1. FishEye Documentation Home ................................................................. 5
  1.1 FishEye 101 ................................................................................. 6
  1.2 FishEye Installation & Upgrade Guide ........................................... 15
    1.2.1 FishEye Installation Guide ...................................................... 15
      1.2.1.1 System Requirements ...................................................... 15
        1.2.1.1.1 Setting JAVA_HOME .................................................. 18
        1.2.1.1.2 Setting JVM System Properties ................................ 19
        1.2.1.2 Installing FishEye ......................................................... 19
        1.2.1.3 Configuring FishEye ....................................................... 21
        1.2.1.4 Best Practices for FishEye Configuration ....................... 22
    1.2.2 FishEye Upgrade Guide ......................................................... 23
  1.3 FishEye User's Guide ..................................................................... 25
    1.3.1 Fisheye Quick Start Guide ..................................................... 25
    1.3.2 Using the FishEye Screens ..................................................... 27
      1.3.2.1 Browsing through a Repository ...................................... 30
      1.3.2.2 Viewing the Changelog ................................................... 31
      1.3.2.3 Viewing a File History .................................................... 33
      1.3.2.4 Searching the Repository .............................................. 34
      1.3.2.5 FishEye Charts .............................................................. 40
      1.3.2.6 Using Favourites ........................................................... 44
      1.3.2.7 Viewing People's Statistics .......................................... 47
      1.3.2.8 Viewing File Content ................................................... 48
    1.3.3 Changing your User Profile ................................................... 49
      1.3.3.1 Re-setting your Password ............................................. 52
    1.3.4 EyeQL Reference Guide ....................................................... 53
    1.3.5 Antglob Reference Guide ..................................................... 57
    1.3.6 Date Expressions Reference Guide ..................................... 58
  1.4 FishEye Administrator's Guide ..................................................... 59
    1.4.1 Managing your Repositories ................................................. 60
      1.4.1.1 Adding a Repository ..................................................... 61
        1.4.1.1.1 ClearCase ........................................................... 61
        1.4.1.1.2 CVS ................................................................. 66
        1.4.1.1.3 Git ................................................................. 67
        1.4.1.1.4 Perforce .......................................................... 68
1.6.10.11 On my Red Hat Linux system, after running for several days FishEye freezes and does not accept any more connections.

1.6.10.12 Problems with very long comments and MySQL migration

1.6.10.13 URLs with encoded slashes don't work, especially in Author constraints

1.7 Glossary

1.8 About FishEye
# FishEye Documentation Home

## Getting Started with FishEye 2.1

- Release Notes (What's new)
- FishEye 101 | Quick Start Guide | Installation Guide
- Download FishEye

## Using/Administering FishEye 2.1

- User's Guide
- Administrator's Guide
- Upgrade Guide

## FishEye IDE Integration

Use the Atlassian Connector for Eclipse or the Atlassian Connector for IntelliJ IDEA to work with FishEye and Crucible right there in your development environment.

## Resources

- FishEye Knowledge Base
- Support | Forums | FAQ
- Development Hub | EyeQL Reference Guide
- Guide to Installing an Atlassian Integrated Suite
- Crucible Documentation (Code Review Tool)

## Previous Versions

- FishEye 2.0 documentation
- FishEye 1.6 documentation
- FishEye 1.5 documentation
- FishEye 1.4 documentation
- FishEye 1.3 documentation
- FishEye 1.2 and older documentation

## Offline Documentation

You can download the FishEye documentation in PDF, HTML or XML format for use offline.

**Recently Updated**
FishEye 101

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

FishEye 101

Thanks for taking the time to try FishEye. To help you make the most of your time, we've taken the liberty of compiling some easy instructions for configuring and using FishEye.

Software developers are the intended audience for this document.

FishEye 20 Minute Setup

Setting up FishEye takes less than half an hour.

1. Install

First things first, if you haven't already got FishEye up and running carry out the relevant steps below:
For Windows: (click to expand)

Check System Requirements before you begin.

1. Download FishEye from the Atlassian Download Center
2. Unzip the downloaded package to the desired installation directory: `c:\fisheye` (see the documentation if you need help with this).
3. Launch FishEye from the command line, typing `start.bat`. A FishEye console window will appear, listing the startup progress. Give it a few minutes to complete its process. When the message "server started" is displayed, you can continue.
   - You will also receive a warning in the console output about a 'missing license'. This is normal.
4. On the same machine go to `http://localhost:8060/` (or type the hostname or IP address in place of `localhost` from another machine) in your web browser. You will be prompted to enter a license key (obtained at `http://my.atlassian.com`) and set the admin password. (Tip: you can just use one Crucible evaluation license key to try both FishEye & Crucible.)

It should only take you a few minutes to have a running instance of FishEye. If you have any difficulty during the installation process (i.e. it takes longer than twenty minutes) please contact our support team for assistance. They're ready to help at any time.

For Linux: (click to expand)

Check System Requirements before you begin.

1. Download FishEye from the Atlassian Download Center
2. Unzip the downloaded package to the desired installation directory: `/opt/fecru-x.x.x` (see the documentation if you need help with this).
3. Launch FishEye from the command line, typing `./run.sh`. Console output will appear, listing the startup progress. Give it a few moments to complete its process. When the message "server started" is displayed, you can continue.
   - You will also receive a warning in the console output about a 'missing license'. This is normal.
4. On the same machine go to `http://localhost:8060/` (or type the hostname or IP address in place of `localhost` from another machine) in your web browser. You will be prompted to enter a license key (obtained at `http://my.atlassian.com`) and set the admin password. (Tip: you can just use one Crucible evaluation license key to try both FishEye & Crucible.)

It should only take you a few minutes to have a running instance of FishEye. If you have any difficulty during the installation process (i.e. it takes longer than twenty minutes) please contact our support team for assistance. They're ready to help at any time.

2. Scan

FishEye's fast performance is built on the way it scans repositories. (click to expand)
Once installed, FishEye needs to do an initial index, where it accesses, indexes and organizes a view of your repository (including all historical items) back to the earliest commit.

FishEye operates in a completely read-only mode, so your repository data is safe. Also, always run FishEye on an SCM account that is limited to read-only access.

Click on the 'Administration' link in the footer bar to access the administration interface, then click 'Add Repository'.

