email address for **self-signups** optional.
- Improved user interface makes user administration easier.
- **Groups** are supported.
- Read the FishEye documentation.

JIRA integration

The new version 1.2 of the FishEye-for-JIRA plugin includes some useful improvements:

- new 'FishEye' tab for JIRA issues and projects
- improved ability to create a Crucible review from the 'FishEye' tab within a JIRA issue — just click the Crucible icon.
Crucible 2.4 Documentation

Crucible 1.2 includes FishEye 1.4

... and provides closer integration than ever before.

- FishEye screens include links to existing Crucible reviews. So you can see which files/changesets have been reviewed.
- EyeQL allows you to search for Crucible data. For example, you can search for files that have not yet been reviewed.
- Crucible now has built-in Crowd/SSO support.
- See the FishEye 1.4 Release Notes.

Plus over 20 improvements and bug-fixes

<table>
<thead>
<tr>
<th>JIRA Issues (32 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-181</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-175</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-166</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-152</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-146</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-143</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-140</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-122</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-121</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-115</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-113</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-109</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-107</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-104</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-102</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-101</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-95</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-94</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-93</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>
Crucible 1.2 Changelog

This page contains information about the Crucible 1.2 minor releases.

Please read the Crucible 1.2 Upgrade Guide before upgrading to any of the minor releases below.

On this page:

- From 1.2.2 to 1.2.3
- From 1.2.1 to 1.2.2
- From 1.2 to 1.2.1

From 1.2.2 to 1.2.3

7 February 2008

This release contains bug fixes (including those in from FishEye 1.4.3).

### JIRA Issues (26 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-274</td>
<td>Create IDE integration with IntelliJ IDEA</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-273</td>
<td>Create IDE integration with Eclipse</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-258</td>
<td>Allow all users to be able to a &quot;Moderator&quot; or &quot;Author&quot; in a review project</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-256</td>
<td>Check boxes for checking off files reviewed in a review are lost if navigate to raw text by &quot;download raw text&quot; icon</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-249</td>
<td>Accept review comments in Japanese (unicode characters)</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-243</td>
<td>race condition between crucible backup and repository scan</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-235</td>
<td>File tabs should be kept open</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-210</td>
<td>Create JIRA issues from code reviews</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>Key</td>
<td>Summary</td>
<td>Priority</td>
<td>Status</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------------------------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td>CRUC-208</td>
<td>Nullpointer exception when creating new review</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-202</td>
<td>update ‘Repositories’ screenshot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-196</td>
<td>Error on attempting to create a Review.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-180</td>
<td>‘Show Full Source’ should continue to highlight the diff</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-177</td>
<td>Comments on reviews are being sent but don’t appear in the review when viewed online</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-165</td>
<td>Folder Issue</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-139</td>
<td>Can’t add new versions of the same file to a review</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-127</td>
<td>Improperly stopping crucible causes data loss!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-112</td>
<td>Point online help links to new Crucible doc space</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-106</td>
<td>Add content to new documentation page ’ Crucible and FishEye’</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-105</td>
<td>Upload PDF, XML and HTML versions of Crucible docs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-103</td>
<td>Create admin page for Projects</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-100</td>
<td>Move Crucible docs to Confluence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-99</td>
<td>Feature request: I would like to see statistics about number of major and minor issues found per developer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-92</td>
<td>Allow Anonymous Access Per Repository</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-84</td>
<td>Ability to create a review directly from JIRA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-50</td>
<td>Allow review of entire file, not a particular diff</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-28</td>
<td>After Crucible forums are moved, update all links in docs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

From 1.2.1 to 1.2.2

This release contains some minor improvements and bug fixes.

- **Trusted Application Support**
  FishEye/Crucible now allows you to set up trusted communications with other Atlassian applications. At this point, the JIRA FishEye plugin supports Trusted Applications. The JIRA FishEye plugin can request information from FishEye on behalf of the currently logged-in user, and FishEye will not ask the user to log in again or to supply a password. Previously FishEye/Crucible would have used a single ‘system’ account to determine permissions. Now, FishEye/Crucible can apply the correct permission settings for the logged-in user.

- **Hyphens are now allowed in project key names.**

### JIRA Issues (3 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-207</td>
<td>allow hyphens in Crucible project keys</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUC-204</td>
<td>Add project to dashboard filter</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Closed
From 1.2 to 1.2.1

This is a small bug-fix release.

**JIRA Issues** (6 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-199</td>
<td>Top of project edit form says Repositories List</td>
<td>1</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-198</td>
<td>multiple copies of revisions added to review if you add changesets in reverse chronological order</td>
<td>1</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-182</td>
<td>Projects Don't Persist Through Pages</td>
<td>1</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-179</td>
<td>Support addition of multiple revisions of the same file that appears in multiple changesets</td>
<td>1</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-178</td>
<td>Hit NPE when perform backup</td>
<td>1</td>
<td>Closed</td>
</tr>
<tr>
<td>CRUC-40</td>
<td>Getting wrong changeset number causes diff to be missing</td>
<td>1</td>
<td>Closed</td>
</tr>
</tbody>
</table>

**Crucible 1.2 Upgrade Guide**

**Upgrade Notes**

- During the upgrade, a default project and default permission scheme will be created. All existing reviews will be assigned to the default project.

**Upgrade Procedure**

- Please read the Release Notes for the version you are upgrading to, as well as any versions you are skipping.
- Follow the instructions on upgrading Crucible.

**Crucible 1.1 Release Notes**

⚠️ Crucible 2.5 has now been released. Read the Release Notes.

**18 September 2007**

Crucible 1.1 allows pre-commit (patch) reviews, side-by-side diff mode, syntax highlighting in diffs, and many other bug fixes and improvements.

**Upgrading Crucible**

You can now download Crucible from here. Information on installing Crucible can be found here. If upgrading from a previous version, please follow the Upgrade Guide.

**Highlights of Crucible 1.1**

- Pre-commit review (patch review).
- Progress tracking through a review by marking each file as ‘done’.
- Side-by-side diff mode within the review display.
- Syntax highlighting when displaying a diff.
- Many small UI fixes and improvements. Refer to the changelog for more details.

**Crucible 1.1 Changelog**

This page contains information about the Crucible 1.1 minor releases.

⚠️ Please read the Crucible 1.1 Upgrade Guide before upgrading to any of the minor releases below.
From 1.3.1 to 1.4

This release updates the included FishEye component and includes a number of performance improvements and bug fixes for Subversion and Perforce repository indexing.

From 1.1.2 to 1.3

This release fixes a bug CRUC-104 that prevented Crucible from correctly displaying large patches.

From 1.1.1 to 1.2

This release adds some new user-related functions and includes bug fixes.

New Features

- You can now allow users to create their own user accounts (sign-up).
- You can now allow anonymous browsing of reviews.
- Users can now add themselves as a reviewer (‘Join a review’). This is an option that is configured per review.

Bug fixes

- Fix problem where Crucible would only display the top part of each diff in a patch.
- Fix various JavaScript and UI errors.
- Fix various IE6 and IE7 problems.
- Fix problem where some users where redirected to /uar/browser.css after login.

From 1.1 to 1.1.1

This is a small bug-fix release. It addresses a stack-overflow problem for some configurations.

**Crucible 1.1 Upgrade Guide**

Upgrade Notes

- As of version 1.0, Crucible now requires a JVM version 1.5 or later. Previously, 1.4+ was required.
- Crucible 1.1.4 includes FishEye 1.3.8.
- Upgrading from 1.0.4 (or earlier) will force a complete re-index of P4 repositories.

Upgrade Procedure

- Please read the Release Notes and Upgrade Guides for the version you are upgrading to, as well as any versions you are skipping.
- Follow the instructions on upgrading Crucible.

**Crucible Release Summary**

**Crucible 2.5 (8-Feb-11)**

- Oracle
- Redesigned Activity Stream
- Improved Header
- Comment Notification Batching
- More in release notes.

**Crucible 2.4 (20-Oct-10)**

- Easier Application Linking
- Native Repository Access
- Starter Licenses
- Adding Changesets to Reviews Simplified
- User Interface Improvements
- Snippets Tweaks
- More in release notes.

**Crucible 2.3 (26-May-10)**
- Snippet Reviews
- Changeset Discussions
- Mercurial SCM Alpha
- Review Coverage report
- Revamped Installation Process
- Gadgets
- More in release notes.

**Crucible 2.2 (18-Feb-10)**

- Smart Pre-Commit (Patch) Support
- 'No Moderator' Reviews
- Wizard-Like Review Creation
- Integrated Timetracking Between Crucible and JIRA
- Edit Mode for Reviews
- More in release notes.

**Crucible 2.1 (12-Nov-09)**

- Wiki Markup Rendering
- Progress Tracking
- Usability and Productivity Updates
- Streamlined JIRA Integration
- Review Time Tracking
- Review History Dialog
- "Blockers" Reports
- Threaded Comments
- Plugin Developer Tools
- More in release notes.

**Crucible 2.0 (30-Jun-09)**

- Support for iterative reviews
- New User Interface
- Indicators for read/unread comments
- Enhanced JIRA integration
- More in release notes.

**Crucible 1.6 (23-Sep-08)**

- Support for non-FishEye repositories
- Confluence page reviews
- Shared file system repositories
- Enhanced pre-commit reviews & image support
- Multiple admin users
- Expanded API
- More in release notes.

**Crucible 1.5 (14-Apr-08)**

- Project Dashboard
- Filtered comments & defects search, with statistical summary
- Customisable email templates
- Improvements to Crucible Plugin API beta
- More in release notes.

**Crucible 1.2 (5-Dec-07)**

- Reviews grouped into projects
- Customisable permission schemes
- Plugin API
- Enhancements to user management
- JIRA integration
- Crucible 1.2 includes FishEye 1.4
- More in release notes.

**Crucible 1.1 (18-Sep-07)**

- Pre-commit review (patch review)
- Review participants can keep track of their progress through a review by marking each file as "done"
Security Advisories

This page lists security advisories for Crucible.

- Crucible Security Advisory 2010-05-04
- Crucible Security Advisory 2010-06-16
- FishEye and Crucible Security Advisory 2011-01-12

Crucible Security Advisory 2010-05-04

The 2.2.3 release of Crucible contains some security related fixes, which are part of the shared FishEye architecture. The following information for FishEye applies equally to Crucible.

The Crucible Download Centre has the updates for Crucible.

In this advisory:

- Admin Escalation Vulnerability
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- XSS Vulnerabilities in FishEye
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- Prevention of Brute Force Attacks
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- Changed Behaviour in FishEye
- Download Patches for Earlier FishEye / Crucible Versions
  - Patch for FishEye / Crucible 2.1.4
  - Patch for FishEye / Crucible 2.0.6
  - Patch for FishEye 1.6.6
  - Patch for Crucible 1.6.6

Admin Escalation Vulnerability

Severity

Atlassian rates this vulnerability as critical, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed an admin escalation vulnerability, which affects FishEye instances. This vulnerability has security implications and is especially important for anyone running publicly accessible instances of FishEye.

Vulnerability

This vulnerability allows a motivated attacker to perform admin actions.

All versions of FishEye from version 1.6.0-beta2 (including 1.6.0) through to 2.2.1 are affected by these admin escalation vulnerabilities.
Crucible 2.4 Documentation

Affected FishEye Versions | Fix Availability | More Details | Severity
---|---|---|---
All versions up to and including 2.2.1 | 2.2.3 update, also available as patches for certain versions, listed on this page. | This vulnerability allows a motivated attacker to perform admin actions. | Critical

Risk Mitigation

We strongly recommend either upgrading or patching your FishEye installation to fix this vulnerability. Please see the 'Fix' section below.

Note: If you are an Atlassian JIRA Studio customer, we have assessed that your system is secure and implemented additional protections for it.

Fix

These issues have been fixed in FishEye 2.2.3 (see the changelog), which you can download from the download centre. Later versions will include protection from this vulnerability.

This fix is also provided as a patch for FishEye 2.1.4, 2.0.6 and 1.6.6, which you can download from this page. Customers on earlier point versions of FishEye will have to upgrade to version 2.1.4, 2.0.6 or 1.6.6 before applying the patch. We recommend you upgrade to FishEye 2.2.3.

**XSS Vulnerabilities in FishEye**

Severity

Atlassian rates these vulnerabilities as critical, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed several cross-site scripting (XSS) vulnerabilities in FishEye, which may affect FishEye instances. These vulnerabilities have security implications and are especially important for anyone running publicly accessible instances of FishEye.

- The attacker might take advantage of the vulnerability to steal other users' session cookies or other credentials, by sending the credentials back to the attacker's own web server.
- The attacker's text and script might be displayed to other people viewing a FishEye page. This is potentially damaging to your company's reputation.

You can read more about XSS attacks at cgisecurity, CERT and other places on the web.

Vulnerability

All versions of FishEye are affected by these XSS vulnerabilities.

<table>
<thead>
<tr>
<th>Affected FishEye Versions</th>
<th>Fix Availability</th>
<th>More Details</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>All versions up to and including 2.2.1</td>
<td>2.2.3 only</td>
<td>An attacker could take advantage of this vulnerability to steal other users' session cookies or other credentials, or the attacker's text and script might be displayed to other people viewing a FishEye page.</td>
<td>Critical</td>
</tr>
</tbody>
</table>

Risk Mitigation

We strongly recommend upgrading your FishEye installation to fix these vulnerabilities. Please see the 'Fix' section below.

Fix

These issues have been fixed in FishEye 2.2.3 (see the changelog), which you can download from the download centre.
**Prevention of Brute Force Attacks**

**Severity**
Atlassian rates this vulnerability as *moderate*, according to the scale published in *Severity Levels for Security Issues*.

**Risk Assessment**
We have improved the security of the following areas in FishEye:

- Prevention of brute force attacks by requiring users to solve a CAPTCHA test after a maximum number of repeated login attempts.

**Vulnerability**
We have identified and fixed a problem where FishEye allows an unlimited number of repeated login attempts, potentially opening FishEye to a brute force attack. Details of this improvement are summarised below.

<table>
<thead>
<tr>
<th>Affected FishEye Versions</th>
<th>Fix Availability</th>
<th>More Details</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>All versions up to and including 2.2.1</td>
<td>2.2.3 only</td>
<td>FishEye allows an unlimited number of login attempts. This makes FishEye vulnerable to a brute force attack.</td>
<td>Moderate</td>
</tr>
</tbody>
</table>

**Risk Mitigation**
We recommend that you upgrade your FishEye installation to fix these vulnerabilities. Please see the 'fix' section below.

You can also prevent brute force attacks by following our guidelines on using Fail2Ban to limit login attempts.

**Fix**
This issue has been fixed in FishEye 2.2.3 (see the changelog). Later versions will include protection from this vulnerability. You can download FishEye 2.2.3 from the download centre.

**Changed Behaviour in FishEye**
In order to fix these issues, we have changed FishEye's behaviour as follows:

- After three consecutive failed login attempts, FishEye will display a CAPTCHA form asking the user to enter a given word when attempting to log in again. This will prevent brute force attacks via the login screen. The number of failed attempts needed to trigger the CAPTCHA testing is configurable. For more information, see the documentation for Brute Force Login Protection.

**Download Patches for Earlier FishEye / Crucible Versions**

These patch releases contain security fixes, which apply to the shared FishEye architecture that is the basis of both FishEye and Crucible.

These patches fix the Admin Escalation vulnerability only. Please note that these patches are for specific older point versions of FishEye (2.1.4, 2.0.6 or 1.6.6). If you are running an earlier version than these, you will need to upgrade to a version specifically addressed by one of these patches. To update a more recent version of the product (2.1.5 through 2.2.1), please upgrade to FishEye 2.2.3 or later. Atlassian strongly recommends that you upgrade to FishEye 2.2.3 or later.

MD5 checksums are provided to allow verification of the downloaded files.

**Patch for FishEye / Crucible 2.1.4**
### Crucible Security Advisory 2010-06-16

The 2.3.3 release of Crucible contains some security related fixes, which are part of the shared FishEye architecture. The following information for FishEye applies equally to Crucible.

The Crucible Download Centre has the updates for Crucible.

#### In this advisory:

- Remote Code Exploit Vulnerability
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- Download Patches for Earlier FishEye / Crucible Versions
  - Patch for FishEye / Crucible 2.3.2
  - Patch for FishEye / Crucible 2.2.3

### Remote Code Exploit Vulnerability

#### Severity

Atlassian rates this vulnerability as **critical**, according to the scale published in [Severity Levels for Security Issues](#). The scale allows us to rank a vulnerability as critical, high, moderate or low.

#### Risk Assessment

We have identified and fixed a remote code exploit vulnerability which affects FishEye and Crucible instances.

#### Vulnerability

This vulnerability allows a motivated attacker to call remote code on the host server.

All versions of FishEye/Crucible up to version 2.3.2 are affected by this vulnerability.

<table>
<thead>
<tr>
<th>Affected FishEye Versions</th>
<th>Fix Availability</th>
<th>More Details</th>
<th>Severity</th>
</tr>
</thead>
</table>

---

**To acquire all of the fixes on this page,** upgrade to **FishEye 2.2.3**, which you can download from the [download centre](#).
This vulnerability has been discovered in XWork by OpenSymphony, a command pattern framework which is used by FishEye and Crucible.

**About the XWork Framework:**

- See the [OpenSymphony XWork page](#) for more information about XWork.

**Risk Mitigation**

We strongly recommend either upgrading or patching your FishEye/Crucible installation to fix this vulnerability. Please see the 'Fix' section below.

**Fix**

These issues have been fixed in FishEye 2.3.3 (see the changelog), which you can download from the download centre.

It has also been fixed in Crucible 2.3.3 (see the changelog), which you can download from the download centre.

Later versions will include protection from this vulnerability.

This fix is also provided as a patch for FishEye/Crucible 2.3.2 and 2.2.3, which you can download from links on this page. Customers on earlier point versions of FishEye/Crucible will have to upgrade to version 2.3.2 or 2.2.3 before applying the patch. Atlassian recommends you upgrade to FishEye/Crucible 2.3.3.

**Download Patches for Earlier FishEye / Crucible Versions**

These patch releases contain security fixes, which apply to the shared FishEye architecture that is the basis of both FishEye and Crucible.

Please note that these patches are for specific point versions of FishEye (2.3.2 and 2.2.3). If you are running an earlier version than these, you will need to upgrade to a version specifically addressed by one of these patches. Atlassian strongly recommends that you upgrade to FishEye 2.3.3 / Crucible 2.3.3 or later.

MD5 checksums are provided to allow verification of the downloaded files.

**Patch for FishEye / Crucible 2.3.2**

<table>
<thead>
<tr>
<th>File</th>
<th>FishEye / Crucible Version</th>
<th>Release Date</th>
<th>MD5 Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>fisheye-crucible-2.3.2-patch1.zip</td>
<td>2.3.2</td>
<td>16th June, 2010</td>
<td>6fe98db821a6d26126907688af2cc84</td>
</tr>
</tbody>
</table>

**Patch for FishEye / Crucible 2.2.3**

<table>
<thead>
<tr>
<th>File</th>
<th>FishEye / Crucible Version</th>
<th>Release Date</th>
<th>MD5 Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>fisheye-crucible-2.2.3-patch1.zip</td>
<td>2.2.3</td>
<td>16th June, 2010</td>
<td>6fe98db821a6d26126907688af2cc84</td>
</tr>
</tbody>
</table>

Our thanks to [Meder Kydyraliev](#) of the Google Security Team who discovered this vulnerability. Atlassian fully supports the reporting of vulnerabilities and appreciates it when people work with Atlassian to identify and solve the problem.

**FishEye and Crucible Security Advisory 2011-01-12**

This advisory announces a number of security vulnerabilities that we have found and fixed in recent versions of FishEye/Crucible. You need to upgrade your existing FishEye/Crucible installations to fix these vulnerabilities. Enterprise Hosted customers should request an upgrade by filing a ticket at [http://support.atlassian.com](http://support.atlassian.com). JIRA Studio is not vulnerable to any of the issues described in this advisory.

Atlassian is committed to improving product security. The vulnerabilities listed in this advisory have been discovered by Atlassian, unless noted otherwise. The reporter may also have requested that we do not credit them.

If you have questions or concerns regarding this advisory, please raise a support request at [http://support.atlassian.com](http://support.atlassian.com).
In this advisory:

- XSS Vulnerabilities
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- Administration password logged in debug log
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- Review comment search returns comments that a user has no permission to view
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix
- Anonymous global access exposes entire user list
  - Severity
  - Risk Assessment
  - Vulnerability
  - Risk Mitigation
  - Fix

**XSS Vulnerabilities**

**Severity**

Atlassian rates the severity level of these vulnerabilities as **high**, according to the scale published in *Severity Levels for Security Issues*. The scale allows us to rank the severity as critical, high, moderate or low.

**Risk Assessment**

We have identified and fixed a number of cross-site scripting (XSS) vulnerabilities which may affect FishEye/Crucible instances, including publicly available instances (that is, internet-facing servers). XSS vulnerabilities potentially allow an attacker to embed their own JavaScript into a FishEye/Crucible page. You can read more about XSS attacks at [cgisecurity.com](http://cgisecurity.com), The Web Application Security Consortium and other places on the web.

**Vulnerability**

The table below describes the FishEye/Crucible versions and the specific functionality affected by the XSS vulnerabilities.

<table>
<thead>
<tr>
<th>FishEye/Crucible Feature</th>
<th>Affected FishEye/Crucible Versions</th>
<th>Issue Tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crucible review – linked JIRA issue</td>
<td>Crucible 2.0.1 – 2.3.7</td>
<td>CRUC-5307</td>
</tr>
<tr>
<td>Crucible email reviews</td>
<td>Crucible 2.2.0 – 2.4.0</td>
<td>CRUC-5308</td>
</tr>
<tr>
<td>Crucible review reload</td>
<td>Crucible 2.2.0 – 2.4.2</td>
<td>CRUC-5309</td>
</tr>
<tr>
<td>Crucible edit review details screen</td>
<td>Crucible 2.2.0 – 2.3.7</td>
<td>CRUC-5345</td>
</tr>
<tr>
<td>FishEye repository configuration</td>
<td>FishEye 2.4.0</td>
<td>CRUC-5310</td>
</tr>
<tr>
<td>FishEye charts</td>
<td>FishEye 2.2.0 - 2.4.0</td>
<td>CRUC-5311</td>
</tr>
<tr>
<td>FishEye/Crucible code macro</td>
<td>FishEye/Crucible 2.2.0 – 2.4.0</td>
<td>CRUC-5312</td>
</tr>
<tr>
<td>FishEye/Crucible changeset page heading</td>
<td>FishEye/Crucible 2.3.2 – 2.4.0</td>
<td>CRUC-5313</td>
</tr>
</tbody>
</table>

**Risk Mitigation**

We recommend that you upgrade your FishEye/Crucible installation to fix these vulnerabilities.

Alternatively, if you are not in a position to upgrade immediately and you judge it necessary, you can disable public signup to your instance until you have applied the upgrade. For even tighter control, you could restrict access to trusted groups.

**Fix**
FishEye/Crucible 2.4.4 fixes all of these issues. View the issues linked above for information on earlier fix versions for each issue. For a full description of this release, see the FishEye 2.4 changelog and Crucible 2.4 Changelog. You can download the latest version of FishEye/Crucible from the download centre (FishEye download centre, Crucible download centre).