There are lots of options when adding a repository, but in most situations you should be able to run with the defaults. For evaluation purposes, we recommend that you just index one project, which is usually a subset of your whole repository (rather than everything). If you choose to index your entire repository, just be aware that this can take a long time (possibly days) for massive or complex repositories and can be more complex to set up (especially for Subversion). Indexing just one project, you can have FishEye up and running quickly for evaluation. The basic process is slightly different for each SCM type. Carry out the respective process for your SCM below:

- Subversion set the Path field to a subdirectory that includes the directories trunk, branches, tags for your project
- Perforce & Git set the Path to a subdirectory that contains your project's source.
- CVS set the CVS dir to point to your project.

The default indexing settings can have challenges digesting non-standard SCM configurations. If the initial index appears to hang, please call support as this usually indicates you have a configuration problem, that can be quickly resolved by our friendly support engineers.

3. Add Users

You can automatically import users from an LDAP directory or similar. (click to expand)

FishEye was designed to improve team communication through sharing SCM artifacts, so you will probably want to add accounts for all your team members. There are a few approaches that you can take to adding users:

1. Leave FishEye in anonymous mode. There are two downsides here, anyone who can see FishEye, can see all the source and you won't be able to use email watches or save preferences.
2. Built-in users with "public sign up". Anyone can create an account, but not especially secure.
3. Built-in users. This is simple to set up for a small user base, however adding each user manually is not recommended for large-scale systems with many users.
4. External authentication. This includes sources such as LDAP, Active Directory or similar.
   a. Atlassian Crowd. If you already have a Crowd server, this is a snap to set up.
   b. LDAP/Active Directory. You can point FishEye at your existing corporate directory. If you are familiar with LDAP, this should be easy.
   c. Host/PAM Authentication. You can use the local accounts from the physical machine FishEye is running on.

If you opt for accounts, be sure to make yourself and other relevant users FishEye administrators. You may also be interested in pre-loading users from LDAP/Crowd and setting repository level access control based on groups.

4. Configure Mail (optional)
Email notifications are a great way to keep up with FishEye activity. (click to expand)

FishEye 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 FishEye about your SMTP server.

Configure JIRA (optional)

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

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

Setup Complete!

Your FishEye 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.

If you do need assistance, our support team is ready to help. Create an issue at our support system under the FishEye project.

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

Browsing Your Code Repository

In essence, FishEye is a tool that lets you view the contents of your Source Code Management (SCM) repository as a web page. Everything in your SCM is accessible in your web browser, quickly and easily.

Browsing & History

Power-browse your repository. (click to expand)

Fisheye’s Windows Explorer-style interface allows you to efficiently navigate your source tree and view the depths of your repository in a structured way. Select a file to view its entire revision history. See the documentation for more.

Full Source View
_view the full source of any file revision. (click to expand)

See the blame information associated with every line (allowing you to pinpoint who is responsible for every change). Author and age can be displayed as an aid to annotation. See the documentation for more.

9. Changesets & Diffs

See every change and collaborate anywhere on the web. (click to expand)

Every commit and any diff can be viewed and easily linked to. This removes ambiguity from discussions in other mediums (such as discussions held via instant messaging, wiki pages, email, issue trackers and so on). See the documentation for more.

Activity & People

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

10. Activity Streams

You can see commits and updates from the users in FishEye 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.

11. People Lists

In FishEye, 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.

12. 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 additions to the repository, showing details of their work and summaries of their activity. Additionally, you can see their work on tracked issues and code reviews if using FishEye with Crucible and JIRA integration is set up. See the documentation for more.
13. 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 personal code commits; your reviews (if you are using Crucible) and your tracked issue updates (if you are using JIRA). See the documentation for more.

14. Filtering The Changelog

- Quickly find what you're looking for. (click to expand)

Find the changes that you're seeking by filtering commits based on log message, path, author, date, branch (and other fields). This control is a snappy filter button under the Source tab. See the documentation for more.

Reports & Search

Aggregate, audit and present your repository data with FishEye’s powerful search and report functions.

15. Embedded Charts

- FishEye now shows charts in various locations. (click to expand)

Throughout the user interface, there are visualisations to help you view the activity in this area. The ubiquitous chart in FishEye plots net lines of code (that's just lines not NLOC for those who are interested) on the top and commit volume (the number of files committed) on the bottom. Note that this chart is generated on the fly based on what you are looking at, from the root down to individual files. On the changelog page the green bar shows you the period that is covered by the changesets on the page you are viewing. Clicking on the chart takes you to that point in time in your changelog.

See the documentation for more.

16. Custom Charts

- You can define specific metrics to be charted within FishEye. (click to expand)
To start using the more complex charts, click on the Chart tab on the right of the header, or "jump to detailed chart" on any of the baby charts. There is a lot of really cool stuff in these charts.

Some things to try:

- Find out who your biggest contributors are - select "Show by Author" as a Pie chart.
- See what file types you have committed - select "Show by Extension" then select your user ID as in the Author field.
- See who has been changing the most lines in the last month - clear your selections and select "Show by Author" and a date a month prior in the "Start Date" field. If you have a lot of source the chart will probably be flat. If so, change the chart type to "Change", this will zero the line count at the start date.
- Compare your changes against another person - pick yourself and another author (hold the 'control' key to select multiple) in the author list. This isn't very useful but can be a bit of fun!

Note that the committer gets a line for each line they add or change. The previous owner of a modified or deleted line loses one. Currently any change, even whitespace, counts.

See the documentation for more.

17. Plugin Reports

FishEye is extensible, allowing you to create your own reports. (click to expand)

You can create your own report as a plugin and have it appear in the menu, under the 'Source' tab, and the 'Reports' sub-tab. For example, the 'Code Metrics' option that appears in the user interface is actually a plugin, added in to FishEye. You can use this plugin as a basis to create your own custom reports and add them in to FishEye. See the FishEye developer documentation for more.

18. Quick Search

FishEye supports some powerful search commands. (click to expand)

The box in the top right corner of every page can be used to quickly search the repository. You can use syntax like author:anna to immediately return results which have "anna" in the author field. See the documentation for more.

19. Running Queries & EyeQL

You can create complex queries with FishEye's own query language. (click to expand)

Clicking 'Search' on the top navigation bar opens the Simple Search screen. Here, you have access to a wide range of powerful searching functionality including file content searching, and grouping results by changeset, revision, file or directory.

Sometimes, you need data from your repository retrieved and sorted in ways that are unique to your own situation, beyond what the pre-fab GUI can give you. From the Simple Search screen, click 'Switch to Advanced Search'. Here, you can build searches using FishEye's powerful embedded query language, called EyeQL.

See the search documentation and EyeQL Reference Guide for more.
20. Leveraging CSV and the REST API

You can export data and extend FishEye programmatically. (click to expand)

When you need to extract repository data for a report, you can export the results of FishEye searches to a CSV (comma-separated values) file which imports directly into spreadsheets and other office applications. Simply select the radio buttons at the bottom of the Simple Search dialog and the CSV file will be generated automatically, prompting you to save the file.

With its own API (Application Programming Interface), FishEye is extensible. If you need FishEye 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.

Tips and Suggestions

Learn about some of the smart labour-saving features in FishEye.

21. Quick Nav

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. FishEye'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 CSID to go directly to that page. See the documentation for more.

22. Favorites as Bookmarks

Everything in FishEye can be bookmarked. (click to expand)

See the documentation for more.

23. 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 FishEye to see the context windows:

- FishEye usernames.
- JIRA Issue Keys (When using JIRA; see the documentation).
- Crucible Review Keys (When using Crucible Documentation Home).

24. Subscribe to SCM Updates
Keep track of FishEye activity when your FishEye session is closed. (click to expand)

Every user can keep an eye on changes from RSS Feeds and Email Watches to the source directories that interest them or even individual files. The idea is to encourage people to subscribe to the level of notifications that suits them so the signal isn't lost in the noise. Here are two ways to test this out:

1. Set up a customised feed by going navigating to the changelog page. On the upper right hand corner of the changelog box, click on the Customised Feed URL. Fill in the feed attributes and use the constraints to get data about a specific users contributions to a specific branch of code. Select to subscribe either as an RSS feed or an email watch. For fun, you can comment on their code the moment they commit until they can't take it anymore.

2. Simulate a spontaneous watch request by surfing through the directories. Once you find an interesting contribution, hit the RSS button on the upper right hand corner of the changelog box to get updated on that particular file.

See the documentation for more.

Advanced Features to Try

25. Send Links to Your Code

Everything in FishEye can be linked. (click to expand)

A changeset, a diff, even a specific line in a specific revision can be linked. You'll find a lot of stuff is underlined. Keep this in mind as you evaluate FishEye, there are plenty of links that are crying out to be copied to clipboard and sent on to your teammates.

If you look at the URL structure, you'll see that FishEye URLs are very predictable and hence can be easily generated by hand. There are also some extra wrinkles that are great for bookmarks.

Useful Hacks:

- Every line in an annotation is a permalink to that line, clicking the revision number in the gutter takes you to that line in the revision it was last changed (even when the line number is different). Add a post commit hook in your SCM that gives developers the FishEye link as soon as they commit.
- Using the keyword HEAD will give you the latest version of file.
- Use branch or tag names in place of revision numbers in diffs.

See the documentation for more.
The FishEye developers try and set sensible defaults so that everything "just works" in most situations. However, if you are a power user, there are lots of knobs to twiddle. Check out the documentation below to learn about the nitty gritty.

- Customize the welcome page
- Enable tarballs
- Proxy FishEye through a separate web server
- Add more indexing threads
- Throttle repository access
- Software update notifications
- ViewVC compatibility

Thank you for reading this guide.

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

FishEye Installation & Upgrade Guide

- FishEye Installation Guide
  - System Requirements
    - Setting JAVA_HOME
    - Setting JVM System Properties
  - Installing FishEye
  - Configuring FishEye
  - Best Practices for FishEye Configuration
- FishEye Upgrade Guide

FishEye Installation Guide

This guide describes the advanced installation options that can be used when installing FishEye.

Evaluate FishEye for the first time? See the FishEye 101 page.

- System Requirements
- Installing FishEye
- Configuring FishEye
- Best Practices for FishEye Configuration

System Requirements

This page contains system requirements for FishEye.

On this page:

- Java Environment and Operating System
- Operating System
FishEye 2.1 Documentation

- Platform Hardware Requirements
- Version Control System
- External Databases
- Web Browser
- Deployment
- Single Sign On with Atlassian Crowd

**Java Environment and Operating System**

- **Java Runtime** (JDK or JRE) version 1.5 or greater. (Solaris requires 1.5.0_15 as a minimum)

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.

**Note:** There appeared to be a problem with some releases of the JRockit JVM that causes corrupted caches in FishEye. This problem has been confirmed in the following versions:

- JRockit 5.0 JVM (R25.0.0-75)
- JRockit 5.0 JVM (R27.3.0)

**Operating System**

FishEye is a pure Java application and should run on any platform provided the above requirements are satisfied.

**Platform Hardware Requirements**

FishEye should ideally run on a standalone dedicated server. The most important aspect for a large-repository deployment will be I/O speed. You definitely want a fast local HDD for FishEye's cache (not NFS or SAN).

<table>
<thead>
<tr>
<th>Component</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>1.8GHz or higher, a single core is sufficient. More cores or higher GHz will result in better load-handling ability.</td>
</tr>
<tr>
<td>RAM</td>
<td>1GB minimum, 2GB will provide performance &quot;headroom&quot;. Your Java heap should be sized at 512MB with the FISHEYE_OPTS environment variable, adjustable up to 1024MB depending on performance.</td>
</tr>
<tr>
<td>I/O</td>
<td>FishEye's input/output is an important element of its overall performance. If FishEye accesses your repository remotely, make sure that the throughput is maximum and the latency minimum (ideally the servers are located in the same LAN, running at 100Mbps or faster).</td>
</tr>
</tbody>
</table>

⚠ While some of our customers run FishEye on SPARC-based hardware, Atlassian only officially supports FishEye running on x86 hardware and 64-bit derivatives of x86 hardware.

**Disk Space Requirement Estimates**

Disk space requirements for FishEye may vary due to a number of variables such as the repository implementation, file sizes, content types, the size of diffs and comments being stored. The following table contains some real-world
examples of FishEye disk space consumption.

<table>
<thead>
<tr>
<th>Repository Technology</th>
<th>Commits</th>
<th>Codebase Size (HEAD of trunk)</th>
<th>FishEye Index Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subversion</td>
<td>14386</td>
<td>466 MB in 12151 files</td>
<td>647 MB</td>
</tr>
<tr>
<td>CVS</td>
<td>8210</td>
<td>115 MB in 11433 files</td>
<td>220 MB</td>
</tr>
</tbody>
</table>

⚠️ These disk space estimates are to be used as a guideline only. We recommend you monitor the disk space that your FishEye instance uses over time, as needs for your specific environment may vary. It may be necessary to allocate more space than indicated here. Additionally, you can reduce disk space consumption by turning off diff storage in FishEye.