There are no patches available to fix these vulnerabilities. You must upgrade your FishEye/Crucible installation.

**Administration password logged in debug log**

**Severity**

Atlassian rates the severity level of these vulnerabilities as **high**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank the severity as critical, high, moderate or low.

**Risk Assessment**

We have identified and fixed a vulnerability in the FishEye/Crucible logging which may affect FishEye/Crucible instances, including publicly available instances (that is, internet-facing servers). This vulnerability allows administrator passwords to be logged in clear text when debug logging is enabled.

**Vulnerability**

This vulnerability affects FishEye and Crucible 2.2.0 to 2.4.0.

**Risk Mitigation**

We recommend that you upgrade your FishEye/Crucible installation to fix these vulnerabilities.

Alternatively, if you are not in a position to upgrade immediately and you judge it necessary, you can disable access logging. See Enabling Access Logging in FishEye and Enabling Access Logging in Crucible. You can also apply file restrictions to your log files. Note, this issue only occurs when DEBUG logging is turned on (off by default) when an administrator logs in.

**Fix**

FishEye/Crucible 2.4.2 and later fix this issue. For a full description of this release, see the FishEye 2.4 changelog and Crucible 2.4 Changelog. You can download the latest version of FishEye/Crucible from the download centre (FishEye download centre, Crucible download centre).

There are no patches available to fix this vulnerability. You must upgrade your FishEye/Crucible installation.

**Review comment search returns comments that a user has no permission to view**

**Severity**

Atlassian rates the severity level of these vulnerabilities as **medium**, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank the severity as critical, high, moderate or low.

**Risk Assessment**

We have identified and fixed a vulnerability in the Crucible review comment search which may affect Crucible instances, including publicly available instances (that is, internet-facing servers). This vulnerability allows review comments to be displayed for projects that are not publicly viewable.

**Vulnerability**

This vulnerability affects Crucible 2.2.0 to 2.4.3.

**Risk Mitigation**

We recommend that you upgrade your FishEye/Crucible installation to fix these vulnerabilities.

**Fix**

FishEye/Crucible 2.2.5, 2.3.8 and 2.4.4 fix this issue. For a full description of this release, see the FishEye 2.4 changelog and Crucible 2.4 Changelog. You can download these versions of FishEye/Crucible via the download centre (FishEye download centre, Crucible download centre).

There are no patches available to fix this vulnerability. You must upgrade your FishEye/Crucible installation.

**Anonymous global access exposes entire user list**
Severity

Atlassian rates the severity level of these vulnerabilities as medium, according to the scale published in Severity Levels for Security Issues. The scale allows us to rank the severity as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a vulnerability in the FishEye/Crucible anonymous global access which may affect FishEye/Crucible instances, including publicly available instances (that is, internet-facing servers). This vulnerability exposes the user list (usernames and emails) of a FishEye/Crucible instance for access when anonymous global access is enabled.

Vulnerability

This vulnerability affects FishEye and Crucible 2.2.0 to 2.4.3.

Risk Mitigation

We recommend that you upgrade your FishEye/Crucible installation to fix these vulnerabilities.

Alternatively, if you are not in a position to upgrade immediately and you judge it necessary, you can disable global anonymous access. See Configuring Anonymous Access.

Fix

FishEye/Crucible 2.2.6, 2.3.8 and 2.4.4 fix this issue. For a full description of this release, see the FishEye 2.4 changelog and Crucible 2.4 Changelog. You can download these versions of FishEye/Crucible via the download centre (FishEye download centre, Crucible download centre).

There are no patches available to fix this vulnerability. You must upgrade your FishEye/Crucible installation.

Crucible Upgrade Guide

- Upgrading to a New Version of Crucible
- Upgrading from FishEye to Crucible

Upgrading to a New Version of Crucible

This page describes the recommended method of upgrading to a new version of Crucible.

Read about how your Crucible installation works with FishEye.

On this page:

- Before you Start
- Upgrade Procedure
  - Method 1 - Using a FISHEYE_INST Directory
  - Method 2 - Without a FISHEYE_INST Directory
  - Method 3 - Without a FISHEYE_INST Directory, but would like to set one up
- Checking for Known Issues and Troubleshooting the Crucible Upgrade

Before you Start

- Back up your entire Crucible instance (see Backing Up and Restoring Crucible Data), i.e.
  - If you are backing up your Crucible instance via the Admin interface, tick all of the ‘Include’ checkboxes (e.g. plugins, templates, uploads, SQL database, etc).
  - If you are backing up your Crucible instance using the command-line interface, do not use any exclusion options.
- Read the Release Notes and Changelog and version-specific Upgrade Guides for the version you are upgrading to, as well as any versions you are skipping.
- Check the Supported Platforms to ensure that your system meets the requirements for the new version.
- Download the Crucible zip file.

Upgrade Procedure

Method 1 - Using a FISHEYE_INST Directory

If you have Crucible configured to use a FISHEYE_INST directory, then simply:

1. Shutdown your existing FishEye server.
2. Make a backup of your FISHEYE_INST directory
Before you Start

We strongly recommend you make a backup of your data before following the steps below. Refer to the documentation on making a backup.

Upgrade Procedure

3. Extract the new Crucible version to a directory.
4. Leave your FISHEYE_INST environment variable set to its existing location.
5. Start Crucible from the new installation.

Read more about the FISHEYE_INST environment variable.

Method 2 - Without a FISHEYE_INST Directory

If you are using the FISHEYE_HOME directory, you will need to copy some files from your old Crucible installation to your new one.

1. Extract the new Crucible archive into a directory such as /NEW_FISHEYE/.
2. Delete the /NEW_FISHEYE/var directory.
3. Shut down the old Crucible instance if it is running.
4. Copy /OLD_FISHEYE/config.xml to /NEW_FISHEYE/.
5. Copy (or move) the /OLD_FISHEYE/var directory to /NEW_FISHEYE/var.
6. Copy your database drivers to your new lib directory under NEW_FISHEYE/lib.
7. Follow any version-specific instructions found in the Release Notes.

Method 3 - Without a FISHEYE_INST Directory, but would like to set one up

1. Shut down the old FishEye instance if it is running.
2. Set up the FISHEYE_INST environment variable, then create the FISHEYE_INST directory on your filesystem.
3. Copy the /OLD_FISHEYE/config.xml to /FISHEYE_INST.
4. Copy the /OLD_FISHEYE/var directory to /FISHEYE_INST.
5. Download Crucible.
6. Extract the new Crucible archive into a directory such as /NEW_FISHEYE/.
7. Start Crucible from the new installation by running NEW_FISHEYE/bin/run.sh. (Use run.bat on Windows).
8. Follow the initial configuration steps outlined below.
9. If your configuration is not automatically picked up and you cannot see your existing repositories, check your Administration > Sys-Info page, where you will see information about FISHEYE_HOME and FISHEYE_INST. Check your FISHEYE_INST is pointing to the right directory.

Checking for Known Issues and Troubleshooting the Crucible Upgrade

If something is not working correctly after you have completed the steps above to upgrade your Crucible installation, please check for known Crucible issues and try troubleshooting your upgrade as described below:

- Check for known issues. Sometimes we find out about a problem with the latest version of Crucible after we have released the software. In such cases we publish information about the known issues in the Crucible Knowledge Base. Please check the Crucible 2.4 Known Issues in the Crucible Knowledge Base and follow the instructions to apply any necessary patches if necessary.

- Did you encounter a problem during the Crucible upgrade? Please refer to the guide to troubleshooting upgrades in the Crucible Knowledge Base.

- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Upgrading from FishEye to Crucible

If you have been using FishEye and now want to move to Crucible, you can do this without losing your FishEye repositories.

Read about how your Crucible installation works with FishEye.

On this page:

- Before you Start
- Upgrade Procedure
  - Method 1 - Without a FISHEYE_INST Directory (default)
  - Method 2 - Using a FISHEYE_INST Directory
  - Method 3 - Without a FISHEYE_INST Directory, but intending to set one up
- Initial Crucible Configuration
• Follow **Method 1** if you have a default configuration and are not using a `FISHEYE_INST` directory (that is, your FishEye binaries and data are all stored under the same location, in the default `FISHEYE_HOME` directory).

• Follow **Method 2** below if you have FishEye configured to use a `FISHEYE_INST` directory (that is, your FishEye binaries are stored in the `FISHEYE_HOME` directory, separate from your FishEye data in the `FISHEYE_INST` directory).

• Follow **Method 3** if you are not using a `FISHEYE_INST` directory but would now like to start using one.

Read more about the `FISHEYE_INST` environment variable.

**Method 1 - Without a `FISHEYE_INST` Directory (default)**

1. Download Crucible.
2. Extract the new Crucible archive into a directory such as `/NEW_FISHEYE/`.
3. Delete the `/NEW_FISHEYE/var` directory.
4. Shut down the old FishEye instance if it is running.
5. Copy `/OLD_FISHEYE/config.xml` to `/NEW_FISHEYE/`.
6. Copy the `/OLD_FISHEYE/var` directory to `/NEW_FISHEYE/var`.
7. If you have a Cenqua-issued FishEye license, copy `/OLD_FISHEYE/fisheye.license` to `/NEW_FISHEYE/`. (Atlassian-issued licenses are included within `config.xml`.)
9. Start Crucible from the new installation by running `NEW_FISHEYE/bin/run.sh`. (Use `run.bat` on Windows.)
10. Follow the initial configuration steps outlined below.

**Method 2 - Using a `FISHEYE_INST` Directory**

1. Shutdown your existing fisheye server.
2. Make a backup of your `FISHEYE_INST` directory.
3. Download Crucible and unzip the archive into a folder. This document assumes you have extracted your Crucible zip file into a directory called `/NEW_FISHEYE/`.
4. Leave your `FISHEYE_INST` environment variable set to its existing location.
5. Start Crucible from the new installation by running `NEW_FISHEYE/bin/run.sh`. (Use `run.bat` on Windows.)
6. Follow the initial configuration steps outlined below.

**Method 3 - Without a `FISHEYE_INST` Directory, but intending to set one up**

1. Shut down the old FishEye instance if it is running.
2. Set up the `FISHEYE_INST` environment variable, then create the `FISHEYE_INST` directory on your filesystem.
3. Copy the `/OLD_FISHEYE/config.xml` to `/FISHEYE_INST`.
4. Copy the `/OLD_FISHEYE/var` directory to `/FISHEYE_INST`.
5. Download Crucible.
6. Extract the new Crucible archive into a directory such as `/NEW_FISHEYE/`.
7. Start Crucible from the new installation by running `NEW_FISHEYE/bin/run.sh`. (Use `run.bat` on Windows.)
8. Follow the initial configuration steps outlined below.
9. If your configuration is not automatically picked up and you cannot see your existing repositories, check your Administration > Sys-Info page, where you will see information about `FISHEYE_HOME` and `FISHEYE_INST`. Check your `FISHEYE_INST` is pointing to the right directory.

**Initial Crucible Configuration**

1. You can access FishEye immediately by going to `http://HOSTNAME:8060/` in a browser.
2. The first time you run FishEye, enter your Crucible license key. To do this, update your Crucible license by opening 'Administration', then 'Sys-Info/Support'. On this screen, you can enter your Crucible license key. You can view your license key here. The Crucible functionality will be instantly unlocked.
3. If you do not already have user accounts configured, you will need to do this via the Administration screens or by configuring Crucible/FishEye to use external authentication. To add users:
   - Open the FishEye Administration screens at `http://HOSTNAME:8060/admin/`.
   - Click 'Users/Security' under 'Global Settings' in the 'Admin Menu'.
   - Read more details about the different ways of creating users.
4. Crucible can email each review participant on a range of changes. Each user can then set up their own preferences. This is described in the User Profile guide. First, you must set up the SMTP Server.

**Supported Platforms**

This page shows the supported platforms for Crucible 2.5.x and its minor releases.

Key: ![Supported] = Supported; ![Not Supported] = Not Supported
### Java Version

| JRE / JDK (1) | ✔ 1.5 or later |

### Operating Systems

| Microsoft Windows (2) | ✔ |
| Linux (2) | ✔ |
| Apple Mac OS X (2) | ✔ |

### Databases

| MySQL           | ✔ MySQL Enterprise Server 5.0.21 or later ✔ MySQL Community Server 5.0.21 or later |
| PostgreSQL      | ✔ 8.1 or later |
| Oracle          | ✔ 11g |
| HSQLDB (3)      | ✔ (bundled; default) |

### Web Browsers

| Microsoft Internet Explorer | ✔ 7.0 or later, IE6 is not supported |
| Mozilla Firefox            | ✔ 3 or later |
| Safari                     | ✔ 4 or later |

### Version Control Systems

| Subversion | ✔ Server 1.1 or later, via an SCM plugin or with FishEye. The client uses SVNkit or native JavaHL. |
| CVS (and CVSNT) | ✔ All versions (requires FishEye) |
| Perforce     | ✔ Client version 2007.3 or later, via an SCM plugin or with FishEye. |
| Git          | ✔ 1.6 or later (requires FishEye) |
| IBM ClearCase | ✔ 2003.06.10 or later (requires FishEye) |

---

Crucible also supports the use of Confluence (v2.9.1 through v2.10.4, via an SCM plugin) or the server file system (via an SCM plugin) as a repository.

### Supported Platform Notes

1. Crucible requires Java Runtime (JDK or JRE) version 1.5 or later (Solaris requires 1.5.0_15 as a minimum). Pre-release/Early access versions of the Java Runtime are not supported.

You can download a Java Runtime for Windows/Linux/Solaris. On Mac OS X, the JDK is bundled with the operating system.
Once you have installed the JDK, you need to set the `JAVA_HOME` environment variable.

We strongly recommend the use of a 32-bit JDK/JRE rather than a 64-bit JDK/JRE. 64-bit JDK/JREs will consume the available RAM more rapidly, and this may result in poor performance.

2. Crucible is a pure Java application and should run on any platform provided the requirements for the JRE or JDK are satisfied.

3. The Crucible built-in database, running HSQLDB is somewhat susceptible to data loss during system crashes. We recommend that you do not use HSQLDB for production systems. External databases are generally more resistant to data loss during a system crash and are more suited for production use.

At this time, Crucible supports the following external databases:

- MySQL Enterprise Server 5.0.21 onwards and MySQL Community Server 5.0.21 onwards (see the Crucible Database documentation).
- PostgreSQL 8.x onwards (see the Crucible Database documentation).
- Oracle 11g onwards (see the FishEye Database documentation).

### Deployment Notes for Source Code Repositories

Crucible also supports the use of Confluence (v2.9.1 through v2.10.4, via an SCM plugin) or the server file system (via an SCM plugin) as a repository.

Crucible can also store uploaded files in its own database, removing the need for any kind of repository. Subversion, CVS, Perforce, Git and IBM ClearCase are supported when Crucible is used with FishEye. See the FishEye Supported Platforms.

### Font size tips

(Especially for Linux users.) For best results you may want to tweak your default monospace font and font-size. The default browser font is usually Courier New which can be hard to read in some browsers. We recommend choosing the same font you use in your IDE and selecting a font size approximately 2 points larger than your variable width font. Firefox 3, Internet Explorer 7 and Safari all have excellent font rendering. It is worth taking some time to tweak your fonts for the best experience.

### End of Support Announcements for Crucible

This page contains announcements of the end of support for various platforms and browsers when used with Crucible. This is summarised in the table below. Please see the sections following for the full announcements.

#### End of Support Matrix for Crucible

<table>
<thead>
<tr>
<th>Platform</th>
<th>Crucible End of Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Explorer 7 web browser</td>
<td>Crucible 2.6 (announcement)</td>
</tr>
<tr>
<td>Java Platform 5 (JDK/JRE 1.5)</td>
<td>Crucible 2.6 (announcement)</td>
</tr>
</tbody>
</table>

The table above summarises information regarding the end of support announcements for upcoming Crucible releases. If a platform (version) has already reached its end of support date, it is not listed in the 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 the complexity of our applications grows, the cost of supporting multiple platforms increases exponentially. Each new feature has to be tested on several combinations of application servers, databases, web browsers, etc, with setup and ongoing maintenance of automated tests. Moving forward, we want to reduce the time spent there to increase Crucible development speed significantly.

On this page (most recent announcements first):

- Deprecated Web Browsers for Crucible (21 March 2011)
- Deprecated Java Platforms for Crucible (21 March 2011)

### Deprecated Web Browsers for Crucible (21 March 2011)

This section announces the end of Atlassian support for certain web browsers for Crucible.
We will stop supporting older versions of web browsers as follows:

- From Crucible 2.6, due in May 2011, support for Internet Explorer 7 will end.

The details are below. Please refer to the Supported Platforms for more details regarding platform support for Crucible. 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 7</td>
<td>When Crucible 2.6 releases (target May 2011)</td>
</tr>
</tbody>
</table>

**Internet Explorer 7 Notes:**

- Crucible 2.5 is the last version to officially support Internet Explorer 7.
- Crucible 2.6 is currently targeted to release in May 2011 and will not be tested with Internet Explorer 7. After the Crucible 2.6 release, Atlassian will not provide fixes in older versions of Crucible for bugs affecting Internet Explorer 7.

---

**Deprecated Java Platforms for Crucible (21 March 2011)**

This section announces the end of Atlassian support for certain Java Platforms for Crucible.

We will stop supporting the following Java Platforms:

- From Crucible 2.6, due in May 2011, 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 assist 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 Crucible. 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 Crucible 2.6 releases (target May 2011)</td>
</tr>
</tbody>
</table>

**Java Platform 5 End of Support Notes:**

- Crucible 2.5 is the last version to officially support Java Platform 5 (JDK/JRE 1.5).
- Crucible 2.6 is currently targeted to release in May 2011 and will not be tested with Java Platform 5 (JDK/JRE 1.5). After the Crucible 2.6 release, Atlassian will not provide fixes in older versions of Crucible for bugs affecting Java Platform 5 (JDK/JRE 1.5).

---

**Copy of End of Support Announcements for Crucible**

This page is a draft in progress and visible to atlassian-staff only. Includes ClearCase EoL.

*Do not use this page directly. Copy the content to End of Support Announcements for Crucible.*

This page contains announcements of the end of support for various platforms and browsers when used with Crucible. This is summarised in the table below. Please see the sections following for the full announcements.

**End of Support Matrix for Crucible**

<table>
<thead>
<tr>
<th>Platform</th>
<th>Crucible End of Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Explorer 7 web browser</td>
<td>Crucible 2.6 (announcement)</td>
</tr>
<tr>
<td>Java Platform 5 (JDK/JRE 1.5)</td>
<td>Crucible 2.6 (announcement)</td>
</tr>
<tr>
<td>IBM ClearCase (all versions)</td>
<td>4 April 2012 (announcement)</td>
</tr>
</tbody>
</table>

The table above summarises information regarding the end of support announcements for upcoming Crucible releases. If a platform (version) has already reached its end of support date, it is not listed in the 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 the complexity of our applications grows, the cost of supporting multiple platforms increases exponentially. Each new feature has to be tested on several combinations of application servers, databases, web browsers, etc, with setup and ongoing maintenance of automated tests. Moving forward, we want to reduce the time spent there to increase Crucible development speed significantly.

On this page (most recent announcements first):

- **Deprecated Web Browsers for Crucible (21 March 2011)**
- **Deprecated Java Platforms for Crucible (21 March 2011)**
- **Deprecated SCM Repository Support for Crucible (4 April 2011)**

**Deprecated Web Browsers for Crucible (21 March 2011)**

This section announces the end of Atlassian support for certain web browsers for Crucible.

We will stop supporting older versions of web browsers as follows:

- From Crucible 2.6, due in May 2011, support for Internet Explorer 7 will end.

The details are below. Please refer to the Supported Platforms for more details regarding platform support for Crucible. 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 7</td>
<td>When Crucible 2.6 releases (target May 2011)</td>
</tr>
</tbody>
</table>

**Internet Explorer 7 Notes:**

- Crucible 2.5 is the last version to officially support Internet Explorer 7.
- Crucible 2.6 is currently targeted to release in May 2011 and will not be tested with Internet Explorer 7. After the Crucible 2.6 release, Atlassian will not provide fixes in older versions of Crucible for bugs affecting Internet Explorer 7.

**Deprecated Java Platforms for Crucible (21 March 2011)**

This section announces the end of Atlassian support for certain Java Platforms for Crucible.

We will stop supporting the following Java Platforms:

- From Crucible 2.6, due in May 2011, 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 assist 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 Crucible. 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 Crucible 2.6 releases, due May 2011</td>
</tr>
</tbody>
</table>

**Java Platform 5 End of Support Notes:**

- Crucible 2.5 is the last version to officially support Java Platform 5 (JDK/JRE 1.5).
- Crucible 2.6 is currently targeted to release in May 2011 and will not be tested with Java Platform 5 (JDK/JRE 1.5). After the Crucible 2.6 release, Atlassian will not provide fixes in older versions of Crucible for bugs affecting Java Platform 5 (JDK/JRE 1.5).

**Deprecated SCM Repository Support for Crucible (4 April 2011)**
This section announces the end of Atlassian support for certain SCM repositories for Crucible. End of support means that Atlassian will remove all functionality related to certain SCM repositories past the specified date. Releases before that date will contain the functionality that supports the SCM, however, Atlassian will fix only critical bugs that affect functionality for that SCM, and will not add any new features for that SCM.

Please refer to the Supported Platforms for more details regarding platform support for Crucible. If you have questions or concerns regarding these announcements, please email eol-announcement at atlassian dot com.

<table>
<thead>
<tr>
<th>SCM Repository</th>
<th>Support End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM ClearCase (all versions)</td>
<td>4 April 2012</td>
</tr>
</tbody>
</table>

**IBM ClearCase End of Support Notes:**

- Atlassian intends to end of life IBM ClearCase functionality on 4 April 2012. The release of Crucible after 4 April 2012 will not contain any IBM ClearCase functionality.
- As mentioned above, the releases of Crucible before 4 April 2012 will contain support for IBM ClearCase. However, we will only be fixing critical bugs related to IBM ClearCase and will not be adding any features.

---

**Crucible FAQ**

Answers to frequently asked questions about configuring and using Crucible.

- Troubleshooting
  - Crucible freezes unexpectedly
  - JIRA Integration Issues
  - Problems with very long comments and MySQL migration
- Increasing the session timeout
- General FAQs
  - Can Crucible be run as a Windows Service?
  - Can I deploy Crucible or FishEye as a WAR?
  - Does Crucible support SSL (HTTPS)?
  - How to Automate Daily Crucible Backups
- Licensing FAQ
  - What happens if I decide to stop using FishEye with Crucible?
  - Do I need a FishEye licence to run Crucible?
  - Advantages of Native Repository Access over lightSCM plugins
- Support Policies
  - 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
- Fix 'Out of Memory' errors by increasing available memory

Most setup issues are likely to be related to the FishEye component of Crucible. Refer to the FishEye documentation:

- FishEye documentation
- FishEye FAQs

- Top Evaluator Questions
  - Can Crucible add support for new repositories?
  - Can I purchase Crucible on it's own?
  - Can I trial Crucible without FishEye?
  - How can I do reviews from the file system?
  - How does Crucible help enforce compliance and auditability?
  - How do I convince my team of the benefits of code review?
  - How do I do pre-commit reviews?
  - How do I raise defects in JIRA?
  - How do I review patch diffs?
  - What user permissions and review security is available?