**Version Control System**

At this time, FishEye supports the following source code management (SCM) systems:

- Subversion
- Perforce
- CVS (and CVS-NT)
- Git
- IBM ClearCase

**Deployment Notes for Version Control Systems**

<table>
<thead>
<tr>
<th>Subversion (server)</th>
<th>FishEye can communicate with any repository running Subversion 1.1 or later.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subversion (client)</td>
<td>FishEye now bundles the SNVkit client, which becomes the default Subversion interface. An alternative is to use the native subversion client, using JavaHL bindings. Please see Subversion Client Setup for more information.</td>
</tr>
<tr>
<td>Perforce (client)</td>
<td>FishEye needs access to the p4 client executable. Due to some problems with earlier versions of the client, we recommend version 2007.3 or later.</td>
</tr>
<tr>
<td>CVS</td>
<td>If you are using CVS, FishEye needs read-access to your CVS repository via the file system. It does not support protocols such as pserver at the moment.</td>
</tr>
</tbody>
</table>

Support for other version control systems is planned for future releases. Let us know what SCM system you would like to see supported by creating a JIRA issue or adding your vote to an issue, if the request already exists.

**External Databases**

At this time, FishEye supports the following external databases:

- MySQL 5.x onwards
- PostGreSQL 8.x onwards.

**Web Browser**

Fisheye has been tested with Firefox 3, Internet Explorer 7, and Safari 4. IE6 is NOT supported. Fisheye should work with most modern browsers.

⚠️ IE6 is not supported and should not be used with Crucible.
FishEye/Crucible is currently a standalone Java program. It cannot be deployed to web application servers such as WebSphere, Weblogic or Tomcat.

**Single Sign On with Atlassian Crowd**

FishEye is bundled with the Crowd 1.3 client library, and therefore is intended to operate with Crowd 1.3 or later.

✅ **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.

**Setting JAVA_HOME**

Once you have installed the JDK (see System Requirements), you need to set the JAVA_HOME environment variable.

To set the JAVA_HOME environment variable on Windows

1. Right click on the 'My Computer' icon on your desktop and select 'Properties'.
2. Click the 'Advanced' tab.
3. Click the 'Environment Variables' button.
4. Click 'New'.
5. In the 'Variable name' field, enter 'JAVA_HOME'.
6. In the 'Variable value' field, enter the directory (including its full path) where you installed the JDK.
7. Restart the computer.

To set the JAVA_HOME environment variable on Linux or UNIX based systems

There are many ways you can do it on Linux or UNIX based systems (including Mac OS X). Here are two:

For your current user,

1. Open up a shell / terminal window
2. vi ~/.profile (replace vi with your favourite text editor)
3. Add export JAVA_HOME=/path/to/java/home/dir on its own line at the end of the file
4. Add export PATH=$JAVA_HOME/bin:$PATH on its own line immediately after
5. Save, and restart your shell
6. Running `java -version` should give you the desired results

For all users in the system,

1. Open up a shell / terminal window
2. vi /etc/profile (replace vi with your favourite text editor)
3. Add export JAVA_HOME=/path/to/java/home/dir on its own line at the end of the file
4. Add export PATH=$JAVA_HOME/bin:$PATH on its own line immediately after
5. Save, and restart your shell
6. Running `java -version` should give you the desired results

If you are using a GUI, you may not need to open up the shell. Instead, you might be able to open the file directly in a graphical text editor.
If you are experiencing memory errors in FishEye, see Fix Out of Memory errors by increasing available memory.

Setting JVM System Properties

FishEye and Crucible use a number of Java Virtual Machine System properties. Most of these are properties set by the virtual machine itself or the FishEye/Crucible launch script. Users can configure the following property to control low level aspects of FishEye's operation:

- jetty.http.headerbuffersize - This controls the size of the largest HTTP header value that FishEye will allow (through its embedded servlet engine, Jetty). Some authentication systems require larger header values.

Setting a Larger Header Buffer Size

In FishEye 1.6.5 onwards, you can set a jetty.http.headerbuffersize system property (in bytes) to adjust the header size. This can be set by passing the -D parameter to the JVM, or by adding the property to FISHEYE_INST/system.properties. This file can be created as a plain text file (it follows the Java .properties format) if you need to use this setting.

The recommended approach is to add the property to FISHEYE_INST/system.properties. Add this as a new text file, in the Java .properties format).

Users integrating FishEye with Single Sign On applications may require an adjustment to the JVM properties, specifically the HTTP header size. This may also be useful in other circumstances where FishEye's default HTTP header is too small (at 4096 bytes).

Installing FishEye

This guide describes the advanced FishEye installation options.

1. Download the FishEye zip file and extract it. This document assumes you have extracted FishEye to /FISHEYE_HOME/.
2. Ensure you have installed an appropriate Java runtime - see System Requirements.
3. Ensure that java is in the PATH, or that the JAVA_HOME environment variable is set.
4. If you intend to use FishEye with Subversion, please ensure you read the System Requirements, Subversion client setup, and granting permission to FishEye to scan your repository.
5. If you intend to use FishEye with Perforce, please ensure you read the Perforce client setup.

Read-only access for FishEye

We recommend you run FishEye as a user that has only read access to your repository.

FishEye Folder Layout

**FISHEYE_HOME (Default)**

By default, FishEye will run self-contained within the /FISHEYE_HOME/ directory. The FishEye directory layout looks like this:

<p>| /FISHEYE_HOME/config.xml | Configuration file. |</p>
<table>
<thead>
<tr>
<th>Directory Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/FISHEYE_HOME/var/</code></td>
<td>Directory under which FishEye stores its data.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/var/data/</code></td>
<td>Persistent information.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/var/cache/</code></td>
<td>Caches and indexes.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/var/log/</code></td>
<td>Log files.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/var/tmp/</code></td>
<td>Temporary files.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/bin/</code></td>
<td>Scripts for controlling FishEye.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/lib/</code></td>
<td>FishEye's dependent libraries.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/ ...</code></td>
<td>Remainder omitted for brevity.</td>
</tr>
</tbody>
</table>

However, this self-contained layout results in tedious copying of files each time you upgrade FishEye. Also, if you want to run multiple instances of FishEye, you need multiple `/FISHEYE_HOME/` installations. These two issues can be avoided by setting a FISHEYE_INST ('FishEye Instance') location.

### FISHEYE_INST (Optional)

FISHEYE_INST is a location where your repository data is stored, separate from the installation location of the FishEye application. This may be necessary for practical or administrative reasons.

A separate FISHEYE_INST location is recommended for production installations of FishEye. Once you have have declared your FISHEYE_INST, you will need to copy your FISHEYE_HOME/config.xml file to your FISHEYE_INST/ directory.

When the FISHEYE_INST environment variable is set, FishEye's directory layout becomes:

<table>
<thead>
<tr>
<th>Directory Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$FISHEYE_INST/config.xml</code></td>
<td>All permanent and temporary data is found under <code>$FISHEYE_INSTALL/var/</code></td>
</tr>
<tr>
<td><code>$FISHEYE_INST/var/</code></td>
<td>Site-specific Java libraries (.jars) that FishEye should load on startup. (Do not copy the dependent <code>/FISHEYE_HOME/lib/</code> files into here.)</td>
</tr>
<tr>
<td><code>$FISHEYE_INST/lib/</code></td>
<td>Site-specific syntax highlighting definitions.</td>
</tr>
<tr>
<td><code>$FISHEYE_INST/syntax/</code></td>
<td>Used for setting system properties within the FishEye JVM, which may sometimes be useful for support purposes. (Note: the other way to set properties is via FISHEYE_OPTS, e.g.: export FISHEYE_OPTS=-Dpropname=value)</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/lib/</code></td>
<td>FishEye's dependent libraries.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/syntax/</code></td>
<td>FishEye bundled highlighting definitions.</td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/bin/</code></td>
<td></td>
</tr>
<tr>
<td><code>/FISHEYE_HOME/ ...</code></td>
<td>Remaining files are found under <code>/FISHEYE_HOME/</code>.</td>
</tr>
</tbody>
</table>

The rest of this Installation Guide refers to `$FISHEYE_INST/`, but if you have not set FISHEYE_INST then it defaults to `/FISHEYE_HOME/` (the directory where you extracted FishEye).

**Next Step - Initial Configuration**
See the guidelines on configuring FishEye.

**Configuring FishEye**

*Initial Configuration*

FishEye runs its own HTTP web server, and additionally listens on a socket for administration/shutdown commands. These default to `:8060` and `127.0.0.1:8059` respectively. You can change both these addresses before starting FishEye by editing `config.xml`.

**Running FishEye for the First Time**

To run FishEye for the first time, simply do the following:

- **On Windows:**

  ```
  C:\> cd FISHEYE_HOME\bin
  C:\FISHEYE_HOME\bin> run.bat
  ```

  If you want to run FishEye as a Windows service, please refer to this document

- **On Unix-based systems:**

  ```
  $ cd /FISHEYE_HOME/bin
  $ ./run.sh
  ```

Once started, FishEye will run its own HTTP web server, on port 8060 by default.

You can access FishEye immediately by going to `http://HOSTNAME:8060/` in a browser.

**Administration Password**

The first time you run FishEye, when you access the FishEye web server you will be asked for:

- An administrator password. This password controls access to the FishEye Administration pages.
- A license key. You can get a trial license here.

If you need to reset the administrator password, delete the `admin-hash` attribute in the `<config>` element. You will be prompted to enter an administrator password next time you start FishEye.

**Accessing the Administration Pages**

Once you have set up an administrator password (described above), you can access the Administration pages at `http://HOSTNAME:8060/admin/`.

One of your first steps will be to add a repository.

You will also want to read about the command-line options for controlling FishEye.
You can disable FishEye's Administration pages by setting admin-hash="" in the `<config>` element of `config.xml` before starting FishEye.

Want a hands-on tour of the best FishEye features? See the FishEye Evaluator Resources page.

**Best Practices for FishEye Configuration**

1. **Ensure your FishEye scan performance is as fast as possible.**

   Use the `file://` protocol for fastest indexing performance. Read more. (If you cannot install FishEye on the server where Subversion is running, use svnsync to mirror the repository onto the fisheye server).

2. **If your repository is really large, consider starting at a sensible revision**

   This affects Subversion and Perforce. Do this when defining your Perforce or SVN repository.

3. **Exclude directories if you don't need them.**

   For example, not everyone may need to access a developer's personal branch on the repository, so you can exclude it from the repository scan. You may also want to exclude large branches/tags that have been deleted (even though they are deleted in your repository, FishEye will still index them as they once existed). Do this by using ‘Allow’ and ‘Exclude’ Admin settings.

4. **Consider skipping Perforce Label processing if not important**

   - Perforce Labels can be slow to process, and thus cause FishEye to index slowly in certain environments.
   - Do this by defining the ‘Skip Labels’ Repository Detail.

5. **Split your repositories into logical components if you can (For example, by product or project).**

   - A logical structure will make it simpler to exclude certain branches when they become less relevant to work in progress. This can garner significant performance gains.
   - Avoid treating an SCM like a file system — don’t alter the structure or move items around without a significant reason for doing so. Make these changes sparingly and as infrequently as possible.
   - The more often you make major changes to the structure inside your SCM, the more scanning is required for FishEye to keep track of its status. This especially applies to Subversion, because of its concept of ‘cheap copies’. The result is that small changes can be essentially unmeasurable and cause a large amount of re-scanning.

6. **Decide on your Subversion tag and branch conventions**

   Decide what conventions you are going to adopt for your subversion repositories and then stick to them. It’s best to stick to one of the standard conventions recommended by Subversion.

7. **Exclude tags and/or branches that you delete and recreate often.**

   You may have a branch or tag that you delete and recreate often, for example a latest tag which holds the latest release. Fisheye will take a long time to index this tag/branch as it needs to index its entire history, which can be very large. It is recommended that you exclude this directory from being indexed. See the documentation on the ‘Excludes’ option.

8. **Avoid using the text $Log$ in your CVS commit messages.**

   This is because FishEye does not handle the `$Log` RCS expansion keyword correctly. Some diff results (and line
numbers in diffs) may appear incorrect in files where $Log is used.

9. Avoid using symbolic links to refer to your FISHEYE_HOME location.

See this existing Knowledge Base document for more information.

10. Configure your index threads & memory usage to an appropriate level.

See the page Tuning FishEye for instructions and the related memory guide, Fix out of Memory Errors.

11. Do not use a 64 bit JVM.

Refer to our System Requirements.