Do you still have a question, or need help with Crucible? Please create a support request.
Troubleshooting

Crucible Troubleshooting

- Crucible freezes unexpectedly
- JIRA Integration Issues
- Problems with very long comments and MySQL migration — h3. (color:#000000)Affects Version(color)

**Crucible Troubleshooting**

The most common cause of FishEye/Crucible issues is an incorrect symbolic setup (trunk/branch/tag) for Subversion repositories. If you are using Subversion and your initial index is taking forever, double-check that your symbolic setup matches your repository.

FishEye runs with the default Java heap of 64 megabytes. This is sometimes problematic for FishEye, especially for Subversion repositories during the initial scan. You can give FishEye's JVM more memory by setting the FISHEYE_OPTS environment variable.

Starting Crucible with the command line options --debug --debug-perf will print a lot of information to Crucible's logs. This can give you an insight into what is happening and possibly where you are stuck. Attach these logs along with your config.xml to an Atlassian support ticket, to speed up your support request.

**Crucible freezes unexpectedly**

**Issue Symptoms**

If your Crucible 2.0 or 2.0.1 instance freezes unexpectedly, this could be caused by a known issue with Crucible and MySQL database technology.

This issue manifests itself in some Crucible pages returning a server timeout error. To identify the issue, check the Crucible error log. For this issue, the following output will appear in the error log:

```
2009-07-15 15:34:45,555 ERROR [btpool0-519] fisheye.app HibernateUtil-commitTransaction - Commit fail
  msg-0:Could not execute JDBC batch update
  msg-1:Lock wait timeout exceeded; try restarting transaction
  ...
  ...
  Caused by: java.sql.BatchUpdateException: Lock wait timeout exceeded; try restarting transaction
  at com.mchange.v2.c3p0.impl.NewProxyPreparedStatement.executeBatch(NewProxyPreparedStatement.java:1723)
  at org.hibernate.jdbc.BatchingBatcher.doExecuteBatch(BatchingBatcher.java:48)
  at org.hibernate.jdbc.AbstractBatcher.executeBatch(AbstractBatcher.java:246)
  ...
```

The Crucible error log can be found under FISHEYE_INST/var/log/fisheye-error.log.YYYY-MM-DD.

See the JIRA issue for more information.

**Workaround**

Until the issue is solved, the suggested course of action is to restart your Crucible instance. This will return Crucible to normal operation.

The Crucible development team is actively working on a solution and this be part of an upcoming point release of Crucible.

**Requesting Support**

If you require assistance in resolving the problem, please raise a support request under the Crucible project.

**JIRA Integration Issues**

Users are mapped to their own accounts when using Trusted Applications.
If you (or the general account used for JIRA access, if not using Trusted Applications) do not have the permissions to carry out the JIRA actions linked from Crucible, an error will occur. Depending on the error returned from JIRA, Crucible may not display the error correctly or display it at all, simply reporting that "An error has occurred". To investigate what the error was, you can access the Crucible debug log, named fisheye-debug.log.YYYY-MM-DD under the dist.inst/var/log folder of your Crucible installation. In the debug log, look for the date and time when your error took place. Here, you will be able to follow the links and see what error the JIRA instance was producing by clicking through to JIRA.

If you are using JIRA 4.0 you will not be able to create subtasks in versions of Crucible prior to 2.0.5. If you are affected by this bug, please upgrade to at least 2.0.6 (2.0.5 is affected by another bug CRUC-2471).

**Problems with very long comments and MySQL migration**

**Affects Version**

This issue was introduced in Crucible 2.0 and fixed in Crucible 2.1.

**Issue Symptoms**

There is a known issue with Crucible 2.0.x and very long comments when migrating your database to MySQL. In some circumstances, this might result in truncation of very long comments, causing data loss.

Depending on your MySQL configuration, you may see an error message like this while migrating to MySQL, causing the migration to fail:

```plaintext
2009-07-16 16:56:12,390 ERROR [ThreadPool1] fisheye.app
com.cenqua.crucible.actions.admin.database.DBEditHelper-doGet -
Database migration failed:
java.sql.BatchUpdateException: Data truncation: Data too long for column 'cru_message' at row 1
java.sql.BatchUpdateException: Data truncation: Data too long for column 'cru_message' at row 1
```

You may not see the message if you are running MySQL with default settings.

For more information, see the [JIRA issue](#).

**Workaround**

If your data contains very long comments or review descriptions (longer than 21,845 multibyte unicode characters), consider avoiding use of MySQL until you can upgrade the product. Alternatively, use PostgreSQL or the default (built-in) HSQLDB database.

This issue is now resolved. This issue was introduced in Crucible 2.0 and fixed in Crucible 2.1.

**Requesting Support**

If you require assistance in resolving the problem, please [raise a support request](#) under the Crucible project.

**Increasing the session timeout**

Crucible comes with remember me functionality, i.e. so long as user hasn’t logged out the computer will remember the user. So technically a user should not be logged out, unless the user has disabled the saving of cookies in their browser settings.

If the user has disabled cookies, and does not want to enable the saving of cookies, then, you can add:

```xml
<session-config>
  <session-timeout>120</session-timeout>
</session-config>
```

in your WEB-INF/web.xml file (see [Jetty Documentation](#)), which will increase the session timeout value to two hours.

**General FAQs**
Can Crucible be run as a Windows Service?

To run Crucible as a service you can either use SRVANY and INSTSRV to run java.exe or create a Java Service Wrapper. A mechanism to run Crucible as a service will be incorporated at a later stage. In the meantime, example wrapper files written by users can be found here.

To install on Windows:

1. Unzip the wrapper zip file into your FISHEYE_HOME directory (Note, the end structure should be FISHEYE_HOME/wrapper, FISHEYE_HOME/wrapper/bin, etc and NOT FISHEYE_HOME/wrapper/wrapper, FISHEYE_HOME/wrapper/wrapper/bin. The location of the wrapper directory is important).
2. Run Fisheye-Install-NTService.bat, found in FISHEYE_HOME/wrapper/bin.
3. Start the Fisheye service under the Windows Control Panel.
4. Set your FISHEYE_INST within your FISHEYE_HOME/wrapper/conf/wrapper.conf as per the instructions below:

Please note, that if you do run as a service, then any Environment Variables that you want to set, need to be set in your FISHEYE_HOME/wrapper/conf/wrapper.conf file.

If there are other java parameters you wish to add, then you will need to add them under the additional parameters, e.g.

<table>
<thead>
<tr>
<th>JDK 1.5 Additional Parameters for jmx</th>
</tr>
</thead>
<tbody>
<tr>
<td>wrapper.java.additional.4=-Dcom.sun.management.jmxremote</td>
</tr>
<tr>
<td>wrapper.java.additional.5=-Dcom.sun.management.jmxremote.port=4242</td>
</tr>
<tr>
<td>wrapper.java.additional.6=-Dcom.sun.management.jmxremote.authenticate=false</td>
</tr>
<tr>
<td>wrapper.java.additional.7=-Dcom.sun.management.jmxremote.ssl=false</td>
</tr>
<tr>
<td>wrapper.java.additional.8=-Dcom.sun.management.jmxremote.authenticate=false</td>
</tr>
<tr>
<td>wrapper.java.additional.9=-Dcom.sun.management.jmxremote.password.file=./wrapper/jmxremote.password</td>
</tr>
<tr>
<td>wrapper.java.additional.10=-Dwrapper.mbean.name=&quot;wrapper:type=Java Service Wrapper Control&quot;</td>
</tr>
</tbody>
</table>

For example if you wish to add a FISHEYE_INST environment variable or add the java parameter "MaxPermSize", or the -Xrs options (should be used if running FishEye as a service under Windows, to prevent the JVM closing when an interactive user logs out) then it would be something like:

```
wrapper.java.additional.11=-Dfisheye.inst="c:/path/to/FISHEYE_INST"
wrapper.java.additional.12=-XX:MaxPermSize=128m
wrapper.java.additional.13=-Xrs
```

Your memory settings can also be found in this file:

<table>
<thead>
<tr>
<th>Initial Java Heap Size (in MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>wrapper.java.initmemory=32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Maximum Java Heap Size (in MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>wrapper.java.maxmemory=256</td>
</tr>
</tbody>
</table>

Increase these values if you have a large repository or expect to use more memory (init of 256, and a max of 1024 would be reasonable).

Can I deploy Crucible or FishEye as a WAR?

Unfortunately FishEye and Crucible cannot be deployed as a WAR. FishEye has some special needs for performance reasons that are not easily supported on third-party containers. Whilst this is an often requested feature, there are no immediate plans to provide a WAR version of FishEye or FishEye+Crucible. However the upcoming separate edition of Crucible (i.e. without FishEye) may at some stage be available as a WAR.
Does Crucible support SSL (HTTPS)?

Crucible does not have any built-in support for running over SSL via the HTTPS protocol. However, it is possible to setup a proxy web server to forward requests to Crucible. Please see the page on [Integrating with Other Web Servers](#).

How to Automate Daily Crucible Backups

Configuring Crucible backups is easy. To set daily Crucible backups, open the administration page, click the 'Backup' link under 'System' on the left navigation bar, and simply follow the instructions set out on the [Backing Up and Restoring Crucible Data](#) page.

Licensing FAQ

<table>
<thead>
<tr>
<th>Crucible Licensing FAQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>• What happens if I decide to stop using FishEye with Crucible? — Crucible can be run as a standalone application without FishEye. However, if you decide to stop using FishEye with Crucible, you will lose certain functionality and will need to make configuration changes.</td>
</tr>
<tr>
<td>• Do I need a FishEye licence to run Crucible? — FishEye and Crucible are separate products. They can be run separately, and they can also be run together.</td>
</tr>
<tr>
<td>• Advantages of Native Repository Access over lightSCM plugins</td>
</tr>
</tbody>
</table>

What happens if I decide to stop using FishEye with Crucible?

Crucible can be run as a standalone application without FishEye. However, if you decide to stop using FishEye with Crucible, you will lose certain functionality and will need to make configuration changes.

On this page:

- How do I run Crucible without FishEye?
- How is Crucible without FishEye different from using Crucible with FishEye?
  - Conducting Reviews
  - Viewing Repositories/Files
  - Charts
- Can I still use lightSCM plugins with Crucible?

How do I run Crucible without FishEye?

- Have a valid Crucible license but not a FishEye license
  To run Crucible without FishEye, you need to have a valid Crucible license but not a FishEye license. Crucible will actually use a “light” mode of FishEye when you do not have valid FishEye license. Light FishEye is bundled with Crucible and does not need to be installed separately. For more information on Crucible with light FishEye, see [How is Crucible without FishEye different from using Crucible with FishEye?](#) below.

- Reconnect your repositories
  Any repositories that you have currently defined in FishEye will not be visible in Crucible after removing FishEye. You will need to reconnect these repositories, as described in the FishEye documentation. Note, all repositories supported in FishEye are supported in light FishEye.

  ⚠️ Legacy “lightSCM” plugins, like the Crucible Subversion SCM plugin, will still work. However, the functionality will be limited compared to using Crucible with light FishEye. See the [Can I still use lightSCM plugins with Crucible?](#) section below for more information.

How is Crucible without FishEye different from using Crucible with FishEye?

The following changes in functionality will occur if you use Crucible without FishEye (i.e. use Crucible with “light” FishEye).

**Conducting Reviews**

- When using Iterative Reviews in Crucible, you will not be prompted when a new version of a file is available.

**Viewing Repositories/Files**

- Files and changesets displayed in activity streams (e.g. the dashboard activity stream) will not render as links to the relevant files/changesets.

- You will not be able to see your content roots and repositories associated with projects.
You will no longer be able to see repository lists and browse repositories using the 'Source' tab.

Charts

- You will not be able to view charts or code metrics.

Can I still use lightSCM plugins with Crucible?

Legacy "lightSCM" plugins, like the Crucible Subversion SCM plugin, will still work with Crucible. However, we recommend that you use the "light" FishEye implementation that is bundled with Crucible, as it supersedes the lightSCM plugins.

For more information, please read this FAQ: Advantages of Native Repository Access over lightSCM plugins.

Do I need a FishEye licence to run Crucible?

FishEye and Crucible are separate products. They can be run separately, and they can also be run together.

We recommend that you run Crucible together with FishEye. If you choose to run Crucible standalone without FishEye, you will have access to your repositories via the "light" FishEye implementation bundled with Crucible. However, a number of FishEye's advanced features will not be available to you, including pre-caching repository content (for improved performance), the ability to search and browse through repositories and FishEye's activity graphs.

For more information, please read the following FAQ: What happens if I decide to stop using FishEye with Crucible?

Advantages of Native Repository Access over lightSCM plugins

Prior to Crucible 2.4, running Crucible without FishEye required the use of "lightSCM" plugins (like the Crucible Subversion SCM plugin). From version 2.4, Crucible provides native repository access which supersedes Crucible's bundled lightSCM plugins. Third-party lightSCM plugins are unaffected and will continue to work with Crucible. The bundled plugins will still be available, so your existing configurations will also continue to work unchanged.

If you are currently using any of the bundled lightSCM plugins, we recommend that you migrate to using native repository access for the following reasons:

- Atlassian's lightSCM plugins (not lightSCM itself) are being deprecated, i.e. we will not update any of the bundled lightSCM plugins after the 2.4 release.
- It is easier for us to support and maintain a single implementation of our SCM interfaces, rather than support the standard FishEye access and the lightSCM implementations.
- Native repository access provides full support for SCMs for which there are no current lightSCM plugin implementations, including CVS and Mercurial.
- Native repository access provides additional functionality that is not available in the lightSCM plugins including:
  - Viewable commits in the activity streams (e.g. the dashboard activity stream).
  - Repositories administration via the administration console.
  - Easier review creation due to the ability to search and browse the repository using the full power of FishEye. For example, browsing for a file to add to a new review (see Selecting the Files for the Review).
  - Improved performance of native repository access over the lightSCM plugins. The lightSCM plugins retrieve data on demand from the underlying repository, rather than using caches and indexes like FishEye and native repository access. Hence, Crucible with native repository access, whilst requiring an initial indexing phase, will be faster than Crucible with lightSCM plugins during day-to-day operations.
  - Native repository access allows for migration to a full FishEye license in future, if desired. Your repositories can simply be re-indexed for full FishEye functionality and existing reviews will then be available on the full repository.

To change over from lightSCM plugins to native repository access,

1. Disable your lightSCM plugins via the Crucible Administration Console ('Plugins' link under the 'Systems Settings' section in the left menu).
2. Do not disable the SCM plugins for connecting to a Confluence instance or a file system. Native repository access does not include functionality to connect these (nor does standalone FishEye), hence you will still need to use plugins.
3. Add native repositories for any repositories that are currently connected via lightSCM plugins. See the FishEye documentation.
4. If you are using Subversion or Perforce, we recommend that you set a "start" revision for the changeover, unless you need to review old code. This will eliminate the need for native repository support to index old repository activity, getting you up and running quickly.

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, visit support.atlassian.com, log in (creating an account if need be) and create an issue under Crucible. Our friendly support engineers will get right back to you with an answer.

**Bug Fixing Policy**

**Summary**

- Atlassian Support will help with workarounds and bug reporting.
- Critical bugs will generally be fixed in the next maintenance release.
- Non critical bugs will be scheduled according to a variety of considerations.

**Raising a Bug Report**

Atlassian Support is eager and happy to help verify bugs — we take pride in it! Please open a support request in our support system providing as much information as possible about how to replicate the problem you are experiencing. We will replicate the bug to verify, then lodge the report for you. We'll also try to construct workarounds if they're possible.

Customers and plugin developers are also welcome to open bug reports on our issue tracking systems directly. Use http://jira.atlassian.com for the stand-alone products and http://studio.atlassian.com for JIRA Studio.

When raising a new bug, you should rate the priority of a bug according to our JIRA usage guidelines. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

**How Atlassian Approaches Bug Fixing**

Maintenance (bug fix) releases come out more frequently than major releases and attempt to target the most critical bugs affecting our customers. The notation for a maintenance release is the final number in the version (ie the 1 in 3.0.1).

If a bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions) then it will be fixed in the next maintenance release provided that:

- The fix is technically feasible (i.e. it doesn't require a major architectural change).
- It does not impact the quality or integrity of a product.

For non-critical bugs, the developer assigned to fixing bugs prioritises the non-critical bug according to these factors:

- How many of our supported configurations are affected by the problem.
- Whether there is an effective workaround or patch.
- How difficult the issue is to fix.
- Whether many bugs in one area can be fixed at one time.

The developers responsible for bug fixing also monitor comments on existing bugs and new bugs submitted in JIRA, so you can provide feedback in this way. We give high priority consideration to security issues.

When considering the priority of a non-critical bug we try to determine a 'value' score for a bug which takes into account the severity of the bug from the customer's perspective, how prevalent the bug is and whether roadmap features may render the bug obsolete. We combine this with a complexity score (i.e. how difficult the bug is). These two dimensions are used when developers self serve from the bug pile.

**Further reading**

See How to Get Legendary Support from Atlassian for more support-related information.

**How to Report a Security Issue**
Finding and Reporting a Security Vulnerability

If you find a security bug in the product, please open an issue on http://jira.atlassian.com in the relevant project.

- Set the priority of the bug to 'Blocker'.
- Provide as much information on reproducing the bug as possible.
- Set the security level of the bug to 'Developer and Reporters only'.

All communication about the vulnerability should be performed through JIRA, so that Atlassian can keep track of the issue and get a patch out as soon as possible.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

New Features Policy

Summary

- We do not publish roadmaps.
- Product Managers review our most popular voted issues on a regular basis.
- We schedule features based on a variety of factors.
- Our Atlassian Bug Fixing Policy is distinct from our Feature Request process.
- Atlassian provides consistent updates on the top 20 feature/improvement requests (in our issue tracker systems).

How to Track what Features are Being Implemented

When a new feature or improvement is scheduled, the 'fix-for' version will be indicated in the JIRA issue. This happens for the upcoming release only. We maintain roadmaps for more distant releases internally, but because these roadmaps are often pre-empted by changing customer demands, we do not publish them.

How Atlassian Chooses What to Implement

In every major release we aim to implement highly requested features, but it is not the only determining factor. Other factors include:

- Direct feedback from face to face meetings with customers, and through our support and sales channels.
- Availability of staff to implement features.
- Impact of the proposed changes on the application and its underlying architecture.
- How well defined the requested feature is (some issues gain in popularity rapidly, allowing little time to plan their implementation).
- Our long-term strategic vision for the product.

How to Contribute to Feature Development

Influencing Atlassian’s release cycle

We encourage our customers to vote on feature requests in JIRA. The current tally of votes is available online in our issue tracking systems, http://jira.atlassian.com and http://studio.atlassian.com. Find out if your improvement request already exists. If it does, please vote for it. If you do not find it, create a new feature or improvement request online.

Extending Atlassian Products

Atlassian products have powerful and flexible extension APIs. If you would like to see a particular feature implemented, it may be possible to develop the feature as a plugin. Documentation regarding the plugin APIs is available. Advice on extending either product may be available on the user mailing-lists, or at our community forums.

If you require significant customisations, you may wish to get in touch with our partners. They specialise in extending Atlassian products and can do this work for you. If you are interested, please contact us.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.
Patch Policy

Atlassian will only provide software patches in extremely unusual circumstances. If a problem has been fixed in a newer release of the product, Atlassian will request that you upgrade your instance to fix the issue. If it is deemed necessary to provide a patch, a patch will be provided for the current release and the last maintenance release of the last major version (e.g. JIRA 3.13.5) only.

Patches are issued under the following conditions:

- The bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions).
- A patch is technically feasible (i.e., it doesn't require a major architectural change) OR
- The issue is a security issue, and falls under our Security Patch Policy.

Atlassian does not provide patches for non-critical bugs.

Provided that a patch does not impact the quality or integrity of a product, Atlassian will ensure that patches supplied to customers are added to the next maintenance release. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

Patches are generally attached to the relevant http://jira.atlassian.com issue.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Advisory Publishing Policy

Publication of Security Advisories

When a security vulnerability in an Atlassian product is discovered and resolved, Atlassian will inform customers through the following mechanisms:

- We will post a security advisory in the latest documentation of the affected product at the same time as releasing a fix for the vulnerability. This applies to all security advisories, including severity levels of critical, high, medium and low.
- We will send a copy of all security advisories to the 'Technical Alerts' mailing list for the product concerned.
  Note: To manage your email subscriptions and ensure you are on this list, please go to my.atlassian.com and click 'Email Prefs' near the top right of the page.
- If the person who reported the vulnerability wants to publish an advisory through some other agency, such as CERT, we will assist in the production of that advisory and link to it from our own.

Early warning of critical security vulnerabilities:

- If the vulnerability is rated critical (see our criteria for setting severity levels) we will send an early warning to the 'Technical Alerts' mailing list approximately one week before releasing the fix. This early warning is in addition to the security advisory itself, described above.
- However, if the vulnerability is publicly known or being exploited, we will release the security advisory and patches as soon as possible, potentially without early warning.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Patch Policy

Product Security Patch Policy

Atlassian makes it a priority to ensure the 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 Studio, since Studio will always be patched by Atlassian without additional notifications.

Critical vulnerabilities
When a **Critical** security vulnerability is discovered by Atlassian or reported by a third party, Atlassian will do all of the following:

- Issue a new, fixed release for the current version of the affected product as soon as possible, usually in a few days.
- Issue a binary patch for the current release.
- Issue a binary patch for the latest maintenance release of the previous version of the product.
- Patches for older versions or releases normally will not be issued.

Patches will be attached to the relevant JIRA issue. You can use these patches as a "stop-gap" measure until you upgrade your installation in order to fully fix the vulnerability.

**Non-critical vulnerabilities**

When a security issue of a **High, Medium or Low** severity is discovered, Atlassian will do all of the following:

- Include the fix into the next scheduled release, both for the current and previous maintenance versions.
- Where practical, provide new versions of plugins or other components of the product that can be upgraded independently.

You should upgrade your installation in order to fix the vulnerability.

**Other information**

Severity level of vulnerabilities is calculated based on [Severity Levels for Security Issues](#).

Visit our general [Atlassian Patch Policy](#) as well.

**Examples**

**Example 1:** A critical severity vulnerability is found in a (hypothetical current release) JIRA 5.3.2. The last bugfix release in 5.2.x branch was 5.2.3. In this case, a patch will be created for 5.3.2 and 5.2.3. In addition, new bugfix releases, 5.3.3 and 5.2.4, which are free from this vulnerability, will be created in a few days.

**Example 2:** A high or medium severity vulnerability is found in the same release as in the previous example. The fix will be included into the currently scheduled releases 5.3.3 and 5.2.4. Release schedule will not be brought forward and no patches will be issued. If the vulnerability is in a plugin module, then a plugin upgrade package may still be supplied.

**Further reading**

See [How to Get Legendary Support from Atlassian](#) for more support-related information.

**Severity Levels for Security Issues**

**Severity Levels**

Atlassian security advisories include a severity level. This severity level is based on our self-calculated CVSS score for each specific vulnerability. CVSS is an industry standard vulnerability metric. You can learn more about CVSS at [FIRST.org](#) web site.

CVSS scores are mapped into the following severity ratings:

- Critical
- High
- Moderate
- 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 include:

- 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, it is advised that you patch or upgrade as soon as possible, unless you have other mitigating measures in place. For example, if your installation is not accessible from the Internet, this may be a mitigating factor.

**Severity Level: High**

Vulnerabilities that score in the High range usually have the following characteristics:

- The vulnerability is difficult to exploit.
- Exploitation does not result in elevated privileges.
- Exploitation does not result in a significant data loss.

**Severity Level: Moderate**

Vulnerabilities that score in the Moderate range usually have the following characteristics:

- Denial of service vulnerabilities that are difficult to set up.
- Exploits that require an attacker to reside on the same local network as the victim.
- Vulnerabilities that affect only nonstandard configurations or obscure applications.
- Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics.
- Vulnerabilities where exploitation provides only very limited access.

**Severity Level: Low**

Vulnerabilities in the Low range typically have very little impact on an organisation's business. Exploitation of such vulnerabilities usually requires local or physical system access.

**Further reading**

See [How to Get Legendary Support from Atlassian](https://confluence.atlassian.com/doc/how-to-get-legendary-support-from-atlassian/) for more support-related information.

---

**Crucible Development Hub**

This page is deprecated. Please see the new [Developer Documentation Space](https://developer.atlassian.com/).

**Documentation for Crucible Development**

This page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

Here you'll find everything you need to code up a storm with Crucible. This includes guides for environment set-up, building a project, plugin creation, and real-world examples you can try.

<table>
<thead>
<tr>
<th><strong>How to Build a Crucible Plugin</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>How to Build a Crucible Plugin</strong></td>
</tr>
<tr>
<td>start here to learn how to set up your development environment, create a plugin template and start coding.</td>
</tr>
</tbody>
</table>

- Development Platform for Crucible
- Crucible API Javadocs
- Crucible REST API

**Crucible's URL Structure**
This page contains information about the Crucible URL structure for plugin developers. Knowing the structure, you will be able to construct hyperlinks for use in plugins or gadgets and find API specifications for your version of Crucible.

On this page:

- Create Review
- Crucible Reviews
- Crucible Projects
- Search Crucible Reviews
- Search Crucible Review Comments

There is also a page about the FishEye structure.

Create Review

This creates a Crucible review on the specified changeset and repository.

In the example below, insert the desired changeset ID in place of "MY_CSID" and the desired repository name in place of "RENAME".

Basic form

```
/cru/create?csid=MY_CSID&repo=RENAME
```

Example with typical values

```
http://example.com/crucible/cru/create?csid=18905&repo=CLOV
```

Crucible Reviews

This opens a Crucible review page with the specified review key.

In the example below, insert the desired review key in place of "MY_REVIEW_KEY".

Basic form

```
/cru/REVIEW_KEY
```

Example with typical values

```
http://example.com/crucible/cru/CR-1
```

Crucible Projects

This opens a Crucible project page with the specified project key.

In the example below, insert the desired project key in place of "MY_PROJECT_KEY".

Basic form

```
/cru/browse/MY_PROJECT_KEY
```

Example with typical values
Search Crucible Reviews

This searches Crucible reviews with the specified search string.

In the example below, insert the desired string that you want to match against review titles in place of "QUERYSTRING".

**Basic form**

```
/cru/search?query=QUERYSTRING
```

**Example with typical values**

```
http://example.com/crucible/cru/search?query=november-audit
```

Search Crucible Review Comments

This searches comments on reviews in Crucible.

In the example below, you would insert your search string in place of the word "TEST".

**Basic form**

```
/cru/commentSearch?search.text=TEST
```

**Example with typical values**

```
http://example.com/crucible/cru/commentSearch?search.text=imho
```

Looking for a page on the FishEye URL structure? Click here.

**Crucible REST API**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

These pages contain information relating to the REST API for Crucible.

A list of available services and a detailed example page are currently documented.

**Crucible REST API documentation:**

- Crucible REST API Usage Example
- Conditional Get — Conditional Get allows lightweight polling of resources.
- Data Types — Definitions of data types used by the REST API.
- Project Service — Provides access to the projects defined in a Crucible instance.
- Repository Service — Provides information about the repositories configured in a Crucible instance.
- Review Service — The Review Service allows you to list, examine, create and modify reviews.

**Crucible REST API Usage Example**
This page describes using the Crucible REST API to retrieve comments from reviews in Crucible. It's an overview of using the API, not a comprehensive reference.

The Crucible REST API provides a reference for all the REST operations supported by Crucible.

On this page:
- Authentication
- Retrieving Reviews
- Retrieving Reviews in a Specific State
- Retrieving Comments From a Review
- Retrieving Properties of a File Under Review
- Creating a New Review
- JSON
- Retrieving a Specific Review
- Making a JSON Request

The Crucible REST API lives under the URL `http://HOSTNAME:PORT/CONTEXT/rest-service/`, where `HOSTNAME:PORT` is the IP address and port of your FishEye instance and `CONTEXT` is the web application context it is deployed under.

This page doesn't assume any particular REST client is being used – it just discusses the URLs to use and the responses which they will give. The information returned is in XML format.

This page assumes Crucible 1.6 – the examples (in particular JSON support) may not work with earlier versions.

**Authentication**

Requests to the REST API are simply HTTP requests, which can use any of the normal Crucible authentication methods. An unauthenticated request will execute as the anonymous user.

Authentication options are:
- The normal Crucible login cookie. A cookie named ‘remember’ in the request with the token returned by the REST authentication service on `http://HOSTNAME:PORT/rest-service/auth-v1/login?userName=jim&password=jimspassword`. This will return `<?xml version="1.0" encoding="UTF-8" standalone="yes"?><loginResult><token>jim:2:4455f9a4387298a83aae6902e8843f89</token></loginResult>.
- Trusted Applications. If Crucible trusts the application which is making the request, the user logged in to the trusted application will be authenticated in Crucible.
- Crowd. If Crucible is configured to use Crowd, then a request containing Crowd authentication will authenticate the Crowd user in Crucible.
- Basic Authentication. An RFC 2617 Basic Authentication header.

**Retrieving Reviews**

This example will use the reviews service, at the URL `http://HOSTNAME:PORT/rest-service/reviews-v1`. A simple `GET` on this URL will return every review in the system. The results will look like this:

```xml
<reviews>
  <reviewData>
    <author>pmcneil</author>
    <creator>pmcneil</creator>
    <description>14699: CRUC-230: allow links to be removed
14698: CRUC-230: don't allow linking cycles</description>
    <moderator>pmcneil</moderator>
    <name>CRUC-214: Generate comment/defect and open review report graphs</name>
    <permaId>
      <id>CR-FE-1</id>
    </permaId>
    <projectKey>CR-FE</projectKey>
    <repoName>FE</repoName>
    <state>Review</state>
  </reviewData>
  ...
</reviews>
```
Retrieving Reviews in a Specific State

If you don't want to retrieve every review, you can specify a value for the state parameter:

http://HOSTNAME:PORT/rest-service/reviews-v1?state=Review,Summarize to retrieve only those reviews in particular states.

The request only returns those reviews that the authenticated user is allowed to see.

Once you have the reviews you can use their permaId to get more details, so:

http://HOSTNAME:PORT/rest-service/reviews-v1/CR-FE-1 will return a single reviewData element, identical to the one shown above.

Retrieving Comments From a Review

URLs like http://HOSTNAME:PORT/rest-service/reviews-v1/CR-FE-1/thing will return information about thing records belonging to the review.

So http://HOSTNAME:PORT/rest-service/reviews-v1/CR-FE-1/comments returns all the comments in the review:

```xml
<comments>
  <versionedComment>
    <createDate>2008-03-03T22:00:00.920+11:00</createDate>
    <defectApproved>false</defectApproved>
    <defectRaised>false</defectRaised>
    <deleted>false</deleted>
    <draft>false</draft>
    <message>why use roll instead of add??</message>
    <permaId>
      <id>CMT:200</id>
    </permaId>
    <reviewItemId>
      <id>CFR-281</id>
    </reviewItemId>
    <user>mquail</user>
    <fromLineRange>196-199, 230-233</fromLineRange>
    <toLineRange>206-209, 240-251</toLineRange>
  </versionedComment>
  ... more versioned comments ...
  <generalComment>
    <createDate>2008-03-25T17:15:20.380+11:00</createDate>
    <defectApproved>false</defectApproved>
    <defectRaised>true</defectRaised>
    <deleted>false</deleted>
    <draft>false</draft>
    <message>when there are no comments in the last week the vertical axis shows -5 as the starting point</message>
    <user>pmcneil</user>
  </generalComment>
  ... more general comments ...
</comments>
```

Retrieving Properties of a File Under Review

If you need more information about the file a versioned comment was on, the URL:

http://HOSTNAME:PORT/rest-service/reviews-v1/CR-FE-1/reviewitems/CFR-281 gives more details:

```xml
<fisheyeReviewItemData>
  <permaId>
    <id>CFR-281</id>
  </permaId>
  <fromPath></fromPath>
  <fromRevision></fromRevision>
  <repositoryName>FE</repositoryName>
  <toPath>branches/iteration03/src/java/com/cenqua/crucible/reports/CommentsDefects/CommentDatasetMaker.java</toPath>
  <toRevision>13947</toRevision>
</fisheyeReviewItemData>
```
That particular review item is a new file, so the `fromPath` and `fromRevision` elements are empty.

**Creating a New Review**

To create a review, do a POST call to the reviews url (http://HOSTNAME:PORT/rest-service/reviews-v1) with the following XML document as request body (note that you need to be authenticated be able to create a new review, so use Basic HTTP authentication for this call):

```xml
<?xml version="1.0"?>
<createReview>
  <reviewData>
    <author> <!-- required element -->
      <userName>joe</userName>
    </author>
    <creator> <!-- required element -->
      <userName>fred</userName>
    </creator>
    <moderator> <!-- required element -->
      <userName>erik</userName>
    </moderator>
    <description>These is the Statement of Objectives.</description>
    <name>Title of the new review</name> <!-- required element -->
    <projectKey>CR</projectKey> <!-- required element -->
    <allowReviewersToJoin>true</allowReviewersToJoin>
  </reviewData>
</createReview>
```

**JSON**

As of Crucible 1.6.3, JSON serialization is supported for REST requests and responses. Using the `Accept` request header, clients can specify whether the response document should be encoded in XML or JSON. Unless specified differently, Crucible will respond using XML and will interpret requests as XML. Crucible will always include the `Content-Type` header in the response to identify the encoding. Likewise, when a client sends a JSON request document, it must use the `Content-Type: application/json` header. It is possible to use a different encoding for the request and the response.

**Note**

JSON support is currently experimental.

**Retrieving a Specific Review**

To retrieve the contents of a specific review as a JSON document, rather than XML, include the `Accept: application/json` header in your HTTP request. The example below includes the HTTP headers of both the request and the response to illustrate this:

**Request:**

```
GET /rest-service/reviews-v1/CR-3/details HTTP/1.1
Host: localhost:8060
Authorization: Basic am9lOm9vZQ==
Accept: application/json
```

**Response:**
When sending a request document using JSON, include the `Content-Type: application/json` header in the HTTP request. The example below creates a new review using JSON. Again, the relevant HTTP request and response headers are included:

```
POST /rest-service/reviews-v1 HTTP/1.1
Host: localhost:8060
Content-Length: 269
Authorization: Basic cm9lOmF5SG9J
Accept: application/json
Content-Type: application/json

{"createReview":
{"reviewData": {
 "allowReviewersToJoin":false,
 "author":{"userName":"joe"},
 "creator":{"userName":"joe"},
 "moderator":{"userName":"matt"},
 "description":null,
 "metricsVersion":1,
 "name":null,
 "permaId":null,
 "projectKey":null,
 "state":null,
 "actions": [null,
{"name":"action:deleteReview"},
{"name":"action:rejectReview"},
{"name":"action:abandonReview"},
{"name":"action:modifyReviewFiles"},
{"name":"action:approveReview"},
{"name":"action:recoverReview"},
{"name":"action:submitReview"},
{"name":"action:createReview"},
{"name":"action:viewReview"},
{"name":"action:reopenReview"},
{"name":"action:closeReview"},

"generalComments":null,
"reviewItems":null,
"transitions": [null,
{"name":"action:approveReview"},
{"name":"action:abandonReview"}]
}],
"versionedComments":null
}}
```

Response:
Conditional Get

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

Conditional Get allows lightweight polling of resources.

The REST API makes Crucible available to remote applications. Depending on the type of application, it can be quite common to poll a certain resource periodically to be able to detect changes. For example, an application may request `/reviews-v1/filter/Review` at regular intervals and notify the user when a new review was added.

However, polling a server constantly for potential updates could cause undesired overhead, especially when the response is large. To better facilitate applications that need to poll frequently, Crucible implements HTTP Conditional Get.

**Conditional Get**

With Conditional Get, the server keeps track of when the last change was made to a resource and sends this timestamp along in each HTTP response ("Last-Modified: Wed, 01 Oct 2008 03:37:58 GMT"). At the same time, a client that understands Conditional Get and polls the same resource periodically, will in turn keep track of the Last-Modified timestamp of each resource it has requested and send it along as a request header at every request ("If-Modified-Since: Mon, 29 Sep 2008 06:47:04 GMT").

When the resource has not been modified since the last time the client requested it (Last-Modified <= If-Modified-Since), the server will not serve the request, but return status 204 "Not Modified" with an empty response body. If the resource was modified, the server will respond normally (200 with the resource in the body).

Note that Crucible also sends the ETag response header along with Last-Modified. The ETag header contains a checksum of the response document and allows the client to detect changes even when the Last-Modified time did not change. A client that implements Conditional Get should send the value of the ETag response header in the If-None-Match request header.

For more information on HTTP Conditional Get, please refer to the HTTP specification.

**Compatibility**

Servers implementing Conditional Get are completely compatible with clients that don't understand it and vice versa.

**Data Types**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

Definitions of data types used by the REST API.

- ReviewData
- ReviewItemData
- DetailedReviewData
ReviewData

Contains basic information about a review.

Sample XML:

```
<reviewData>
  <allowReviewersToJoin>false</allowReviewersToJoin>
  <author>
    <displayName>Matt Quail</displayName>
    <userName>matt</userName>
  </author>
  <createDate>2008-08-25T12:38:14.603+1000</createDate>
  <creator>
    <displayName>Matt Quail</displayName>
    <userName>matt</userName>
  </creator>
  <description>Review things and stuff</description>
  <metricsVersion>1</metricsVersion>
  <moderator>
    <displayName>Matt Quail</displayName>
    <userName>matt</userName>
  </moderator>
  <name>Test review 1</name>
  <permaId><id>CR-1</id></permaId>
  <projectKey>CR</projectKey>
  <state>Review</state>
</reviewData>
```

Sample JSON:

```
{"reviewData": {
  "allowReviewersToJoin":false,
  "author":{
    "displayName":"Matt Quail","userName":"matt"},
  "createDate":"2008-10-27T09:50:05.064+1100",
  "creator":{
    "displayName":"Matt Quail","userName":"matt"},
  "description":"Review things and stuff",
  "metricsVersion":1,
  "name":"Test review 1",
  "permaId":{
    "id":"CR-3"},
  "projectKey":"CR",
  "state":"Draft"
}}
```

ReviewItemData

Describes a single item that is under review. An item can represent the changes between two or more revisions of a file in a source repository, a change that was uploaded to Crucible as a unified diff or patch file, or it can represent any arbitrary file uploaded and attached to a review. Below are three examples of reviewItemData in XML, followed by the same three in JSON:

Sample XML:
Sample JSON
The above section contains three reviewItemData instances that illustrate the use of the individual elements. The main elements are the <to/> and <from/> elements. These describe the two revisions of a review item where the to.. is the most recent and from.. the oldest revision of the item that is under review.

When using Iterative Reviewing, an item can contain more than two file revisions. The <revisions/> element contains information on every revision under review, while <to../> and <from../> always point to the first and the last revisions (the cumulative changes). By default, the <revisions/> element is collapsed and contains the size= attribute that indicates the total number of file revisions in the review item. To expand this list, use the ?append=revisions url parameter, e.g.:


Note that when a new file is added, it will not have the from.. elements and likewise, when a file gets removed, it will lack the to.. elements. The <to../> and <from../> elements also include urls that point to the file contents hosted in Crucible. These are the <fromContentUrl/> and <toContentUrl/> elements. These are relative URLs that come after the web application context, so for example to download the file from the third reviewItemData, access: http://HOSTNAME:PORT/CONTEXT/cru/CR-4/rawcontent\52\jackrabbit-scm-plugin.tgz

Note that <fromContentUrl/> and <toContentUrl/> only apply to either uploaded files or revisions on files in one of the Crucible repositories. Uploaded patch files lack these elements because a unified diff file usually only contains the sections of two files that were changed,
but not the code that was unchanged. As a result, Crucible is unable to provide links for the individual files. Instead, the \texttt{<patchUrl/>} element contains a relative link to the original patch file that was uploaded by the creator of the review.

**DetailedReviewData**

Note that the \texttt{reviewItems} element is empty when multiple reviews are retrieved via REST. To include the \texttt{reviewItems} in a \texttt{detailedReviewData} structure you must retrieve a single review via the URL \texttt{/rest-service/reviews-v1/<review id>/details}.

Sample XML:

```xml
<detailedReviewData>
  <allowReviewersToJoin>false</allowReviewersToJoin>
  <author>
    <displayName>Matt Quail</displayName>
    <userName>matt</userName>
  </author>
  <createDate>2008-09-16T10:50:26.862+1000</createDate>
  <creator>
    <displayName>Matt Quail</displayName>
    <userName>matt</userName>
  </creator>
  <description/>
  <metricsVersion>1</metricsVersion>
  <moderator>
    <displayName>Matt Quail</displayName>
    <userName>matt</userName>
  </moderator>
  <permaId>
    <id>CR-1</id>
  </permaId>
  <projectKey>CR</projectKey>
  <state>Draft</state>
  <actions>
    <actionData>
      <name>action:abandonReview</name>
    </actionData>
    <actionData>
      <name>action:closeReview</name>
    </actionData>
    <actionData>
      <name>action:submitReview</name>
    </actionData>
    <actionData>
      <name>action:reopenReview</name>
    </actionData>
    <actionData>
      <name>action:summarizeReview</name>
    </actionData>
    <actionData>
      <name>action:rejectReview</name>
    </actionData>
    <actionData>
      <name>action:deleteReview</name>
    </actionData>
    <actionData>
      <name>action:approveReview</name>
    </actionData>
    <actionData>
      <name>action:modifyReviewFiles</name>
    </actionData>
    <actionData>
      <name>action:viewReview</name>
    </actionData>
    <actionData>
      <name>action:commentOnReview</name>
    </actionData>
  </actions>
</detailedReviewData>
```
<name>action:recoverReview</name>
</actionData>
<actionData>
<name>action:createReview</name>
</actionData>
</actions>
<generalComments/>
<reviewItems>
<reviewItem>
<permId>
  <id>CFR-1</id>
</permId>
<authorName>admin</authorName>
<commitDate>2008-08-27T10:19:17.000+1000</commitDate>
<commitType>Modified</commitType>
<fileType>File</fileType>
<fromPath>ds/Home</fromPath>
<fromRevision>1</fromRevision>
<repositoryName>localhost</repositoryName>
<toPath>ds/Home</toPath>
<toRevision>2</toRevision>
</reviewItem>
<reviewItem>
<permId>
  <id>CFR-2</id>
</permId>
<authorName>tomd</authorName>
<commitDate>2008-09-09T16:42:28.786+1000</commitDate>
<commitType>Added</commitType>
<fileType>File</fileType>
<fromPath/>
<fromRevision/>
<repositoryName>mylocalsvn</repositoryName>
<toPath>aaa/bbb/qqq.txt</toPath>
<toRevision>3</toRevision>
</reviewItem>
</reviewItems>
<reviewers/>
<transitions>
<transitionData>
<name>action:approveReview</name>
</transitionData>
<transitionData>
<name>action:abandonReview</name>
</transitionData>
Error

When a request cannot be serviced properly due to either a server-side problem, or invalid client input, Crucible will return an error document, combined with an HTTP status code other than 200. This XML document contains a number of elements that describe the problem. Note that the HTTP status code distinguishes between client- and server-side causes.

Below is the error that is returned when asking for a non existent resource. The status code for this response is 404 "Document Not Found". Other possible status codes for error responses include 400 "Bad Request" (for example when a request contains an invalid POST body) and 403 "Forbidden" (when accessing a resource without permission).
Sample XML:

```xml
<error>
  <code>NotFound</code>
  <message>Unknown metrics version: 45</message>
  <stacktrace>com.atlassian.crucible.spi.services.NotFoundException: Unknown metrics version: 45
    at com.atlassian.crucible.spi.impl.DefaultReviewService.getMetrics(DefaultReviewService.java:689)
    at com.atlassian.crucible.spi.rpc.RestReviewService$22.doGet(RestReviewService.java:645)
    at com.atlassian.crucible.spi.rpc.RestReviewService$22.doGet(RestReviewService.java:644)
    at com.atlassian.crucible.spi.rpc.ConditionalGet.doConditionalGet(ConditionalGet.java:46)
    at com.atlassian.crucible.spi.rpc.RestReviewService.getMetrics(RestReviewService.java:643)
    ...
</stacktrace>
</error>
```

Sample JSON:

```json
{
  "code": "NotFound",
  "message": "No review exists with permId 'CR-333'",
  "stacktrace": "com.atlassian.crucible.spi.services.NotFoundException: Unknown metrics version: 45
    at com.atlassian.crucible.spi.impl.DefaultReviewService.getMetrics(DefaultReviewService.java:689)
    at com.atlassian.crucible.spi.rpc.RestReviewService$22.doGet(RestReviewService.java:645)
    at com.atlassian.crucible.spi.rpc.RestReviewService$22.doGet(RestReviewService.java:644)
    at com.atlassian.crucible.spi.rpc.ConditionalGet.doConditionalGet(ConditionalGet.java:46)
    at com.atlassian.crucible.spi.rpc.RestReviewService.getMetrics(RestReviewService.java:643)
    ...
  "
}
```

Project Service

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

Provides access to the projects defined in a Crucible instance. At present this interface is read-only.

**Get Project List**

**Method:** GET  
**URL:** /projects-v1

**Description:**  
Returns a list of projects.  

**Status Code:**  
200 (OK) on success

**Example XML:**
<projects>
  <projectData>
    <allowReviewersToJoin>false</allowReviewersToJoin>
    <id>1</id>
    <key>CR</key>
    <name>Default Project</name>
    <permissionSchemeId>1</permissionSchemeId>
  </projectData>
  ...
</projectData>
</projects>

Example JSON:

```json
{"projects":{"projectData":{
  "allowReviewersToJoin":false,
  "id":1,
  "key":"CR",
  "name":"Default Project",
  "permissionSchemeId":1}
}}
```

Repository Service

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

Provides information about the repositories configured in a Crucible instance.

Get Repositories

**Method:** GET

**URL:**

/repositories-v1

**Description:**

Get a list of all the repositories.

**Status Code:**

200 (OK) on success.