See Also

- Improve FishEye Scan Performance
- Tuning FishEye

FishEye Upgrade Guide

The first time you run a new version of FishEye, it will automatically upgrade its data. This may involve a complete re-index of your repository.

- Before you start
- Upgrade Procedure
  - Method 1: Using a Separate FISHEYE_INST Directory
  - Method 2: No Separate FISHEYE_INST Directory
  - Method 3 - Without a FISHEYE_INST Directory, but would like to set one up
- FishEye 2.0 Upgrade Notes
- FishEye 1.6 Upgrade Notes
- FishEye 1.5 Upgrade Notes
- FishEye 1.4 Upgrade Notes
- FishEye 1.3 Upgrade Notes

Before you start

- Before upgrading you should always read the Release Notes and Changelog for the version you are upgrading to, as well as any versions you are skipping.

Upgrade Procedure

Your upgrade procedure depends on whether you are using a separate FISHEYE_INST directory. Read more about FISHEYE_INST in the Installation Guide.

Method 1: Using a Separate FISHEYE_INST Directory

1. Shutdown your existing fisheye server
2. Make a backup of your FISHEYE_INST directory
3. Extract the new FishEye version to a directory, leaving your FISHEYE_INST environment variable set to its existing location.
4. Start FishEye from the new installation.
5. Follow any version-specific instructions found in the Release Notes.

Method 2: No Separate FISHEYE_INST Directory
You will need to copy some files from your old FishEye installation to your new one.

1. Extract the new FishEye instance into a directory such as `/NEW_FISHEYE/`.
2. Delete the `/NEW_FISHEYE/var` directory.
3. Shut down the old FishEye 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 FishEye license, copy your `fisheye.license` to `/NEW_FISHEYE/`. (Atlassian-issued licenses are included within `config.xml`.)
7. Start FishEye from the new installation.

**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. Extract the new FishEye archive into a directory such as `/NEW_FISHEYE/`.
6. Start FishEye from the new installation by running `NEW_FISHEYE/bin/run.sh`. (Use `run.bat` on Windows).
7. 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.

**FishEye 2.0 Upgrade Notes**

- Please note that upgrading from 1.6.x will require an (automatic) full reindex.
- Please note that, unless "Store Diff Info" is disabled, FishEye’s disk space requirements are much greater than in previous versions. On an existing FishEye instance, this will first become apparent when the instance is reindexed. Please also see the disk space recommendations on the System Requirements page.
- Supported browsers are: Safari 3+, FireFox 3+ and Internet Explorer 7+ (not IE6).

**FishEye 1.6 Upgrade Notes**

- To take advantage of the new diff contents search in Quick Search, you will need to re-index your repository.
- Crowd 1.3 users will need to upgrade to Crowd 1.4.4 or later due to an incompatibility with this version of FishEye.

**FishEye 1.5 Upgrade Notes**

- Per-Author Line Counts require the ‘Store Diff’ setting to be true, but this will be false for existing repositories. Existing repositories will require this setting to be changed to ‘true’, then a full re-scan of that repository must be done. This is not essential to continue using your FishEye instance — it is only required for per-author graphs on the FishEye chart page.
- Upgrading from 1.2.5 (or earlier) or 1.3beta8 (or earlier) will force a complete re-scan of CVS repositories.
- Upgrading to this version will force a complete re-scan of P4 repositories.
- Upgrading from from 1.1.x (or earlier) or 1.2beta2 (or earlier) will force a complete re-scan of SVN repositories.

**FishEye 1.4 Upgrade Notes**
• If you were using the old Crowd Authenticator plugin in FishEye 1.3.x, you need to switch to the in-built Crowd Authenticator in FishEye 1.4+. Please see the Crowd documentation for details.
• As of version 1.3, FishEye requires a JVM version 1.5 or later. Previously, 1.4+ was required.
• Upgrading from 1.2.5 (or earlier) or 1.3beta8 (or earlier) will force a complete re-index of CVS repositories.
• Upgrading to this version will force a complete re-index of P4 repositories.
• Upgrading from from 1.1.x (or earlier) or 1.2beta2 (or earlier) will force a complete re-index of SVN repositories.

FishEye 1.3 Upgrade Notes

• As of version 1.3, FishEye requires a JVM version 1.5 or later. Previously, 1.4+ was required.
• Upgrading from 1.2.5 (or earlier) or 1.3beta8 (or earlier) will force a complete re-index of CVS repositories.
• Upgrading to this version will force a complete re-index of P4 repositories.
• Upgrading from from 1.1.x (or earlier) or 1.2beta2 (or earlier) will force a complete re-index of SVN repositories.

FishEye User’s Guide

- Work with FishEye inside your IDE
  Use the Atlassian Connector for Eclipse or the Atlassian Connector for IntelliJ IDEA to work with FishEye and Crucible right there in your development environment.

- Fisheye Quick Start Guide
- Using the FishEye Screens
  • Browsing through a Repository
  • Viewing the Changelog
  • Viewing a File History
  • Searching the Repository
  • FishEye Charts
  • Using Favourites
  • Viewing People's Statistics
  • Viewing File Content
- Changing your User Profile
  • Re-setting your Password
- EyeQL Reference Guide
- Antglob Reference Guide
- Date Expressions Reference Guide

Fisheye Quick Start Guide

This guide will explain how to get FishEye installed and running as easily as possible. For more advanced installation topics, see the Installation Guide.

Step 1. Install FishEye

1. Download the FishEye zip file and extract it. This document assumes you have extracted FishEye to /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.
3. If you intend to use FishEye with Subversion, please be sure to read about the requirements, Subversion client setup, and granting permission to FishEye to scan your repository.
4. If you intend to use FishEye with Perforce, please ensure you read about the requirements and Perforce client setup.

Step 2. Run FishEye

1. You can start FishEye immediately with the following:
   - For Unix-based systems:
     
     ```
     $ cd /FISHEYE_HOME/bin
     $ ./run.sh
     ```
   - For Windows-based systems:
     
     ```
     C:\> cd FISHEYE_HOME\bin
     C:\FISHEYE_HOME\bin> run.bat
     ```

2. Once started, FishEye will run its own HTTP web server on port 8060. You can access FishEye immediately by going to `http://HOSTNAME:8060/` in a browser.

   **Default ports**
   By default, FishEye 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 FishEye Administration pages or by editing `/FISHEYE_HOME/config.xml` and restarting FishEye.