**Example XML:**

```xml
<repositories>
  <repoData>
    <enabled>true</enabled>
    <name>local</name>
    <type>svn</type>
  </repoData>
  <repoData>
    <enabled>true</enabled>
    <name>test</name>
    <type>cvs</type>
  </repoData>
</repositories>
```

**Example JSON:**
Get Repository By Name

Method: GET
URL:
/repositories-v1/<repositoryName>

Description:
Returns the repository of which the `name` attribute equals `repositoryName`.

Status Code:
200 (OK) on success

Example XML:

```xml
<svnRepositoryData>
  <enabled>true</enabled>
  <name>local</name>
  <type>svn</type>
  <path></path>
  <url>file:///Users/tomd/dev/svn/</url>
</svnRepositoryData>
```

Example JSON:

```json
{"svnRepositoryData": {
  "enabled":true,
  "name":"Local",
  "type":"svn",
  "path":",
  "url":"file:///\Users\ervzijst\var\repo"
}}
```

Review Service

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

The Review Service allows you to list, examine, create and modify reviews.
See the Data Types Page for the structure of the reviewData and detailedReviewData tags.

Click an item in the list below to see full details, all options and example code.

- Review Service - Reviews
  - Add Changeset To Review
  - Add Patch Revisions To Review
  - Create Review
  - Delete Review
  - Get All Reviews
  - Get All Reviews (limited)
  - Get Allowed Review Actions
  - Get Allowed Review Transitions
  - Get Review
  - Get Review Details
  - Get Review Details by Path
  - Get Review Details through Custom Filter Criteria
  - Get Reviews by Filter
  - Get Reviews by Path
  - Get Reviews through Custom Filter Criteria
  - Get Single Review Details
  - Get Version Info

- Review Service - Reviewers
  - Add Reviewers
  - Get Finished Reviewers
  - Get Incomplete Reviewers
  - Get Reviewers
  - Remove Single Reviewer

- Review Service - Review Items
  - Add Revision to Review
  - Get Review Items
  - Get Single Revision Details
  - Remove Revision from Review

- Review Service - Workflow
  - Close a Review
  - Complete a Review
  - Move a Review to a New State
  - Uncomplete a Review

- Review Service - Comments
  - Add a Reply to a Comment
  - Add Comment to a Review Item
  - Add General Comment to a Review
  - Delete a Comment
  - Delete a Reply
  - Get a Comment
  - Get Comments on a Review Item
  - Get Comments on Files
  - Get General Comments
  - Get Review Comments
  - Get the Replies to a Comment
  - Mark a Comment as Read
  - Mark a Comment as Leave Unread
  - Mark All Comments as Read
  - Publish a Draft Comment
  - Publish All Draft Comments
  - Update a Comment

- Review Service - Miscellaneous
  - Get Metrics

### Review Service - Comments

⚠️ The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

### On this page:

- Comments
- Add a Reply to a Comment
- Add Comment to a Review Item
- Add General Comment to a Review
- Delete a Comment
- Delete a Reply
- Get a Comment
- Get Comments on a Review Item
- Get Comments on Files
- Get General Comments
- Get Review Comments
- Mark a Comment as Read
- Mark a Comment as Leave Unread
- Mark all Comments as Read
- Get the Replies to a Comment
- Publish a Draft Comment
- Publish All Draft Comments
- Update a Comment

Comments

Add a Reply to a Comment

URL:

POST /reviews-v1/<review id>/comments/<comment id>/replies

Description:
Add a reply to an existing comment.

The POST data is a generalCommentData structure.

Status Code:
201 (Created) on success. The response will contain the Location response header with the URL of the newly created resource.

Add Comment to a Review Item

URL:

POST /reviews-v1/<review id>/reviewitems/<review item id>/comments

Description:
Add a comment to a review. Returns the completed versionedLineCommentData structure.

The POST data is a versionedLineCommentData structure.

Status Code:
201 (Created) on success. The response will contain the Location response header with the URL of the newly created resource.

Add General Comment to a Review

URL:

POST /reviews-v1/<review id>/comments

Description:
Add a general comment to a review. Returns the completed generalCommentData structure.

The POST data is a generalCommentData structure.

Status Code:
201 (Created) on success. The response will contain the Location response header with the URL of the newly created resource.

Delete a Comment

URL:
DELETE /reviews-v1/<review id>/comments/<comment id>

Description:
Remove an existing comment.
Status Code:
204 (No Content) on success.

Delete a Reply

URL:
DELETE /reviews-v1/<review id>/comments/<comment id>/replies/<reply id>

Description:
Delete a reply.
Status Code:
204 (No Content) on success.

Get a Comment

URL:
GET /reviews-v1/<review id>/comments/<comment id>

Description:
Retrieve an existing comment
Status Code:
200 (OK) on success.

Comment Read Status:
This attribute details the read status of the comment for the particular user that the request has been made for. There are three possible values:

- UNREAD: The comment has not been read.
- READ: The comment has been viewed.
- LEAVE_UNREAD: The comment has been read but marked by the user to be left for later referral.

Anonymous access requests and users who do not have permission to comment on a review always return the READ value.

Example XML Return Data:

```
<comments>
  <versionedLineCommentData>
    ...
  </versionedLineCommentData>
</comments>
```

Example JSON Return Data:

```
{"comments": [
  {"versionedCommentData": ...}
]}
```

Get Comments on a Review Item

URL:
GET /reviews-v1/<review id>/reviewitems/<review item id>/comments

**Description:**
Get all the comments made on a review item.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```
readonlyCommentsData
...
</versionedLineCommentData>
...
</comments>
```

**Example JSON Return Data:**
```
{"comments": [
{"versionedCommentData": ...}
{"versionedCommentData": ...}]
}
```

---

**Get Comments on Files**

**URL:**

GET /reviews-v1/<review id>/comments/versioned

**Description:**
Get all the versioned comments made on a review – that is, comments which are on a particular file in the review.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```
<comments>
<versionedLineCommentData>
...
</versionedLineCommentData>
...
</comments>
```

**Example JSON Return Data:**
```
{"comments": [
{"versionedCommentData": ...}
{"versionedCommentData": ...}]
}
```

---

**Get General Comments**

**URL:**

---
GET /reviews-v1/<review id>/comments/general

Description:
Get all the general comments made on a review – that is, comments which are not attached to a particular file in the review.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<comments>
  <generalCommentData>
    ...
  </generalCommentData>
</comments>
```

Example JSON Return Data:

```json
{"comments": [
  {
    "generalCommentData": {
      "createDate": "2008-10-30T16:32:51.032+1100",
      "defectApproved": false,
      "defectRaised": true,
      "deleted": false,
      "draft": false,
      "readStatus": "UNREAD",
      "message": "A general comment.",
      "metrics": [
        {
          "entry": {
            "key": "classification",
            "value": [{"configVersion": 1}, "Not conforming to standards"]
          }
        },
        {
          "entry": {
            "key": "rank",
            "value": [{"configVersion": 1}, "Minor"]
          }
        }
      ],
      "permIdAsString": "CMT:1",
      "replies": "",
      "user": {
        "displayName": "joe lowercase",
        "userName": "joe",
        "permId": {"id": "CMT:1"}
      }
    },
    {
      "generalCommentData": {
        "createDate": "2008-11-03T10:26:50.951+1100",
        "defectApproved": false,
        "defectRaised": false,
        "deleted": false,
        "draft": false,
        "message": "Another general comment.",
        "metrics": "",
        "permIdAsString": "CMT:4",
        "replies": "",
        "user": {
          "displayName": "joe lowercase",
          "userName": "joe",
          "permId": {"id": "CMT:4"}
        }
      }
    }
  ]
]}
```

Get Review Comments

URL:

```
GET /reviews-v1/<review id>/comments
```

Description:
Get all the comments made on a review. The `versionedLineCommentData` tag may contain `fromLineRange` and `toLineRange` tags, indicating that the comment was made against a specific range of lines.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<comments>
  ...
</comments>
```
This is a general comment.

This is a revision level defect.

This is a comment covering two lines of a revision.
Mark a Comment as Read

URL:
POST /reviews-v1/<review id>/comments/<comment id>/markAsRead

Description:
Marks a particular comment as read.

Status Code:
200 (OK) on success.

Mark a Comment as Leave Unread

URL:

POST /reviews-v1/<review id>/comments/<comment id>/markAsLeaveUnread

Description:
Marks a particular comment as leave unread.

Status Code:
200 (OK) on success.

Mark all Comments as Read

URL:

POST /reviews-v1/<review id>/comments/markAllAsRead

Description:
Marks all comments in the review which are in an unread state as read. Any comments which are in the leave unread state are not modified.

Status Code:
200 (OK) on success.

Get the Replies to a Comment

URL:

GET /reviews-v1/<review id>/comments/<comment id>/replies

Description:
Get the replies to an existing comment.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<comments>
  <generalCommentData>
    ...
  </generalCommentData>
  ...
</comments>
```

Example JSON Return Data:

```json
{"comments": [
  {"generalCommentData": ...}
  {"generalCommentData": ...}
]}
```
Publish a Draft Comment

POST /reviews-v1/<review id>/publish/<comment id>

Description:
Publish a draft comments on the review.

Status Code:
200 (OK) on success.

Publish All Draft Comments

POST /reviews-v1/<review id>/publish

Description:
Publish all the user's draft comments on the review.

Status Code:
200 (OK) on success.

Update a Comment

URL:

POST /reviews-v1/<review id>/comments/<comment id>

Description:
Update an existing comment. The readStatus attribute is ignored when updating a comment.

The POST data is a generalCommentData structure.

Status Code:
200 (OK) on success.

Review Service - Miscellaneous

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

On this page:

- Miscellaneous
  - Get Metrics

Miscellaneous

Get Metrics

URL:

GET /reviews-v1/metrics/<version>

Description:
Get the replies to an existing comment.

Status Code:
200 (OK) on success.
Example XML Return Data:

```xml
<?xml version="1.0"?>
<metrics>
  <metricsData>
    <configVersion>1</configVersion>
    <defaultValue>
      <name>Minor</name>
      <value xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:int">0</value>
    </defaultValue>
    <label>Ranking</label>
    <name>rank</name>
    <type>INTEGER</type>
    <values>
      <name>Major</name>
    </values>
    <values>
      <name>Minor</name>
      <value xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:int">0</value>
    </values>
  </metricsData>
  <metricsData>
    <configVersion>1</configVersion>
    <defaultValue>
      <name>Improvement desirable</name>
    </defaultValue>
    <label>Classification</label>
    <name>classification</name>
    <type>INTEGER</type>
    <values>
      <name>Missing</name>
    </values>
    <values>
      <name>Extra (superfluous)</name>
    </values>
    <values>
      <name>Ambiguous</name>
    </values>
    <values>
      <name>Inconsistent</name>
    </values>
    <values>
      <name>Improvement desirable</name>
    </values>
    <values>
      <name>Not conforming to standards</name>
    </values>
  </metricsData>
  <metricsData>
    <configVersion>1</configVersion>
    <defaultValue>
      <name>Risk-prone</name>
    </defaultValue>
    <label>Return Data</label>
    <name>returnData</name>
    <type>INTEGER</type>
    <values>
      <name>Successful</name>
    </values>
    <values>
      <name>Failed</name>
      <value xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:type="xs:int">0</value>
    </values>
  </metricsData>
</metrics>
```
<values>
  <name>Factually incorrect</name>
</values>

<values>
  <name>Not implementable</name>
</values>

<values>
  <name>Editorial</name>
</values>
Example JSON Return Data:

```json
{  "metrics": {    "metricsData": [      {        "configVersion": 1,        "defaultValue": {"name": "Minor", "value": 0},        "label": "Ranking",        "name": "rank",        "type": "INTEGER",        "values": [          {"name": "Major", "value": 1},          {"name": "Minor", "value": 0}        ]      },      {        "configVersion": 1,        "defaultValue": {"name": "Improvement desirable", "value": 5},        "label": "Classification",        "name": "classification",        "type": "INTEGER",        "values": [          {"name": "Missing", "value": 1},          {"name": "Extra (superfluous)", "value": 2},          {"name": "Ambiguous", "value": 3},          {"name": "Inconsistent", "value": 4},          {"name": "Improvement desirable", "value": 5},          {"name": "Not conforming to standards", "value": 6},          {"name": "Risk-prone", "value": 7},          {"name": "Factually incorrect", "value": 8},          {"name": "Not implementable", "value": 9},          {"name": "Editorial", "value": 10}        ]      ]  }}
```

Review Service - Reviewers

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

### Reviewers

**Add Reviewers**

**URL:**

```plaintext
POST /reviews-v1/<review id>/reviewers
```

**Description:**

Add new reviewers to the review. Send a string of comma separated user names.

**Status Code:**

200 (OK) on success.
Get Finished Reviewers

**URL:**

```
GET /reviews-v1/<review id>/reviewers/completed
```

**Description:**
Return a list of the reviewers who have completed the review.

**Status Code:**
200 (OK) on success.

**Example Return Data:**
Return value as `/reviews-v1/<review id>/reviewers`, but only completed reviewers are included.

Get Incomplete Reviewers

**URL:**

```
GET /reviews-v1/<review id>/reviewers/uncompleted
```

**Description:**
Return a list of the reviewers who have not yet completed the review.

**Status Code:**
200 (OK) on success.

**Example Return Data:**
Return value as `/reviews-v1/<review id>/reviewers`, but only incomplete reviewers are included.

Get Reviewers

**URL:**

```
GET /reviews-v1/<review id>/reviewers
```

**Description:**
Return a list of the reviewers participating in the review.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**
```
<reviewers>
  <reviewer>
    <displayName>Conor MacNeill</displayName>
    <userName>conor</userName>
    <completed>false</completed>
  </reviewer>
  ... more reviewers ...
</reviewers>
```

**Example JSON Return Data:**
```json

```
Remove Single Reviewer

URL:
DELETE /reviews-v1/<review id>/reviewers/<username>

Description:
Remove a reviewer from a review.

Status Code:
204 (No Content) on success.

Review Service - Review Items

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

On this page:

- Review Items
  - Add Revision to Review
  - Get Review Items
  - Get Single Revision Details
  - Remove Revision from Review

Review Items

Add Revision to Review

URL:
POST /reviews-v1/<review id>/reviewitems

Description:
Add a revision to a review. Send a reviewItem with the repository name, and from and to paths and revisions specified. Other values can be omitted. This returns the completed reviewItem structure.

Status Code:
201 (Created) on success. The response will contain the Location response header with the URL of the newly created resource and the reviewItemData document in the response body.

Get Review Items

URL:
GET /reviews-v1/<review id>/reviewitems

Description:
Get a list of the items in a review.
Status Code:
200 (OK) on success.

Example XML Return Data:

```
<reviewItems>
    <reviewItem>
        <permId>
            <id>CFR-1</id>
        </permId>
        <authorName>tomd</authorName>
        <commitDate>2008-01-29T14:41:43.202+1100</commitDate>
        <commitType>Modified</commitType>
        <fileType>File</fileType>
        <toContentUrl>/cru/CR-4/rawcontent/53/foo.txt</toContentUrl>
        <fromPath>/path/to/file.txt</fromPath>
        <fromRevision>21</fromRevision>
        <repositoryName>local</repositoryName>
        <toContentUrl>/cru/CR-4/rawcontent/51/foo.txt</toContentUrl>
        <toPath>/path/to/file.txt</toPath>
        <toRevision>22</toRevision>
    </reviewItem>
    ... more reviewItems ...
</reviewItems>
```

Example JSON Return Data:

```
{"reviewItems": [ { "permId": { "id": "CFR-4" },
    "authorName": "ervzijst",
    "commitDate": "2008-10-16T17:19:52.119+1000",
    "commitType": "Modified",
    "fileType": "File",
    "fromContentUrl": "/cru/CR-4/rawcontent/53/foo.txt",
    "fromPath": "path/to/file.txt",
    "fromRevision": "3",
    "repositoryName": "Local",
    "toContentUrl": "/cru/CR-4/rawcontent/51/foo.txt",
    "toPath": "path/to/file.txt",
    "toRevision": "13"
},
... more reviewItems ...
]}
```

Get Single Revision Details

URL:
```
GET /reviews-v1/<review id>/reviewitems/<review item id>
```

Description:
Get the details of a single revision in a review. Returns the `reviewItem` structure for the item.

Status Code:
200 (OK) on success.

Remove Revision from Review

URL:
DELETE /reviews-v1/<review id>/reviewitems/<review item id>

**Description:**
Remove a revision from a review.

**Status Code:**
204 (No Content) on success.

---

**Review Service - Reviews**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

**On this page:**

- Reviews
  - Add Changeset To Review
  - Add Patch Revisions To Review
  - Upload Files To Review
  - Create Review
  - Delete Review
  - Get All Reviews
  - Get All Reviews (limited)
  - Get Allowed Review Actions
  - Get Allowed Review Transitions
  - Get Review
  - Get Review Details
  - Get Review Details by Path
  - Get Review Details through Custom Filter Criteria
  - Get Reviews by Filter
  - Get Reviews by Path
  - Get Reviews through Custom Filter Criteria
  - Get Single Review Details
  - Get Version Info

---

**Reviews**

**Add Changeset To Review**

**URL:**

POST /reviews-v1/<review id>/addChangeset

**Description:**
Add the revisions in a set of changesets to an existing review.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```xml
<addChangeset>
  <repository>aRepositoryName</repository>
  <changesets>
    <changesetData>
      <id>...the id...</id>
    </changesetData>
    ... more change sets ...
  </changesets>
</addChangeset>
```

**Example JSON Return Data:**
The response is the `reviewData` structure of the review.

**Add Patch Revisions To Review**

**URL:**

```
POST /reviews-v1/<review id>/addPatch
```

**Description:**
Add the revisions in a patch to an existing review.

**Status Code:**
200 (OK) on success.

**Example XML Request Data:**

```
<addPatch>
  <repository>aRepositoryName</repository>
  <patch>
    <![CDATA[
      ... text of patch goes here ...
    ]]>  
  </patch>
</addPatch>
```

**Example JSON Request Data:**

```
{"addPatch":{"repository":"aRepositoryName","patch":"... text of patch goes here ..."}}
```

The response is the `reviewData` structure of the review.

**Upload Files To Review**

**URL:**

```
POST /reviews-v1/<review id>/addFile
```

**Description:**
Uploads a local file to the review. In contrast to a patch, files can be either binary or text. Depending on the filetype, size and contents, Crucible may be able to display either parts, or the entire file in the review. It is possible to upload two versions of the file, in which case Crucible will display a diff and report that the file was modified. When only a single file is uploaded, Crucible treats the file as newly added.

This action returns the `ReviewData` document on success.

This resources uses `multipart form-data` to receive the file(s), character set indication and optional comments (it does not expect an XML document with embedded files, as that would require the client to first encode the files in Base64). Making a multipart form-data request can be done manually, but you will probably want to use a library. During testing it is inconvenient to let your browser generate the requests using the test html form below:
Example HTML Form for Testing REST-Based File Uploads

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head><title>Rest File Upload Test</title></head>
<body>
<!-- Point the URL to the review you want to upload to. -->
<form action="http://hostname:port/rest-service/reviews-v1/<review id>/addFile" enctype="multipart/form-data" method="POST">
<table>
<tr>
<td>File (required):</td>
<td><input name="file" type="file"/></td>
</tr>
<tr>
<td>Diff to (optional):</td>
<td><input name="diffFile" type="file"/></td>
</tr>
<tr>
<td>Character Set (optional):</td>
<td><input name="charset" type="text" value="UTF-8"/></td>
</tr>
<tr>
<td>Comments (optional):</td>
<td><input type="text" name="comments"/></td>
</tr>
<tr><td><input type="submit" value="Upload"/></td></tr>
</table>
</form>
</body>
</html>
```

The form requests understands the following 4 fields:

- **file** - the file to add to the review (required)
- **diffFile** - if supplied, Crucible will use this file as the base for a diff with file
- **charset** - if supplied, specifies the character set of the (text)file (when omitted, the server's default character set will be used)
- **comments** - optional user string that is stored along with the file

When uploading files, make sure you (or your http client library) supplies the proper `Content-Type` header. For text files, use "text/plain". Crucible will preserve the original file name.

**Status Code:**
201 (Created) on success.
The response is the `reviewItemData` structure describing the new item. Also, the `Location` response header is present and contains the PermId URL of the new item.

**Create Review**

**URL:**

```
POST /reviews-v1
```

**Description:**
Create a review. A review can be created in one of three ways(see examples).

**Status Code:**
201 (Created) on success. The response will contain the `Location` response header with the URL of the newly created resource.

**Example XML Request Data:**
1. An empty review, i.e. no revisions in the review. This uses the request below:
2. A patch review, containing diffs from a patch file, e.g. created by `svn diff >patch.txt` (note that the text of the patch must be properly escaped in the JSON object):

```
<createReview>
  <reviewData>
     ...
  </reviewData>
  <patch>
    <![CDATA[
    ... text of patch goes here ...
    ]]>}
  </patch>
</createReview>
```

```
{createReview: {
   "reviewData": {
      "allowReviewersToJoin":false,
      "author":{"userName":"joe"},
      "creator":{"userName":"joe"},
      "moderator":{"userName":"matt"},
      "description":"
      "metricsVersion":1,
      "name":"Stuff to review",
      "projectKey":"CR"}
   }
   ``
```

3. A review containing revisions from a set of changesets. XML/JSON:

```
<createReview>
  <reviewData>
     ...
  </reviewData>
</createReview>
```

```
{createReview: {
   "reviewData": {
      "allowReviewersToJoin":false,
      "author":{"userName":"joe"},
      "creator":{"userName":"joe"},
      "moderator":{"userName":"matt"},
      "description":"
      "metricsVersion":1,
      "name":"
      "projectKey":"CR"},
   }
   ``
```

Carbon Dioxide Information Analysis Center, 
Environmental Data and Information Division, 
Oak Ridge National Laboratory, 
Oak Ridge, Tennessee 37831, 
The Oak Ridge Institute for Science and Education, 
Operated by UT-Battelle, LLC for the U.S. Department of Energy. 
DOE/SC-0012615.
XML

```xml
<createReview>
  <reviewData>
    ...
  </reviewData>
  <changesets>
    <changesetData>
      <id>...the id...</id>
    </changesetData>
    ... more changesets ...
  </changesets>
</createReview>
```

JSON

```json
{
  createReview: {
    "reviewData": {
      "allowReviewersToJoin":false,
      "author":{
        "userName":"joe"},
      "creator":{
        "userName":"joe"},
      "moderator":{
        "userName":"matt"},
      "description":"
      "metricsVersion":1,
      "name":"readme",
      "projectKey":"CR"},
    "changesets": {
      "changesetData":[
        {"id":"234"},
        {"id":"237"}],
      "repository":"Local"}
  }
}
```

In all these cases, the reviewData structure shouldn't have the permaId or state attributes set. The response is a reviewData structure fully populated.

**Delete Review**

**URL:**

DELETE /reviews-v1/<id>

**Description:**
Delete a review. The review must have been abandoned.

**Status Code:**
204 (No Content) on success.

**Get All Reviews**

**URL:**

GET /reviews-v1?state=<states>

**Description:**
Get all reviews as a list of ReviewData structures. Note that this may return a lot of data, so using /reviews-v1/filter/<filter> (see below) is usually better. The state parameter is a comma separated list of state names from the set Draft, Approval, Review, Summarize, Closed, Dead, Rejected, Unknown.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**
Example JSON Return Data:

```json
{"reviews": {
"reviewData": [
{"allowReviewersToJoin":true,
...,
}], ...
}
}
```

**Get All Reviews (limited)**

**URL:**

```
GET /reviews-v1/details?state=<states>
```

**Description:**
Get all reviews as a list of DetailedReviewData structures. The state parameter is a comma separated list of state names from the set Draft, Approval, Review, Summarize, Closed, Dead, Rejected, Unknown.

Note that the reviewItems list in the detailedReviewData elements will not appear because this URL retrieves multiple reviews.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```xml
<reviews>
 <reviewData>
 ...,
 </reviewData>
 ...,
</reviews>
```

**Example JSON Return Data:**

```json
{"reviews": {
 "reviewData": [
  {"allowReviewersToJoin":true,
   ...
  }, ...
 ]
}
```

**Get Allowed Review Actions**

**URL:**

```
```
GET /reviews-v1/<id>/actions

Description:
Get a list of the actions which the current user is allowed to perform on the review. This shows actions the user has permission to perform - the review may not be in a suitable state for all these actions to be performed.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<actions>
  <actionData>
    <name>action:summarizeReview</name>
  </actionData>
  <actionData>
    <name>action:viewReview</name>
  </actionData>
  <actionData>
    <name>action:approveReview</name>
  </actionData>
  <actionData>
    <name>action:closeReview</name>
  </actionData>
  <actionData>
    <name>action:modifyReviewFiles</name>
  </actionData>
  <actionData>
    <name>action:rejectReview</name>
  </actionData>
  <actionData>
    <name>action:deleteReview</name>
  </actionData>
  <actionData>
    <name>action:createReview</name>
  </actionData>
  <actionData>
    <name>action:recoverReview</name>
  </actionData>
  <actionData>
    <name>action:commentOnReview</name>
  </actionData>
  <actionData>
    <name>action:reopenReview</name>
  </actionData>
  <actionData>
    <name>action:abandonReview</name>
  </actionData>
  <actionData>
    <name>action:submitReview</name>
  </actionData>
</actions>
```

Example JSON Return Data:
Get Allowed Review Transitions

URL:
GET /reviews-v1/<id>/transitions

Description:
Get a list of the actions which the current user can perform on this review, given its current state and the user's permissions.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<transitions>
  <transitionData>
    <name>action:summarizeReview</name>
  </transitionData>
  <transitionData>
    <name>action:abandonReview</name>
  </transitionData>
</transitions>
```

Example JSON Return Data:

```json
{"transitions": [
  "transitionData": [
    "name": "action:approveReview",
    "name": "action:abandonReview"]
]}
```

Get Review

URL:
GET /reviews-v1/<id>

Description:
Get a single review.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```
<reviewData>
 ... 
</reviewData>
```

**Example JSON Return Data:**

```
{"reviewData": { 
   "allowReviewersToJoin":false,
   ... 
 }}
```

---

**Get Review Details**

**URL:**

```
GET /reviews-v1/filter/<filter>/details
```

**Description:**
Get details of all the reviews which match the given filter. See above for filter names.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```
<detailedReviews>
 <detailedReviewData>
  ... 
 </detailedReviewData>
  ... 
</detailedReviews>
```

**Example JSON Return Data:**

```
{"detailedReviews": { 
   "detailedReviewData": [ 
      {"allowReviewersToJoin":true,
       ... 
     ], ...
   }
  }
```

---

**Get Review Details by Path**

**URL:**

```
GET /reviews-v1/search/<repository>/details?path=<path>
```
**Description:**
Return a list of Review details which include a particular file. The path parameter must be the full path name of a file in repository, with no leading slash.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**
```
<detailedReviews>
  <detailedReviewData>
    ...
  </detailedReviewData>
  ...
</detailedReviews>
```

**Example JSON Return Data:**
```
{"detailedReviews": { 
  "detailedReviewData": [ 
    {"allowReviewersToJoin":true,
      ...
    }, ...
  ]
}
```

---

**Get Review Details through Custom Filter Criteria**

**URL:**
```
GET
/reviews-v1/filter/details?title=..&author=..&moderator=..&creator=..&reviewer=..&orRoles=true|false&complete=true|false&allReviewersComplete=true|false&project=..
```

**Description:**
Get details of all the reviews which match the specified filter criteria. Criteria are supplied as normal query parameters in the URL.
Filter criteria are:
- **title** - Reviews whose title contain this substring.
- **author** - Reviews authored by this user.
- **moderator** - Reviews moderated by this user.
- **creator** - Reviews created by this user.
- **reviewer** - Reviews reviewed by this user.
- **orRoles** - Whether to the value for author, creator, moderator and reviewer should be combined using OR (orRoles=true) or AND (orRoles=false).
- **complete** - Reviews that the specified reviewer has completed.
- **allReviewersComplete** - Reviews that all reviewers have completed.
- **project** - Reviews for the specified project.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**
```
<reviews>
  <reviewData>
    ...
  </reviewData>
  ...
</reviews>
```

**Example JSON Return Data:**
Get Reviews by Filter

URL:

GET /reviews-v1/filter/<filter>

Description:
Get all the reviews which match the given filter, for the current user.
Filter names are:

- allReviews - All reviews for everyone.
- allOpenReviews - Open reviews for everyone.
- allClosedReviews - Closed reviews for everyone.
- draftReviews - Draft reviews for everyone.
- toReview - Reviews on which the current user is an uncompleted reviewer.
- requireMyApproval - Reviews waiting to be approved by the current user.
- toSummarize - Completed reviews which are ready for the current user to summarize.
- outForReview - Reviews with uncompleted reviewers, on which the current reviewer is the moderator.
- drafts - Draft reviews created by the current user.
- open - Open reviews created by the current user.
- closed - Closed reviews created by the current user.
- trash - Abandoned reviews created by the current user.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<reviews>
  <reviewData>
    ...
  </reviewData>
  ...
</reviews>
```

Example JSON Return Data:

```json
{"reviews": {
  "reviewData": [
    {"allowReviewersToJoin":true,
     ...}
  ],...}
}
```

Get Reviews by Path

URL:
GET /reviews-v1/search/<repository>?path=<path>

**Description:**
Return a list of Reviews which include a particular file. The `path` parameter must be the full path name of a file in `repository`, with no leading slash.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**

```
<reviews>
  <reviewData>
    ...
  </reviewData>
  ...
</reviews>
```

**Example JSON Return Data:**

```
{"reviews": {
  "reviewData": [
    {"allowReviewersToJoin":true,
      ...
    }, ...
  ]
}
```

---

**Get Reviews through Custom Filter Criteria**

**URL:**

```
GET /reviews-v1/filter?title=..&author=..&moderator=..&creator=..&reviewer=..&orRoles=true|false&complete=true|false&allReviewersComplete=true|false&project=..
```

**Description:**
Get all the reviews which match the specified filter criteria. Criteria are supplied as normal query parameters in the URL.

Filter criteria are:

- **title** - Reviews whose title contain this substring.
- **author** - Reviews authored by this user.
- **moderator** - Reviews moderated by this user.
- **creator** - Reviews created by this user.
- **reviewer** - Reviews reviewed by this user.
- **orRoles** - Whether to the value for `author`, `creator`, `moderator`, and `reviewer` should be combined using OR (`orRoles=true`) or AND (`orRoles=false`).
- **complete** - Reviews that the specified `reviewer` has completed.
- **allReviewersComplete** - Reviews that all reviewers have completed.
- **project** - Reviews for the specified project.

**Status Code:**
200 (OK) on success.

**Example XML Return Data:**
Example JSON Return Data:

```json
{"reviews": {
  "reviewData": [
    {
      "allowReviewersToJoin":true,
      ...
    }, ...
  ]
}}
```

Get Single Review Details

URL:

GET /reviews-v1/<id>/details

Description:
Get details of a single review.

Status Code:
200 (OK) on success.

Example XML Return Data:

```xml
<detailedReviewData>
  ...
</detailedReviewData>
```

Example JSON Return Data:

```json
{"detailedReviewData": {
  "allowReviewersToJoin":false,
  ...
}}
```

Get Version Info

URL:

GET /reviews-v1/versionInfo

Description:
Returns the release number and build date of Crucible.

Status Code:
200 (OK) on success.
Example XML Return Data:

```xml
<versionInfo>
  <buildDate>2008-10-28</buildDate>
  <releaseNumber>1.6.3</releaseNumber>
</versionInfo>
```

Example JSON Return Data:

```json
{"versionInfo":{"buildDate":"2008-10-28","releaseNumber":"1.6.3"}}
```

Review Service - Workflow

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

On this page:

- **Workflow**
  - Close a Review
  - Complete a Review
  - Move a Review to a New State
  - Uncomplete a Review

**Workflow**

**Close a Review**

POST /reviews-v1/<review id>/close

**Description:**
Close the review.

**Status Code:**
200 (OK) on success.

**Complete a Review**

POST /reviews-v1/<review id>/complete

**Description:**
Indicate that the current user has completed the review.

**Status Code:**
200 (OK) on success.

**Move a Review to a New State**

POST /reviews-v1/<review id>/transition?action=<action>

**Description:**
Change the state of the review.

**Status Code:**
200 (OK) on success.
Valid actions are:

- action: abandonReview
- action: deleteReview
- action: submitReview
- action: approveReview
- action: rejectReview
- action: summarizeReview
- action: closeReview
- action: reopenReview
- action: recoverReview
- action: completeReview
- action: uncompleteReview

**Uncomplete a Review**

```
POST /reviews-v1/<review id>/uncomplete
```

**Description:**
Indicate that the current user has **not** completed the review.

**Status Code:**
200 (OK) on success.

---

### Crucible Plugin Types

Crucible plugins come in a variety of flavours, read on to see how the plugin technology interacts with the core of Crucible and what rules can be bent, or possibly *broken* in this world.

<table>
<thead>
<tr>
<th>Source Code Management (SCM) Plugins</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Crucible SCM Plugins</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Servlets</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Servlet Modules</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event Listeners</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Event Listener Plugins</td>
</tr>
</tbody>
</table>

---

### Crucible Web Items

Web UI plugin modules allow you to add links, interactive elements and page segments to the Crucible user interface. By adding a link to a servlet plugin you can add your own pages to the UI. Your pages will need to ask Crucible to provide standard headers and footers by specifying a Decorator. There are also FishEye Web Items.

On this page:

- Web Items Listing and Reference
- Web Item Conditions
  - Condition Parameters
  - Example of a Web Item Condition in Use
**Visual Locations of Crucible Web Items**

For an example of the code syntax, see FishEye Web Items.

### Web Items Listing and Reference

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Helpers available</th>
</tr>
</thead>
<tbody>
<tr>
<td>system.admin</td>
<td>Links on the admin menu. Sections: repositories, global, system.</td>
<td>application, user</td>
</tr>
<tr>
<td>system.crucible.dashboard</td>
<td>After 'My Dashboard' in the dropdown project/dashboard selector menu</td>
<td>application, user</td>
</tr>
<tr>
<td>system.crucible.review</td>
<td>Actions which can be performed on a review. These appear both as buttons on the review page and as links on an expanded review in a review list.</td>
<td>application, user, project, review</td>
</tr>
<tr>
<td>system.crucible.review.comment</td>
<td>Actions which can be performed on a review comment. These appear as buttons in the comment header bar, left of the reply, edit and delete buttons.</td>
<td>application, user, project, review, reviewItem, repository, comment</td>
</tr>
<tr>
<td>system.crucible.review.fileitem</td>
<td>Actions which can be performed on a revision in a review. Displayed next to the 'Remove', 'Change Diff' buttons.</td>
<td>application, user, project, review, reviewItem, repository</td>
</tr>
<tr>
<td>system.header.application</td>
<td>Links like 'Crucible' and 'Fisheye' in the header. Sections: fisheye, crucible (sections are shown when a particular application is selected)</td>
<td>application, user</td>
</tr>
<tr>
<td>system.header.item</td>
<td>Links in the header, separated by a pipe.</td>
<td>application, user</td>
</tr>
<tr>
<td>system.main</td>
<td>Links on the Crucible or FishEye main page. These appear at the bottom of the Crucible or FishEye main page. Sections: fisheye, crucible</td>
<td>application, user</td>
</tr>
<tr>
<td>system.userprofile.tab</td>
<td>The user profile tabs.</td>
<td>application, user</td>
</tr>
</tbody>
</table>

### Web Item Conditions

Conditions control whether a given web item will be displayed.

- **com.atlassian.fisheye.plugin.web.conditions.HasCrucible**
  This condition measures whether the product runs with a Crucible license. This is useful to prevent a Crucible plugin from rendering in an instance that only has a FishEye license.

- **com.atlassian.fisheye.plugin.web.conditions.HasFishEye**
  This condition measures whether the product runs with a FishEye license.

- **com.atlassian.fisheye.plugin.web.conditions.HasProjectPermission**
  This condition measures whether the user has project permission. Takes parameters.

- **com.atlassian.fisheye.plugin.web.conditions.HasReviewPermission**
  This condition measures whether the user has review permission (i.e. is able to take part in the review). Takes parameters.

- **com.atlassian.fisheye.plugin.web.conditions.IsFile**
  This condition passes if there is a context repository and path, and that path references a repository file.

- **com.atlassian.fisheye.plugin.web.conditions.IsReviewInState**
  This condition measures whether the review is in a given state. Takes parameters.

- **com.atlassian.fisheye.plugin.web.conditions.IsRootOrDirectory**
  This condition passes if there is either: a context repository and no repository context path; or a repository path is present, and that path references a directory.

- **com.atlassian.fisheye.plugin.web.conditions.IsSystemAdministrator**
  This condition measures whether the user has system administrator permissions.

- **com.atlassian.fisheye.plugin.web.conditions.UserCanAccessCrucible**
  This condition measures whether the user can access Crucible.

- **com.atlassian.fisheye.plugin.web.conditions.UserLoggedInCondition**
  This condition measures whether the user is logged in.

### Condition Parameters

The following conditions take parameters:
The usage and conditions that these parameters apply to are tabled below.

<table>
<thead>
<tr>
<th>Parameter Value</th>
<th>Parameter Name</th>
<th>Description</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>action:abandonReview</td>
<td>actionName</td>
<td>Causes the current review to be abandoned.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:approveReview</td>
<td>actionName</td>
<td>Causes the current review to be approved.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:closeReview</td>
<td>actionName</td>
<td>Causes the current review to be closed.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:recoverReview</td>
<td>actionName</td>
<td>Causes the current review to be recovered.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:reopenReview</td>
<td>actionName</td>
<td>Causes the current review to be re-opened.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:rejectReview</td>
<td>actionName</td>
<td>Causes the current review to be rejected.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:submitReview</td>
<td>actionName</td>
<td>Causes the current review to be submitted.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>action:summarizeReview</td>
<td>actionName</td>
<td>Causes the current review to be summarised.</td>
<td>HasProjectPermission, HasReviewPermission</td>
</tr>
<tr>
<td>Approval</td>
<td>stateName</td>
<td>Measures whether the current review is in the approval state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Closed</td>
<td>stateName</td>
<td>Measures whether the current review is in the closed state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Dead</td>
<td>stateName</td>
<td>Measures whether the current review is in the dead state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Draft</td>
<td>stateName</td>
<td>Measures whether the current review is in the draft state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Review</td>
<td>stateName</td>
<td>Measures whether the current review is in the review state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Rejected</td>
<td>stateName</td>
<td>Measures whether the current review is in the rejected state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Summarize</td>
<td>stateName</td>
<td>Measures whether the current review is in the summarize state.</td>
<td>IsReviewInState</td>
</tr>
<tr>
<td>Unknown</td>
<td>stateName</td>
<td>Measures whether the current review is in the unknown state.</td>
<td>IsReviewInState</td>
</tr>
</tbody>
</table>

Applying these values will cause the action to be enacted on the currently logged-in user.

Example of Condition Parameters in Use

```xml
<condition class="com.atlassian.fisheye.plugin.web.conditions.HasReviewPermission">
    <param name="actionName" value="action:approveReview"/>
</condition>
```

Example of a Web Item Condition in Use
Visual Locations of Crucible Web Items

system.admin

This item relates to links in the left navigation bar, in the Crucible admin menu.

Screenshot: Crucible’s system.admin/repositories Web Item

```
Admin Menu
Repository Settings
  - Repository List (new)
  - Repository Defaults
  - System Admin/Repositories
```

Screenshot: Crucible’s system.admin/global Web Item

```
Global Settings
  - Server Settings
  - Security
  - Users
  - Groups
  - Administrators
  - ViewCVS URL Mappings
  - Change Admin Password
  - Customize Crucible Defect Classifications
  - Projects
  - Permission Schemes
  - Trusted Applications
  - Customize Front Page
  - System Admin/GLOBAL
```

Screenshot: Crucible’s system.admin/system Web Item

```
System
  - Sys-Info/Support
  - Content
  - Backup
  - Plugins
  - Shutdown
  - System Admin/System
```
system.crucible.dashboard

This item relates to dashboard links in the Crucible dashboard/project drop-down menu.

Screenshot: Crucible's system.crucible.dashboard Web Item

---

system.crucible.review

This item relates to actions that can be performed on a review, appearing in various places inside the Crucible UI.

Screenshot: Crucible's system.crucible.review Web Item

---

system.crucible.review.comment

This item relates to actions that can be performed on a review, appearing in various places inside the Crucible UI.

Screenshot: Crucible's system.crucible.review.comment Web Item

---

system.crucible.review.fileitem

This item relates to actions which can be performed on a revision in a review, in the Crucible UI.

Screenshot: Crucible's system.crucible.review.fileitem Web Item
system.header.application

This item relates to product name links in the Crucible header.

Note that a system.header.application item must go into a section of either Crucible or FishEye. In this case, we have put it into the Crucible section.

Screenshot: Crucible's system.header.application Web Item

system.header.item

This item relates to links in the Crucible header, at the top right of the Crucible screen.

Screenshot: Crucible's system.header.item Web Item

system.main/crucible

This item relates to links at the bottom of the Crucible main page.

Screenshot: Crucible's system.main/crucible Web Item

system.userprofile.tab

This item relates to user profile tabs in the Crucible UI.

Screenshot: Crucible's system.userprofile.tab Web Item
Looking for the FishEye web items? Click here.

**Live Code Examples for Crucible Development**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

On this page is a list of real-world plugin examples that showcase the various sides of Crucible development. The following items are an excellent resource for the Atlassian developer community. Feel free to investigate these examples, hack them to pieces, or use them as inspiration to really innovate.

<table>
<thead>
<tr>
<th>SCM Plugin Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Example Crucible SCM Plugin for JSR-170 (Apache JackRabbit)</td>
</tr>
<tr>
<td>• Crucible ClearCase plugin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Servlet Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Basic Servlet Example</td>
</tr>
<tr>
<td>• Crucible Reporting plugin</td>
</tr>
</tbody>
</table>

**Bundled Plugins from Crucible**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

There are a number of Crucible features that ship as plugins with the product. See below for listings and more information.

- Confluence SCM Plugin
- File System SCM Plugin
- Perforce SCM Plugin
- Subversion SCM Plugin

**Confluence SCM Plugin**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Crucible Confluence SCM plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1.0-SNAPSHOT</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.6.2 and later</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CRUCIBLE/">http://confluence.atlassian.com/display/CRUCIBLE/</a></td>
</tr>
<tr>
<td>Price</td>
<td>Bundled with Crucible</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>IssueTracking</td>
<td>Jira</td>
</tr>
<tr>
<td></td>
<td>FishEye: <a href="http://svn.atlassian.com/fisheye/browse/public/contrib/crucible/crucible-confluence-scm-plugin">http://svn.atlassian.com/fisheye/browse/public/contrib/crucible/crucible-confluence-scm-plugin</a></td>
</tr>
</tbody>
</table>
Description/Features

This page provides links to resources for Crucible plugin developers. Access the source code and distributed packages from the links in the table above.

In conjunction with the Confluence Crucible Plugin this allows the creation of review of changes to Confluence pages.

Usage

To set up this plugin, see the documentation.

Installation

This plugin ships with the Crucible distribution.

Configuration

Follow the Crucible configuration documentation.

Related Links

- How to build a Crucible Plugin

File System SCM Plugin

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Crucible File System SCM plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1.2</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.6.0+</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CRUCIBLE/">http://confluence.atlassian.com/display/CRUCIBLE/</a></td>
</tr>
<tr>
<td>Price</td>
<td>Bundled with Crucible</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>IssueTracking</td>
<td><a href="http://developer.atlassian.com/jira/browse/CRU">http://developer.atlassian.com/jira/browse/CRU</a></td>
</tr>
</tbody>
</table>

Description/Features

This page provides links to resources for Crucible plugin developers. Access the source code from the links in the table above.

This plugin allows Crucible to review files directly on the server file system.

Usage

To use this plugin, see the documentation.

Installation

This plugin ships with the Crucible distribution.

Related Links
How to build a Crucible Plugin

**Perforce SCM Plugin**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Crucible Perforce SCM plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1.2</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.6.4+</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CRUCIBLE/">http://confluence.atlassian.com/display/CRUCIBLE/</a></td>
</tr>
<tr>
<td>Price</td>
<td>Bundled with Crucible</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>IssueTracking</td>
<td><a href="http://developer.atlassian.com/jira/browse/CRU">http://developer.atlassian.com/jira/browse/CRU</a></td>
</tr>
</tbody>
</table>

**Description/Features**

This page provides links to resources for Crucible plugin developers. Access the source code from the links in the table above.

This plugin allows Crucible to access Perforce repositories without requiring FishEye.

**Usage**

To use this plugin, see the documentation.

**Installation**

This plugin ships with the Crucible distribution.

**Related Links**

- How to build a Crucible Plugin

**Subversion SCM Plugin**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Crucible Subversion SCM plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1.2</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.6.0+</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CRUCIBLE/">http://confluence.atlassian.com/display/CRUCIBLE/</a></td>
</tr>
<tr>
<td>Price</td>
<td>Bundled with Crucible</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
**Description/Features**

This page provides links to resources for Crucible plugin developers. Access the source code from the links in the table above.

This plugin allows Crucible to access Subversion repositories without requiring FishEye.

**Usage**

To use this plugin, see the [documentation](http://confluence.atlassian.com/display/CRUCIBLE/Crucible+Git+plugin).

**Installation**

This plugin ships with the Crucible distribution.

**Related Links**

- [How to build a Crucible Plugin](http://confluence.atlassian.com/display/CRUCIBLE/Crucible+Git+plugin)

**SCM Plugin Examples**

This page contains examples of SCM plugins that can be added to Crucible.

*Crucible SCM plugin listing:*

- Crucible Git Plugin
- Example Crucible SCM Plugin for JSR-170 (Apache JackRabbit)

**Crucible Git Plugin**

This page contains elements. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Crucible Git plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.6.6</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CRUCIBLE/Crucible+Git+plugin">http://confluence.atlassian.com/display/CRUCIBLE/Crucible+Git+plugin</a></td>
</tr>
<tr>
<td>Price</td>
<td>Free</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>TBA</td>
</tr>
<tr>
<td>IssueTracking</td>
<td>JIRA</td>
</tr>
<tr>
<td>Subversion URL</td>
<td>Subversion FishEye</td>
</tr>
<tr>
<td>Download JAR</td>
<td>Atlassian Maven Repository (supports Crucible 1.6.6)</td>
</tr>
<tr>
<td>Download Source</td>
<td>TBA</td>
</tr>
</tbody>
</table>

**Description/Features**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.
A plugin for Crucible that facilitates the usage of Git source code repositories.

**Usage**

The Git plugin is an early-access implementation of a Crucible SCM plugin for Git. It allows users to perform code reviews on a local Git repository (local to the Crucible server). The plugin does not 'pull' updates from a remote master repository. Synchronising with the master repository needs to be executed manually (via the command line), for the changes to appear in the plugin.

**Installation**

Firstly, download the plugin .JAR file to your local computer.

The plugin is installed by placing the .JAR file in the FISHEYE_INST/var/plugins/user directory of your Crucible install. Once installed, you need to enable the plugin in the Crucible Admin interface. Detailed instructions on the plugin installation steps can be found at the Managing Plugins page.

The plugin requires the Git command to be available in the system path when starting Crucible.

**Configuring the plugin**

Once the plugin has been installed, under the 'Administration' - 'Repository List' option, there should be a 'Plugin Repository List: Git' entry. Select 'Configure Plugin', then 'Add a repository'. The fields required are:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name for the repository eg. Project</td>
</tr>
<tr>
<td>Repository Path</td>
<td>The location of the local Git repository clone</td>
</tr>
</tbody>
</table>

Once configured, the Git repository can be selected as the review source when creating a new review, whereupon reviews can be created either using changesets or by selecting files in the repository view.

**Feedback**

If you have any feedback on this plugin and its operation, we would appreciate users posting feedback in the Crucible Forums.

**Version History**

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>10 Feb 2009</td>
<td>Early Access as part of Crucible 1.6.6</td>
</tr>
</tbody>
</table>

**Example Crucible SCM Plugin for JSR-170 (Apache JackRabbit)**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Example Crucible SCM Plugin for JSR-170 (Apache JackRabbit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1.0-SNAPSHOT</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.6.3</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Erik van Zijst, Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CRUCIBLE/">http://confluence.atlassian.com/display/CRUCIBLE/</a></td>
</tr>
<tr>
<td>Price</td>
<td>Free</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>Crucible SCM API JavaDoc</td>
</tr>
<tr>
<td>Browse Source</td>
<td>FishEye</td>
</tr>
<tr>
<td>Download Source</td>
<td>Subversion</td>
</tr>
</tbody>
</table>
Crucible 2.4 Documentation

Description/Features
This plugin contains an implementation of Crucible SCM for Java Content Repositories (JSR-170), using the Apache JackRabbit implementation.
Note that although this code can be used without modification, it is NOT supported and intended as example code only. It should not be used in a
production environment, because of several known issues discussed below. Having said that, feel free to improve on this code

Usage
Check out the source and build the plugin jar file using mvn package. Note that this plugin depends on com.sun.jdmk:jmxtools:1.2.1 and
com.sun.jmx:jmxrmi:1.2.1. These libraries are provided by Sun, but may not be present in your maven repository, as Sun requires each
user to agree to its license terms. If you get the following build error:
[INFO] -----------------------------------------------------------------------[ERROR] BUILD ERROR
Missing:
---------1) com.sun.jdmk:jmxtools:jar:1.2.1
Try downloading the file manually from:
Then, install it using the command:
mvn install:install-file -DgroupId=com.sun.jdmk -DartifactId=jmxtools -Dversion=1.2.1
-Dpackaging=jar -Dfile=/path/to/file
Alternatively, if you host your own repository you can deploy the file there:
mvn deploy:deploy-file -DgroupId=com.sun.jdmk -DartifactId=jmxtools -Dversion=1.2.1
-Dpackaging=jar -Dfile=/path/to/file -Durl=[url] -DrepositoryId=[id]
Path to dependency:
1) com.atlassian.crucible.example.scm:jackrabbit-scm-plugin:atlassian-plugin:1.0-SNAPSHOT
2) com.sun.jdmk:jmxtools:jar:1.2.1
2) com.sun.jmx:jmxri:jar:1.2.1
Try downloading the file manually from the project website.
Then, install it using the command:
mvn install:install-file -DgroupId=com.sun.jmx -DartifactId=jmxri -Dversion=1.2.1
-Dpackaging=jar -Dfile=/path/to/file
Alternatively, if you host your own repository you can deploy the file there:
mvn deploy:deploy-file -DgroupId=com.sun.jmx -DartifactId=jmxri -Dversion=1.2.1 -Dpackaging=jar
-Dfile=/path/to/file -Durl=[url] -DrepositoryId=[id]
Path to dependency:
1) com.atlassian.crucible.example.scm:jackrabbit-scm-plugin:atlassian-plugin:1.0-SNAPSHOT
2) log4j:log4j:jar:1.2.15
3) com.sun.jmx:jmxri:jar:1.2.1
---------2 required artifacts are missing.

Then download the two libraries from Sun, extract the jar files from the zip files and follow the instructions above to manually install the artifact in
your local repository. You should then be able to complete the build.
Installation

Copy the jackrabbit-scm-plugin-1.0-SNAPSHOT.jar file from the 'target/' directory to the var/plugins/user directory in your Crucible
installation. Then login to the administration section, go to Plugins and click the link "Check for new plugins in...". This should detect your plugin:


Configuring the plugin

Next, click "Configure" and add a repository. The current version of this plugin can only read JackRabbit repositories directly from the local file system (as opposed to connecting to a remote JackRabbit server), so when configuring a repository, specify the location of the repository's XML file and the repository's home directory (the directory containing workspaces/ repository/ version/).

The plugin comes with a ready to use, pre-populated JackRabbit repository in testrepo.zip (the unit tests run against this repository). When trying things out, unzip this file somewhere on the file system and point the plugin to it.

Known Limitations

As this plugin is intended as example material for those interested in building Crucible SCM Plugins, readability of the source code is more important than features, flexibility or performance and as result there are quite a number of known limitations:

- Only file based JackRabbit 1.x repositories on the local file system are currently supported. This makes this plugin impossible to use with "active" repositories, because the JackRabbit acquires an exclusive lock on the repository.
- The plugin only recognizes nodes of JCR types "nt:file" and "nt:folder". All other nodes are ignored.
- All file and folder nodes must have the mixin type "mix:versionable".
- Every file or folder node must have at least one checked-in version (hence: at least one version of each resource must have been committed).
- JSR-170 does not support ChangeSets where a changset represents a collection of related changes to multiple entities at once. As a consequence, the plugin represents every individual change as a ChangeSet containing one file. The UUID of the version node is used as the changset ID.
- This plugin does not detect files that are deleted.
- Due to file locking issues in JackRabbit, the plugin uses a single JCR Session instance per repository. Access is synchronized.
- The implementation for listing a set of changsets traverses the entire repository and does not cache results, which kills scalability.

Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0-SNAPSHOT</td>
<td>10-November-2008</td>
<td>Initial release</td>
</tr>
</tbody>
</table>

Screenshots
The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

This page lists the Servlet code examples for Crucible plugin developers.

- Basic Servlet Example
  - Crucible Reporting plugin

### Basic Servlet Example

- The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Example FishEye Servlet Plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>0.1</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.7</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Anna Buttfield, Atlassian</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/FISHEYE/">http://confluence.atlassian.com/display/FISHEYE/</a></td>
</tr>
<tr>
<td>Price</td>
<td>Free</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>Data Package Summary</td>
</tr>
<tr>
<td></td>
<td>Services Package Summary</td>
</tr>
<tr>
<td>IssueTracking</td>
<td>N/A</td>
</tr>
<tr>
<td>Subversion URL</td>
<td>FishEye svn</td>
</tr>
<tr>
<td>Download JAR</td>
<td>Attached to this page: compiled jar sources jar</td>
</tr>
<tr>
<td>Download Source</td>
<td>svn</td>
</tr>
</tbody>
</table>

### Description/Features

Basic plugin showing the use of the FishEye API in a servlet. This can be used as the basis for more advanced FishEye plugins.
**Usage**

If compiling from source, follow the instructions listed in the 'readme' file.

Warning: Requires a version 1.7 (or later) FishEye development build, will not work with released versions.

**Installation**

1. Copy the plugin .jar file from the 'target' directory to the var/plugins/user directory in your FishEye installation.
2. Run FishEye and point your browser at this location:

```
FISHEYE_HOME/plugins/servlet/[servlet url]
```

to view the servlet (where the servlet url is set in 'url-pattern' in `atlassian-plugin.xml` and is set to `example-servlet` by default).

**Configuring the plugin**

No configuration is required, just start FishEye and point a browser at this URL:

```
FISHEYE_HOME/plugins/servlet/example_servlet
```

**Version History**

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>7-November-2008</td>
<td>Initial release</td>
</tr>
</tbody>
</table>

**Screenshots**

**Screenshots**

![Screenshots](image)

**Crucible Reporting plugin**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

<table>
<thead>
<tr>
<th>Name</th>
<th>Crucible Reporting plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>2.0.0</td>
</tr>
<tr>
<td>Product Versions</td>
<td>1.5.x, 1.6.x</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Ross Rowe</td>
</tr>
<tr>
<td>Homepage</td>
<td><a href="http://confluence.atlassian.com/display/CODEGEIST/Crucible+Reporting+plugin">http://confluence.atlassian.com/display/CODEGEIST/Crucible+Reporting+plugin</a></td>
</tr>
<tr>
<td>Price</td>
<td>Free</td>
</tr>
<tr>
<td>License</td>
<td>BSD</td>
</tr>
<tr>
<td>JavaDocs</td>
<td>crucible-export-plugin-javadoc.zip</td>
</tr>
<tr>
<td>Issue Tracking</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Description/Features

A plugin for Crucible that facilitates the generation of a consolidated report for a specific review. This is especially useful if you are required to keep hard copies of your code review (like in the case of an audit 😊).

Usage

⚠️ This plugin requires Crucible 1.5 or higher

Crucible 1.5 installation

The plugin can be installed by copying the `crucible-export-1.0.0.jar` file into the CRUCIBLE_HOME/var/plugins directory. You will also need to copy the iText jar file (available from [http://www.lowagie.com/iText/download.html](http://www.lowagie.com/iText/download.html)) into the CRUCIBLE_HOME/lib directory.

Crucible 1.6 and higher installation

The plugin can be installed by copying the `crucible-export-1.6.2.jar` file into the CRUCIBLE_HOME/var/plugins/user directory.

Running the plugin

As Crucible does not currently have a mechanism to include user interface components via it's plugin api, the export mechanism can be opened by visiting [http://YourCrucibleHost/plugins/servlet/export](http://YourCrucibleHost/plugins/servlet/export). From this page, the user must enter their username, password and the review id they wish to export.

Once the details are entered and the 'Run' button is clicked, a PDF including the Crucible Review details is generated. This report includes the summary information of the review, as well as any general and specific file comments.

Version History

Crucible 1.6 and higher support
Introduction

Crucible uses the standard Atlassian Plugins framework, so many of the tasks involved in developing a plugin for Crucible are the same as for other Atlassian products.

The differences are:

- The set of plugin types available.
- And, of course, the API available for plugins to interact with the Crucible application.

Building a Crucible Plugin

The simplest way to build a Crucible plugin is via Maven.

Atlassian provides an Archetype for FishEye/Crucible plugins.
You can create a maven 2 project containing a sample Servlet Plugin Module with the following command:

```
mvn org.apache.maven.plugins:maven-archetype-plugin:1.0-alpha-7:create \
    -DarchetypeGroupId=com.atlassian.maven.archetypes \
    -DarchetypeArtifactId=crucible-plugin-archetype \
    -DarchetypeVersion=1-SNAPSHOT \
    -DremoteRepositories=https://maven.atlassian.com/repository/public/ \
    -DgroupId=com.foo -DartifactId=foo-crucible-plugin
```

This will create your project in a subdirectory of your current directory named `foo-crucible-plugin`. Change into that directory (cd `foo-crucible-plugin`). You can create the plugin jar with the command `mvn package`, and install it in a running Fisheye or Crucible instance by copying `target/foo-crucible-plugin-1.0-SNAPSHOT.jar` to the `var/plugins/user` directory of your Fisheye/Crucible instance.

**Crucible Plugin Module Types**

**Servlet Modules**

Create a servlet which is deployed to the same web application context as Fisheye/Crucible. See Servlet Plugin Modules for more details.

**Crucible SCM Plugins**

An SCM plugin module lets Crucible create reviews based on files stored in another source code management system. See Crucible SCM Plugins for details.

**Event Listener Plugins**

An event listener plugin module will be called when certain events occur inside Crucible. See Crucible Event Listener Plugins for details.

**The Crucible API**

Your plugin will need to use The Crucible API to retrieve data from Crucible and to perform operations on it, such as changing the state of reviews.

**Debugging your plugin**

You can start Crucible in debug mode with the environment variable setting:

```
export FISHEYE_OPTS="-Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=5005"
```

This allows you to connect your IDE to the debugger listening on port 5005.

**Crucible Event Listener Plugins**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

An event listener plugin module is an object which is notified when certain internal Crucible or FishEye events occur.

To include an event listener module add a `listener` element to your `atlassian-plugins.xml` file:

```
<listener key="example-listener" 
    class="com.atlassian.crucible.example.plugin.spring.ExampleListener"/>
```

and create a class which implements `com.atlassian.event.EventListener`. See the FishEye event and Crucible event javadoc for specific event types. See the [javadoc for EventListener](#) to understand the general details regarding events.

For example, if we want to listen for all events, and print a message to standard output we would write:
public class ExampleListener implements EventListener {
    public void handleEvent(Event event) {
        System.out.println("Got event: " + event);
    }

    public Class[] getHandledEventClasses() {
        return new Class[0];
    }
}

Event listeners may implement `StateAware` if they need to be notified when the module is enabled or disabled.

A plugin containing an event listener module needs to declare a dependency on `atlassian-events` in its `pom.xml`:

```xml
<dependency>
    <groupId>com.atlassian.event</groupId>
    <artifactId>atlassian-event</artifactId>
    <version>0.5</version>
    <scope>provided</scope>
</dependency>
```

Note that this is a `provided` dependency – the plugin does not need to include the `atlassian-events` classes.

**Crucible SCM Plugins**

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

On this page:

- Crucible SCM Plugins
- Creating a Project
- Crucible SCM Plugin API
- Servlet Based Administration Pane
- Packaging, Deploying and Running

**Crucible SCM Plugins**

Crucible SCM modules are plugins that make version control systems accessible to Crucible. An SCM plugin can be used to give Crucible the ability to work with a custom version control system that is not supported out of the box. SCM plugins are independent from FishEye’s version control integrations and allow Crucible to run standalone. Crucible ships with a number of built-in SCM plugins, including Subversion and Perforce.

In this section we will implement a new Crucible SCM Plugin and explore Crucible’s public SCM API. The example builds a module that exposes the underlying file system as the “repository”, so that users can perform reviews of files on the server file system.

**Creating a Project**

To start, we use the Crucible Plugin archetype to create a new empty Maven2 project:

```bash
mvn org.apache.maven.plugins:maven-archetype-plugin:1.0-alpha-7:create \
    -DarchetypeGroupId=com.atlassian.maven.archetypes \
    -DarchetypeArtifactId=crucible-plugin-archetype \
    -DarchetypeVersion=1-SNAPSHOT \
    -DremoteRepositories=https://maven.atlassian.com/repository/public/ \
    -DgroupId=com.atlassian.crucible.example.scm \
    -DartifactId=example-scm-plugin
```

This creates a new project that has a dependency on `atlassian-fisheye-api`. This library contains the basic API components required by plugins. However, as we are building an SCM plugin that can be configured through a servlet, we need to add a dependency on `atlassian-crucible-scmutils` as well as `atlassian-plugins-core` by editing the generated `pom.xml`:
 IDEA Users
If you are using IntelliJ for development, be sure to run `mvn idea:idea` to generate the project files. Opening the pom file directly is known to miss the parent dependencies.

Crucible SCM Plugin API

Crucible’s public API can be browsed online and contains the functionality needed to develop a custom SCM plugin in the package com.atlassian.crucible.scm. It consists of a set of interfaces, some of which are optional, for browsing a repository, accessing its directories, retrieving file contents and exploring changes between revisions.

At the very least, your SCM plugin should implement the com.atlassian.crucible.scm.SCMModule interface that defines the new plugin. The module is then used to create one or more repository instances:
When your module is instantiated, Crucible passes a ModuleDescriptor instance to it containing information about the plugin. The getRepositories() method returns the repositories offered by this plugin. Currently we’re returning an empty collection.

To be able to use the Crucible administration console to configure our plugin and specify the locations of the repositories we want to use, we will also implement the Configurable interface that allows for the injection of a custom configuration bean (by implementing SimpleConfiguration) whose properties can be manipulated through the administration interface for which we will write a small servlet. In our custom configuration bean we’ll add a property for the base path or root directory of the file system based repositories we want to offer.

The plugin configuration is written to disk and fed to our SCMModule when Crucible starts up. Our plugin is responsible for generating and parsing that data, so we’re free to choose the format. The ModuleConfigurationStore provides persistent storage and will automatically be injected into our plugin if we create a constructor that takes it as an argument. For the serialization, let’s use simple XML serialization through XStream (using XStream is convenient as it is one of the dependencies for atlassian-crucible-scmutils):
package com.atlassian.scm;
import com.atlassian.fisheye.plugins.scm.utils.SimpleConfiguration;

public class ExampleConfiguration implements SimpleConfiguration {
    private String name;
    private String basePath;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getBasePath() {
        return basePath;
    }

    public void setBasePath(String basePath) {
        this.basePath = basePath;
    }
}

Now we make the required changes to our SCMModule to read and write the configuration:

public class ExampleSCMModule implements SCMModule, Configurable<List<ExampleConfiguration>> {
    private ModuleDescriptor moduleDescriptor;
    private ModuleConfigurationStore store;

    public ExampleSCMModule(ModuleConfigurationStore store) {
        this.store = store;
    }

    public List<ExampleConfiguration> getConfiguration() {
        byte[] configData = store.getConfiguration(moduleDescriptor);
        if (configData != null) {
            try {
                return (List<ExampleConfiguration>)getXStream().fromXML(new String(configData, "UTF8"));
            } catch (Exception e) {
                throw new RuntimeException("Error reading configuration:" + configData, e);
            }
        }
        return new ArrayList<ExampleConfiguration>();
    }

    public void setConfiguration(List<ExampleConfiguration> config) {
        try {
            store.putConfiguration(moduleDescriptor, getXStream().toXML(config).getBytes("UTF8"));
        } catch (UnsupportedEncodingException e) {
            throw new RuntimeException("UTF8 encoding not supported", e);
        }
    }

    private XStream getXStream() {
        XStream xstream = new XStream();
        xstream.setClassLoader(moduleDescriptor.getPlugin().getClassLoader());
        return xstream;
    }
}

[...]
Now that we have access to the configuration data, which describes the repositories, we can go ahead and implement our file system based repository class.

The `SCMRepository` interface offers basic functionality for retrieving file contents of specific file revisions. It is queried by Crucible when a user adds files to a review. Depending on the optional interfaces you implement in addition to `SCMRepository`, your implementation could also have the ability to browse the repository and to explore different versions of each file. Because a standard file system does not store version information, we'll only offer directory browsing in this example. As a revision key or version number we shall simply use the last modification date that is stored by the file system.

```java
package com.atlassian.scm;

import com.atlassian.crucible.scm.SCMRepository;
import com.atlassian.crucible.scm.RevisionData;
import com.atlassian.crucible.scm.RevisionKey;
import com.atlassian.crucible.scm.DetailConstants;
import com.cenqua.crucible.model.Principal;
import java.io.OutputStream;
import java.io.IOException;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStream;
import java.util.Date;
import java.net.MalformedURLException;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import org.apache.commons.io.IOUtils;

public class ExampleSCMRepository implements SCMRepository {
    private final ExampleConfiguration config;

    public ExampleSCMRepository(ExampleConfiguration config) {
        this.config = config;
    }

    public boolean isAvailable(Principal principal) {
        return true;
    }

    public String getName() {
        return config.getName();
    }

    public String getDescription() {
        return getName() + " file system repo at: " + config.getBasePath();
    }

    public String getStateDescription() {
        return "Available";
    }

    public RevisionData getRevisionData(Principal principal,
            RevisionKey revisionKey) {
        if (revisionKey.equals(currentKey(revisionKey.getPath()))) {
            File f = getFile(revisionKey.getPath());
            RevisionData data = new RevisionData();
            data.setDetail(DetailConstants.COMMIT_DATE, new Date(f.lastModified()));
            data.setDetail(DetailConstants.FILE_TYPE, f.isDirectory() ? "dir" : "file");
            data.setDetail(DetailConstants.ADDED, true);
            data.setDetail(DetailConstants.DELETED, false);
            try {
                data.setDetail(DetailConstants.REVISION_LINK, f.toURL().toString());
            } catch (MalformedURLException e) {
            }
            return data;
        }
    }
}
```


```java
public void streamContents(Principal principal, RevisionKey revisionKey,
                          OutputStream outputStream) throws IOException {
    if (revisionKey.equals(currentKey(revisionKey.getPath()))) {
        InputStream is = new FileInputStream(getFile(revisionKey.getPath()));
        try {
            IOUtils.copy(is, outputStream);
        } finally {
            IOUtils.closeQuietly(is);
        }
    } else {
        throw new RuntimeException("Revision " + revisionKey.getRevision() + " of file " +
                                   revisionKey.getPath() + " is no longer available.");
    }
}

public RevisionKey getDiffRevisionKey(Principal principal,
                                       RevisionKey revisionKey) {
    // diffs are not supported in this example
    return null;
}

/**
 * Returns a RevisionKey instance for the specified file. Because we
 * do not support versioning, the revision string will be set to the file's
 * last modification date.
 * @param path
 * @return
 */
private RevisionKey currentKey(String path) {
    File f = getFile(path);
    return new RevisionKey(path, createDateFormat().format(new Date(f.lastModified())));
}

/**
 * Takes the name of a file in the repository and returns a file handle to the
 * file on disk.
 * @param path
 * @return
 */
private File getFile(String path) {
    return new File(config.getBasePath() + File.separator + path);
}

private DateFormat createDateFormat() {
```
In the above code, the `getRevisionData()` method is used by Crucible to retrieve versioning properties for a specific revision of a file in the repository. Although the file system does not keep track of older versions, we can provide some of the properties. Most important are the predefined constants `DetailConstants.FILE_TYPE`, `DetailConstants.ADDED`, `DetailConstants.DELETED` (the last two indicate whether the file was newly created (ADDED), or has been removed from the repository (DELETED) as part of the revision) and `DetailConstants.REVISION_LINK`. In addition to the predefined constants, a repository implementation is free to add custom properties.

We are not able to implement `getDiffRevisionKey()` due to the lack of version information on the file system.

Before we continue to extend the functionality of the `ExampleSCMRepository`, we should go back to `ExampleSCMModule` and implement `getRepositories()`:

```java
[...]
    // initialize at null to trigger loading from the configuration
    private List<SCMRepository> repos = null;

    public synchronized Collection<SCMRepository> getRepositories() {
        if (repos == null) {
            repos = new ArrayList<SCMRepository>();
            for (ExampleConfiguration config : getConfiguration()) {
                repos.add(new ExampleSCMRepository(config));
            }
        }
        return repos;
    }

    public void setConfiguration(List<ExampleConfiguration> config) {
        try {
            store.putConfiguration(moduleDescriptor, xstream.toXML(config).getBytes("UTF8"));
            // we're given a new configuration, so reset our repositories:
            repos = null;
        } catch (UnsupportedEncodingException e) {
            throw new RuntimeException("UTF8 encoding not supported", e);
        }
    }
    [...]
```

Our SCMModule now properly creates the repository instances according to the configuration.

The above code gives us a very simple Crucible SCM plugin. However you would normally also want to implement the `com.atlassian.crucible.scm.DirectoryBrowser` and `com.atlassian.crucible.scm.HasDirectoryBrowser` interfaces. The `DirectoryBrowser` gives Crucible the ability to let the user interactively browse the repository and select files to review. If you do not provide a `DirectoryBrowser`, the only way to create a review for files in your repository is when the required files and file revisions are known up front.

In this example, we'll implement `DirectoryBrowser`: 
public class FileSystemSCMRepository implements HasDirectoryBrowser, DirectoryBrowser {

[...]

public DirectoryBrowser getDirectoryBrowser() {
    return this;
}

public List<FileSummary> listFiles(Principal principal, String path) {
    List<FileSummary> files = new ArrayList<FileSummary>();
    for (String p : list(path, true)) {
        files.add(new FileSummary(currentKey(p)));
    }
    return files;
}

public List<DirectorySummary> listDirectories(Principal principal, String path) {
    List<DirectorySummary> files = new ArrayList<DirectorySummary>();
    for (String p : list(path, false)) {
        files.add(new DirectorySummary(p));
    }
    return files;
}

public FileHistory getFileHistory(Principal principal, String path) {
    return new FileHistory(Collections.singletonList(currentKey(path)));
}

private List<String> list(String path, boolean returnFiles) {
    File parent = getFile(path);
    List<String> files = new ArrayList<String>();
    if (parent.isDirectory()) {
        File[] children = parent.listFiles();
        // this may be null if we can't read the directory, for instance.
        if (children != null) {
            for (File f : children) {
                if (f.isFile() && returnFiles || f.isDirectory() && !returnFiles) {
                    files.add(getPath(f));
                }
            }
        }
    }
    return files;
}

/**
 * @return the path for a given File relative to the base configured for this
 *         repository -- the path doesn't include the base component.
 */
private String getPath(File file) {
    String s = file.getAbsolutePath();
    if (!s.startsWith(config.getBasePath())) {
        throw new RuntimeException("Invalid file with path " + s + " is not under base " +
          config.getBasePath());
    }
    return s.substring(config.getBasePath().length() + 1);
}
[...]

This is as far as we can go with the file system. In most cases you will be integrating version control systems that keep track of all previous revisions of the resources in the repository and you would expose this to Crucible by also implementing HasChangelogBrowser and ChangelogBrowser.

**Servlet Based Administration Pane**

With the code for the module and the repository in place, we can focus on our servlet that provide plugin administration in Crucible's administration section. The easiest way to do this is to subclass
com.atlassian.fisheye.plugins.scm.utils.SimpleConfigurationServlet and implement the three abstract methods:

```java
package com.atlassian.crucible.example.scm;

import com.atlassian.fisheye.plugins.scm.utils.SimpleConfigurationServlet;
import com.atlassian.plugin.PluginAccessor;
import com.atlassian.crucible.spi.FisheyePluginUtilities;

public class ExampleSCMConfigServlet extends SimpleConfigurationServlet<ExampleConfiguration> {
    public ExampleSCMConfigServlet(PluginAccessor pluginAccessor,
                                      FisheyePluginUtilities fisheyePluginUtilities) {
        super(pluginAccessor, fisheyePluginUtilities);
    }

    protected ExampleConfiguration defaultConfig() {
        return new ExampleConfiguration();
    }

    protected String getProviderPluginModuleKey() {
        return "com.atlassian.crucible.example.scm.example-scm-plugin:scmprovider";
    }

    protected String getTemplatePackage() {
        return "/examplescm-templates";
    }
}
```

The `getTemplatePackage()` method returns the name of the resource directory that contains the velocity templates that determine how the configuration pane will be rendered. The template directory must be in `src/main/resources` so Crucible can find them. We'll create three different pages: one that lists the current configuration `list.vm`, one to edit a repository's configuration `edit.vm` and one that is displayed when the user tries to manipulate a non-existing repository instance `nosuchrepo.vm`:

```html
<html>
<head>
    <link rel="stylesheet" href="$request.contextPath/$STATICDIR/main.css" type="text/css" />
</head>
<body class="plugin">
    <div class="box formPane">
        <table class="adminTable">
            #if ($configs.empty)
            <tr><td>No File System repositories are configured.</td></tr>
            #else
            <tr>
                <th>Name</th>
                <th>Base Path</th>
                <!-- for edit link -->
                <!-- for delete link -->
            </tr>
            #foreach ($config in $configs)
            <tr>
                <td>$config.name</td>
                <td>$config.basePath</td>
                <td><a href="./examplescm?name=$config.name">Edit</a></td>
                <td><a href="./examplescm?name=$config.name&amp;delete=true">Delete</a></td>
            </tr>
            #end
            #end
            <tr>
                <td class="verb"><a href="./examplescm?name=_new">Add a repository.</a></td>
            </tr>
        </table>
    </div>
</body>
</html>
```
Finally we tie everything together in the mandatory <atlassian-plugin.xml> file that describes the new plugin, contains its name, location of the servlet and theclassnames Crucible uses to instantiate the components. Because this is an SCM plugin, we must add the <scm/> element:
Packaging, Deploying and Running

Now we can package everything up using `mvn package` and you should end up with `target/example-scm-plugin-1.0-SNAPSHOT.jar` that can be deployed in Crucible by copying the jar file to the `CRUCIBLE_HOME/var/plugins/user` directory. Then login to the administration section, go to `Plugins` and click the link “Check for new plugins in...”: This should detect your plugin and add it to the list in “disabled” state as illustrated below:

**Screenshot: Detecting Your Plugin**

![Plugins](https://example.com/plugins.png)

Next, click “Configure” to create a file system based repository:

**Screenshot: Creating a File-System Based Repository**

![Configure](https://example.com/configure.png)
When the repository is created, navigate to "Repository List". Our custom Crucible SCM Plugin will now show up in the list and is ready to use:

**Screenshot: The Custom SCM Plugin in Crucible**

When reviewing files from the plugin repository, click on the "Manage Files" tab in a new or existing review and then select the repository from the pull down list and select the files and revisions you want to review:

**Screenshot: Selecting Files and Revisions for Review**
The Crucible API

The content on this page is deprecated. Please see the separate documentation space for developer reference material about FishEye and Crucible.

This page covers the Crucible API and how you can use service interfaces that are exposed to plugins.

On this page:

- Services
- Overview
- Using a Service in your Plugin
- Maven dependencies for Spring

Services

Crucible exposes a set of service interfaces to plugins. The parameters and return types of the methods on these interfaces are 'plain old Java objects'. Having an API specifically designed to be used by plugins protects plugins from internal changes in Crucible's implementation, and presents plugins with a simpler API.

Overview

The service interfaces are in the package com.atlassian.crucible.spi.services.

The types the use as parameters are in the package com.atlassian.crucible.spi.data.

Refer to the Crucible API javadoc for details of the services.

Using a Service in your Plugin

The services are all available in your plugin's Spring context.

We can inject a spring bean using constructor injection, e.g.:
public class ExampleServlet extends HttpServlet {
    private ProjectService projectService;

    @Autowired
    public ExampleServlet(ProjectService projectService) {
        this.projectService = projectService;
    }

    ...
}

You can also use setter injection on your plugin class:

    public void setReviewService(ReviewService reviewService) {
        this.reviewService = reviewService;
    }

Note that you cannot mix constructor and setter injection in the same class -- if you mark a constructor with @Autowired, no setters will be used for injection.

All plugin module classes which are created by the plugin system are injected by Spring. That is, the HttpServlet subclass of a servlet plugin module, the SCMModule implementation of a Light SCM plugin module and theEventListener implementation of an event listener plugin module.

Maven dependencies for Spring

If you are using Spring annotations in your plugin you will need the following dependencies in your pom.xml:

<dependencies>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring</artifactId>
        <version>2.5.5</version>
        <scope>provided</scope>
    </dependency>
    <dependency>
        <groupId>com.sun.xml.bind</groupId>
        <artifactId>jaxb-impl</artifactId>
        <version>2.1</version>
        <scope>provided</scope>
    </dependency>
</dependencies>

Crucible Resources

Resources for Evaluators

- Free Trial
- Feature Tour

Resources for Administrators

- Crucible Knowledge Base
- Crucible FAQ
- Tips of the Trade
- Guide to Installing an Atlassian Integrated Suite
- The big list of Atlassian gadgets

Downloadable Documentation

- Crucible documentation in PDF, HTML or XML formats

Plugins
Code review terminology can be confusing as there are many different words for the concepts, roles and process. Crucible has adopted the following terms (click for definitions):

approve
author
code review
comment
creator
defect
moderator
participant
permission
permission scheme
project
review duration
reviewer
role
state
statement of objective
user
approve
Issuing a review to the reviewers is known as approving the review.

**author**

The *author* is the person primarily responsible for acting on the outcomes of the review. In the vast majority of cases the author will be the person who made the code change under review.

Note: to map your repository username to your FishEye/Crucible username, see Changing your User Profile.

**code review**

Without prejudice to 'code inspection', 'peer review' or a myriad of other terms, Crucible uses the phrase code review for simplicity.

See About Crucible and Background Reading.

**comment**

A comment is a short textual note that is linked to a review, revision/diff, source line, or to another comment.

See Adding Comments.

**creator**

The creator is the person who creates the review. In most cases this person will also act as moderator.

**defect**

A defect is a comment flagged as something that requires addressing and includes optional defect classifications.

See Flagging Defects and Customising the Defect Classifications.

**moderator**

The moderator is the person responsible for creating the review, approving the review, determining when reviewing is finished, summarising the outcomes and closing the review. By default, the moderator is the creator.

**participant**

Crucible uses the terms creator, author, moderator, and reviewer to describe the roles of review participants.

**permission**

A permission is the ability to perform a particular action in Crucible, e.g. 'Create Review'. Permissions are assigned to particular users, groups or review roles by means of permission schemes.

The following permissions are available:

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
<th>Default Assignees</th>
</tr>
</thead>
<tbody>
<tr>
<td>'Edit'</td>
<td>Ability to edit a review's details and change the set of revisions being reviewed.</td>
<td>'Creator' 'Moderator'</td>
</tr>
<tr>
<td>Permission</td>
<td>Description</td>
<td>Permissions</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>View</td>
<td>Ability to view a review. (People without this permission will not know that the review exists.)</td>
<td>Anonymous users, All logged-in users</td>
</tr>
<tr>
<td>Abandon</td>
<td>Ability to abandon (i.e. cancel) a review.</td>
<td>'Moderator', 'Creator'</td>
</tr>
<tr>
<td>Re-Open</td>
<td>Ability to re-open a closed or abandoned review.</td>
<td>'Creator', 'Moderator'</td>
</tr>
<tr>
<td>Uncomplete</td>
<td>Ability of a reviewer to change their individual review status from 'Complete' to 'Uncomplete'.</td>
<td>'Reviewer'</td>
</tr>
<tr>
<td>Reject</td>
<td>Ability to reject a review submitted for approval (i.e. prevent it from being issued to reviewers).</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>Complete</td>
<td>Ability of a reviewer to change their individual review status to 'Complete'.</td>
<td>'Reviewer'</td>
</tr>
<tr>
<td>Comment</td>
<td>Ability to add or remove a comment to or from a review.</td>
<td>'Creator', 'Author', 'Reviewer', 'Moderator'</td>
</tr>
<tr>
<td>Approve</td>
<td>Ability to approve a review (i.e. issue it to the reviewers).</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>Submit</td>
<td>Ability to submit a review for approval (i.e. request that the review be issued to the reviewers).</td>
<td>'Creator', 'Author'</td>
</tr>
<tr>
<td>Close</td>
<td>Ability to close a review once it has been summarised.</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>Delete</td>
<td>Ability to delete a review.</td>
<td>'Creator', 'Moderator'</td>
</tr>
<tr>
<td>Summarise</td>
<td>Ability to summarise a review. (Normally this would be done after all reviewers have completed their review.)</td>
<td>'Moderator'</td>
</tr>
<tr>
<td>Create</td>
<td>Ability to create a review.</td>
<td>All logged-in users</td>
</tr>
<tr>
<td>Recover</td>
<td>Ability to resurrect an abandoned (i.e. cancelled) review.</td>
<td>'Creator', 'Moderator'</td>
</tr>
</tbody>
</table>

permission scheme

A permission scheme assigns particular permissions to any or all of the following:

- Particular Users.
- Particular Groups.
- All logged-in users.
- Anonymous Users
- People in particular Review Roles, such as:
  - 'Author';
  - 'Reviewer';
  - 'Creator';
  - 'Moderator'.

The scheme's permissions will apply to all reviews belonging to the project(s) with which the scheme is associated.

You can create as many permission schemes as you wish. Each permission scheme can be associated with many projects or just one project, allowing you to tailor appropriate permissions for individual projects as required.

See Creating a Permission Scheme.

project
A Crucible project is a collection of reviews, typically reviews that all relate to the same application. In addition to providing a logical way of grouping reviews together, a project allows you to

- define default moderators, authors and reviewers for the reviews in that project.
- define which people are eligible to be reviewers for the reviews in that project.
- use permission schemes to restrict who can perform particular actions (e.g. 'Create Review') in that project.

Every Crucible review belongs to a project. Each project has a name (e.g. ACME Development) and a key (e.g. ACME). The project key becomes the first part of that project's review keys, e.g. ACME-101, ACME-102, etc:

By default, Crucible contains one project. This default project has the key 'CR' and the name 'Default Project'.

See Creating a Project.

review duration

The review duration is the period of time for which a review will run.

See Setting the Default Review Duration for a Project.

reviewer

A reviewer is a person assigned to review the change. Reviewers can make comments and indicate when they have completed their review. The moderator and author are implicitly considered to be participants of the review, but are not reviewers.

role

See participant.

state

A Crucible review moves through the following states in the following sequence:

<table>
<thead>
<tr>
<th>State</th>
<th>See</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft</td>
<td>Creating a Review.</td>
</tr>
<tr>
<td>Require Approval</td>
<td>Relevant only when the moderator is not the creator. See Issuing a Review.</td>
</tr>
<tr>
<td>Under Review</td>
<td>See Issuing a Review and Reviewing the Code.</td>
</tr>
<tr>
<td>Summarize</td>
<td>See Summarising and Closing the Review.</td>
</tr>
<tr>
<td>Closed</td>
<td>See Summarising and Closing the Review.</td>
</tr>
</tbody>
</table>

Tips: Reviews can be re-opened, i.e. moved from Summarize or Closed back to Under Review.

A review may also be in the following states:

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abandoned</td>
<td>This happens when a review is deleted.</td>
</tr>
<tr>
<td>Rejected</td>
<td>Any reviews that a moderator has rejected.</td>
</tr>
</tbody>
</table>

statement of objective

A statement of objective is an optional text description of the review and any specific areas the reviewers should focus on.

user

A user is a person using Crucible.
Contributing to the Crucible Documentation

Would you like to share your Crucible hints, tips and techniques with us and with other Crucible users? We welcome your contributions.

On this page:

- Blogging your Technical Tips and Guides - Tips of the Trade
- Contributing Documentation in Other Languages
- Updating the Documentation Itself
  - Getting Permission to Update the Documentation
  - Following our Style Guide
  - How we Manage Community Updates

Blogging your Technical Tips and Guides – Tips of the Trade

Have you written a blog post describing a specific configuration of Crucible or a neat trick that you have discovered? Let us know, and we will link to your blog from our documentation. More....

Contributing Documentation in Other Languages

Have you written a guide to Crucible in a language other than English, or translated one of our guides? Let us know, and we will link to your guide from our documentation. More....

Updating the Documentation Itself

Have you found a mistake in the documentation, or do you have a small addition that would be so easy to add yourself rather than asking us to do it? You can update the documentation page directly.

Getting Permission to Update the Documentation

Our documentation wiki contains developer-focused documentation (such as API guides, plugin and gadget development guides and guides to other frameworks) as well as product documentation (user's guides, administrator's guides and installation guides). The wiki permissions are different for each type of documentation.

- If you want to update the Crucible developer documentation, the Developer Network or other developer-focused wiki spaces, just sign up for a wiki username then log in and make the change.
- If you want to update the Crucible product documentation, we ask you to sign the Atlassian Contributor License Agreement (ACLA) before we grant you wiki permissions to update the documentation space. Please read the ACLA to see the terms of the agreement and the documentation it covers. Then sign and submit the agreement as described on the form attached to that page.

Following our Style Guide

Please read our short guidelines for authors.

How we Manage Community Updates

Here is a quick guide to how we manage community contributions to our documentation and the copyright that applies to the documentation:

- Monitoring by technical writers. The Atlassian technical writers monitor the updates to the documentation spaces, using RSS feeds and watching the spaces. If someone makes an update that needs some attention from us, we will make the necessary changes.
- Wiki permissions. We use wiki permissions to determine who can edit the various types of documentation spaces.
  - Developer documentation (API guides, plugin development and gadget development): Anyone can edit these spaces, provided they have signed up for a wiki username and logged in to the wiki.
  - Product documentation (user's guides, administrator's guides, installation guides): We ask people to sign the Atlassian Contributor License Agreement (ACLA) and submit it to us. That allows us to verify that the applicant is a real person. Then we give them permission to update the documentation.
- Copyright. The Atlassian documentation is published under a Creative Commons ‘cc-by’ license. Specifically, we use a Creative Commons Attribution 2.5 Australia License. This means that anyone can copy, distribute and adapt our documentation provided they acknowledge the source of the documentation. The cc-by license is shown in the footer of every page, so that anyone who contributes to our documentation knows that their contribution falls under the same copyright.

RELATED TOPICS

Tips of the Trade
Author Guidelines
Tips of the Trade

Below are some links to external blog posts and articles containing technical tips and instructions on setting up and using Crucible. This page presents an opportunity for customers and community authors to share information and experiences.

The references here are specific to Crucible and are technical 'how to' guides written by bloggers who use Crucible. For more general information on code review solutions, best practices and business cases, please refer to the Atlassian website.

Please be aware that these are external blogs and articles. Most of the links point to external sites, and some of the information is relevant to a specific release of Crucible. Atlassian provides these links because the information is useful and relevant at the time it was written. Please check carefully whether the information is still relevant when you read it, and whether it is relevant to your version of Crucible. Unless explicitly stated, Atlassian does not offer support for third-party extensions or plugins. The information in the linked blog posts has not been tested or reviewed by Atlassian. We recommend that you test all solutions on a test server before trying them on your production site.

On this page:

- Running Atlassian Crucible (or Fisheye) on Linux Startup
- Reviewing wiki documentation via Crucible

Running Atlassian Crucible (or Fisheye) on Linux Startup

- By: Jarrod, on blog 'The stupid people made me do it!'  
- About: How to get Crucible to start automatically when the server boots, via an initialisation script that runs on RedHat-like Linux boxes  
- Date and Crucible version: 9 March 2009; Crucible 1.6  
- Related documentation: Auto Start on Windows via a Windows Service

Reviewing Confluence Pages

Reviewing wiki documentation via Crucible

- By: Sarah Maddox, on blog 'ffeathers'  
- About: Setting up Crucible and Confluence so that you can use Crucible to review Confluence wiki pages  
- Date and software versions: 17 January 2009; Crucible 1.6 and Confluence 2.10  
- Related documentation:  
  - Setting Up Reviewing of Confluence Pages in Crucible  
  - Confluence SCM Plugin

Have you written a technical tip for Crucible?

Add a comment to this page, linking to your blog post or article. We will include it if the content fits the requirements of this page.

Feedback?

Your first port of call should be the author of the linked blog post. If you want to let us know how useful (or otherwise) a linked post is, please add a comment to this page.

Other Sources of Information

- Crucible documentation  
- Atlassian website  
- Atlassian forums  
- Atlassian Blog  
- Crucible plugins

Crucible Documentation in Other Languages
Below are some links to Crucible documentation written in other languages. In some cases, the documentation may be a translation of the English documentation. In other cases, the documentation is an alternative guide written from scratch in another language. This page presents an opportunity for customers and community authors to share documentation that they have written in other languages.

⚠️ Please be aware that these are external guides.
Most of the links point to external sites, and some of the information is relevant to a specific release of Crucible. Atlassian provides these links because the information is useful and relevant at the time it was written. Please check carefully whether the information is still relevant when you read it, and whether it is relevant to your version of Crucible. The information in the linked guides has not been tested or reviewed by Atlassian.

On this page:
- No guides yet

<table>
<thead>
<tr>
<th>None</th>
</tr>
</thead>
</table>

No guides yet

We do not yet have any guides to link here. Be the first to suggest one!

Adding Your Own Guide to this Page

Have you written a guide for Crucible in another language? Add a comment to this page, linking to your guide. We will include it if the content fits the requirements of this page.

Giving Feedback about One of the Guides

If you have feedback on one of the guides listed above, please give the feedback to the author of the linked guide.

If you want to let us know how useful (or otherwise) one of these guides is, please add a comment to this page.

Other Sources of Information

- Crucible documentation
- Atlassian website
- Atlassian blog
- Crucible plugins

TreeNavigation

Index

Changeset Discussions

When using Crucible with FishEye, you can have threaded discussions with other users, on any changeset. To start a discussion, you simply start by adding a comment to a changeset.

⚠️ You need to be logged in to create changeset comments.

Adding Comments to Changesets

To add a comment to a changeset,
1. Open the changeset view for the desired code commit.
2. Display comments by clicking 'Discuss' at the upper right corner, or the speech bubble icon beside the left navigation bar.
3. When the comment bar is visible, you can add a comment by clicking 'Add a Comment'. Type your content and click 'Post' to submit it.
4. You can tag your comment as a defect note by clicking the 'Defect' tick box.
5. Once submitted, others can respond to your comment by clicking 'Reply'. Replies are threaded as separate comment discussions. You can click on the link icon to save a permalink to that comment. The comment author can edit or delete their comments.
6. To hide the changeset comments, click the page icon. You can open the comments bar by clicking the speech bubble icon again.

As you compose a comment, it will auto-save periodically.

Screenshot: Opening Changeset Discussions

Screenshot: Composing a Changeset Comment

Screenshot: A Changeset Discussion
Turning Changeset Discussions On and Off

You can turn off changeset discussions in the Admin screens.

Go to the Admin screen, then choose 'Repository List' from the left navigation bar. Find your repository from the list that appears, and click 'View' beside it to see the repository settings page. Scroll to the bottom of the list and find 'Changeset Discussions'. Click 'Edit' to change the value to true or false. If set to false, changeset discussions are disabled.

By default, changeset discussions are on.

Notifications

- Comments show up in the activity stream,
- The author of the changeset will get email notifications when comments are added,
- Comment authors will get email notifications when someone replies to their comments.

TreeNavigationVersions

- Click for all versions