   **Read-only access for FishEye**
   We recommend you run FishEye as a user that has only **read** access to your repository.

   An exception to this rule is users running the JIRA FishEye plugin with Perforce Job Integration. In that scenario, you must give FishEye write access.

Step 3. Set up FishEye

1. The first time you access FishEye from a browser, you will be asked to enter an administrator password. This password will give you access to the FishEye Administration pages.
2. You will also be prompted for a trial license, which you can find here.
3. Once you have set up an administrator password, you can access the Administration pages at `http://HOSTNAME:8060/admin/`.
4. One of the first steps will be to add a repository.

Step 4. Use FishEye

1. Once you have added a repository, you can view it in FishEye at `http://HOSTNAME:8060/`.
2. 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.
Stopping FishEye

To stop the FishEye server:

- For Unix-based systems:

  ```bash
  $ cd /FISHEYE_HOME/bin
  $ ./stop.sh
  ```

- For Windows-based systems:

  ```bash
  C:\> cd FISHEYE_HOME\bin
  C:\FISHEYE_HOME\bin> stop.bat
  ```

⚠️ Want a hands-on tour of the best FishEye features? See the FishEye 101 page.

Using the FishEye Screens

The sections below describe the different screens in FishEye and the information you can retrieve from them. Each page (tab) has a number of panes, and each pane is described separately.

Header

The header along the top of the FishEye screen will remain the same as you browse through the different screens. You can

- Click the 'Dashboard' tab to see your personal code commits; your reviews (if you are using Crucible); and your issues (if you are using JIRA).
- Click the 'Source' tab to see the following sub-tabs:
  - 'Repositories' — the list of all FishEye repositories. Click a repository name to browse the repository. A number of sub-tabs are then available as described below (see 'Repository Sub-Tabs').
  - 'Activity' — sub-tabs allow you to see the following across all repositories, for all users: code commits; reviews (if you are using Crucible); and issues (if you are using JIRA).
- **If you are using Crucible**: Click the 'Projects' tab to see a list of all projects (see the Crucible documentation).
- Click the 'People' tab to view statistics about committers to your FishEye repositories (see Viewing People's Statistics (draft)).
- **If you are using Crucible**: Click the 'Reviews' tab to go to your code reviews (see the Crucible documentation).
- Click your name to change your user settings (see Changing your User Profile).

Repository Sub-Tabs

Once you have selected a repository, you can navigate through it by selecting files and folders on the tree in the left navigation bar. When you reach a source file, a summary page is shown, displaying recent revisions to the file.

The horizontal sub-tabs above the file provide different views into the repository:
<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. See Viewing a File History.</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 (see Viewing the Changelog):</td>
</tr>
<tr>
<td></td>
<td>• All Activity — The default view, showing commits, reviews¹ and JIRA issues².</td>
</tr>
<tr>
<td></td>
<td>• Commits — Shows commits in the activity stream.</td>
</tr>
<tr>
<td></td>
<td>• Reviews¹ — 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></td>
<td>¹ If you are using Crucible</td>
</tr>
<tr>
<td></td>
<td>² If you are using JIRA</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 (see FishEye Charts).</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>

**Screenshot: The Repositories View**

**Screenshot: The Activity View**
Screenshot: Repository Sub-Tabs
Browsing through a Repository

Once you have selected a repository, you can drill down into a subdirectory using the directory tree in the left-hand column.

You can use the ‘Info’ icon at the extreme left of the screen to view the following repository details:

- **‘Statistics’** — Total number of committers; last commit; commits this week; total Lines of Code (Loc); change in LoC this week.
- **‘Line History’** — This graph shows the total line-count of MAIN or trunk over time for this directory subtree. This line-count does not include binary files, but does include every other file. If you have a branch-constraint specified, then the line-count history of that branch is also shown.
- **‘Commit Activity’**
- **‘Most Active Committers’** — Over the previous 90 days
- **‘Subversion Details’**

(The ‘Folder’ icon will toggle you back to the directory tree.)

At the right of the directory tree, the ‘Tools’ menu allows you to:

- ‘Watch’ (ie. receive email notifications about changes to) the current repository/branch/folder.
• Subscribe to an RSS feed of changes to the current repository/branch/folder/file.

**Screenshot: Browsing a Repository - Folder View**

**Screenshot: Browsing a Repository - Statistics View**

**Viewing the Changelog**
The ‘Activity’ sub-tab in FishEye 2.0 has similar functionality to the ‘Changelog’ in previous versions of FishEye.

When browsing a repository, the ‘Activity’ sub-tab in the right-hand column displays the changes made to your repository/branch/directory/file (ordered with the most recent first). The ‘Activity’ sub-tab provides the following options:

- **All Activity** — The default view, showing commits, reviews¹ and JIRA issues².
- **Commits** — Shows commits in the activity stream.
- **Reviews¹** — Shows review activity in the activity stream.
- **Scroll to Changeset** — Opens the changeset ID specified in the text field (press Enter to carry out the action).
- **Filter (see below)** — Applies constraints to the current activity stream.
- **Show Revisions** — If this is selected, then changeset items are automatically expanded to show modified files.
- **Earlier Activity** (Left Arrow icon) — Loads a page of earlier activity.
- **Later Activity** (Right Arrow icon) — Loads a page of later activity.
  ¹ If you are using Crucible
  ² If you are using JIRA

**Screenshot: Viewing the ‘Activity’ sub-tab**

### Filter mode

You can specify a ‘Filter’ to control the information that is shown in the ‘Activity’ sub-tab:

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch</td>
<td>Shows only changesets on that branch.</td>
</tr>
<tr>
<td>Committer</td>
<td>Shows only changesets checked in by the given committer/author.</td>
</tr>
<tr>
<td>File Name</td>
<td>Shows only changesets that contain a given file.</td>
</tr>
<tr>
<td>Log Comment</td>
<td>Shows only changesets where the commit comment matches the given text.</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>Start Date</td>
<td>Shows only changesets created on or after that date. Must be of the form YYYY-MM-DD, YYYY-MM or YYYY (you can use '/' instead of '-').</td>
</tr>
<tr>
<td>End Date</td>
<td>Shows only changesets created on or before that date. Must be of the form YYYY-MM-DD, YYYY-MM or YYYY (you can use '/' instead of '-').</td>
</tr>
</tbody>
</table>

**Screenshot: Using the Filter**

**Viewing a File History**

When browsing a repository, the 'Revisions' sub-tab in the right-hand column displays the different revisions of a file.

**Arbitrary Diffs**

The 'Diff ... and ...' boxes allow you to request a diff between any two revisions of the selected file. You can use revision numbers or tag names.

**Screenshot: Revisions**
Searching the Repository

FishEye allows you a number of ways to search through the repository to find particular changesets or files. Methods of searching are described below:

- Quick Nav
- Quick Search
- Simple Search
- Advanced Search

Quick Nav

Quick Nav results appear as soon as you start typing into the 'Quick Search' box in FishEye, before you've pressed
Enter or activated the search. You can click on the Quick Nav result items to visit each one. Try typing a CSID into this box to go directly to that page.

*Screenshot: Quick Nav in FishEye*

**Quick Search**

To use this search, enter your search term in the 'Quick Search' box in the top right hand corner of the FishEye screens.

*Screenshot: Quick Search box*

You can search for the following:

- Authors
- Branch names
- Commit comments
- Changeset ids
- Filenames/paths
- File content
- Tags
- Date (YYYY-MM-DD format, or any substring of that)
Added/Removed diff text. ⚠ Requires a re-index of the repository.h

Results are sorted by relevance, with limited results per page. Click 'Next' to load following pages. Search matches inside the results are highlighted in yellow.

Restricting searches by prefixing database field
You can search matches against a given field, by using a search in this format:

```
author:anna
```

This would return all results from the author field that match the string 'anna'.

Searches can be specifically restricted to the following available fields:

- Author
- Comment
- Contents
- File (You can use Antglobs)
- Branch
- Tag
- Cs
- Date
- AddedLine (searches diff text)
- RemovedLine (searches diff text).

Searching for discrete strings with precise case
To search for a specific string that appears discretely, with specific capital or non-capital letters, search with quotation marks, as in the following example:

```
"Monkey"
```

This search will ignore occurrences of the string that appear embedded in other strings, have different case, or are part of a path or disk location. The example above would not return "ProjectMonkey", "monkey", or "/zoo/mammals/monkey/archie/".

⚠ Note that regular quick searches do not take case into account and phrases cannot be searched in Quick Search at present.
File content Quick Search and Diff Text searching are new features in FishEye 1.6 and will continue to be improved. If you want access to full diff text and historical file contents in the repository, you will need to re-index it.

A note about searching multiple repositories:
Cross-repository searching has a 5-second limitation on searches, to prevent it from becoming unresponsive and consuming server resources on FishEye instances that have large numbers of repositories. This means that cross-repository quicksearch is not an exhaustive search, and may not include all repositories in a large Fisheye instance. Preference is given to repositories with the most recent changes. For exhaustive searches, you should:

1. Limit your search to a particular repository, if possible.
2. Perform a full search, rather than a QuickSearch.

Simple Search

To access the simple search screen, click the ‘Query’ sub-tab when browsing a repository.

Screenshot: FishEye Simple Search panel
You can use this search to retrieve a list of changesets/files using the filters that are available. You can search using one or more of the following filters:

- Commit comments
- Contents of files — files must be non-binary, less than 5MB, and located on the trunk/head
- Added text/removed text
- File names/paths — Antglobs can be used
- Authors
- Branch names
- Tag names
- Revision dates before and after.

Results can be grouped by the following:

- Changeset
- Revision
You can choose to include any or all of the following fields in the results:

- Path
- Revision
- Author
- Date
- Comment
- Changeset
- Total lines
- Total lines added
- Total lines removed
- Tags
- Reviews *(if you are using Crucible).*

The results are shown in a tabular format. You can link to the search results, and you can save the results to a CSV file.

**Advanced Search**

In some circumstances the results of a simple search may not be specific enough. Using the advanced search, you can create your own complex searches using FishEye's powerful query language called EyeQL.

**Screenshot: FishEye Advanced Search**

To do an advanced search, click the *Switch to EyeQL Search* link found at the bottom of the Simple Search screen.
Use Simple Search to build your basic query first
You can flick between Simple and Advanced Search. The EyeQL statement will contain the basics of the statement and you can adapt it as required.

FishEye Charts

When browsing a repository, the 'Reports' sub-tab in the right-hand column displays graphical information about the lines of code (LOC) committed to the repository, over time. The following options are available:

- Charts
- Code Metrics

Charts

You can view chart information controlled by various criteria. Simply select the desired constraints and click the 'Apply' button.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Explanation</th>
<th>Values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch</td>
<td>Limits the chart to the selected branch, defined as the repository's trunk (unless the directory displayed is a branch directory).</td>
<td>Any branch from the current repository.</td>
<td>Displays the trunk, unless the current directory has been identified as a branch.</td>
</tr>
<tr>
<td>Author</td>
<td>Limits the chart to show specific author(s).</td>
<td>Any author configured in the system.</td>
<td>All</td>
</tr>
<tr>
<td>Extension</td>
<td>Limits the chart to show specific file type(s).</td>
<td>Any file extension; e.g. '.java'.</td>
<td>All</td>
</tr>
<tr>
<td>Chart type</td>
<td>Changes the chart's presentation.</td>
<td>Area, line, pie or change* chart.</td>
<td>Area</td>
</tr>
<tr>
<td>Show by</td>
<td>Secondary data by which to refine the chart.</td>
<td>Subdirectory, author or extension.</td>
<td>None</td>
</tr>
<tr>
<td>Start Date</td>
<td>Date of the earliest data to show.</td>
<td>Date in format YYYY-MM-DD.</td>
<td>None (show all)</td>
</tr>
<tr>
<td>End Date</td>
<td>Date of the latest data to show.</td>
<td>Date in format YYYY-MM-DD.</td>
<td>None (show all)</td>
</tr>
<tr>
<td>Y Axis</td>
<td>Choosing 'Tight' zooms in the charts view to the limits of the range that the data covers. Only applies to Line charts.</td>
<td>Full or Tight</td>
<td>Tight</td>
</tr>
<tr>
<td>Sub Directories</td>
<td>Limit the chart to a folder under the current branch. Files in the current directory are represented by an element labelled '.(this dir)'.</td>
<td>A single folder.</td>
<td>None (show all)</td>
</tr>
</tbody>
</table>
The 'change' chart displays the change in lines of code, for a specific date range, expressed as a line graph. For example, if the lines of code at the start date is 100, the start point will be zero and the rest of the graph shifted by 100 lines.

To return to the default chart settings, click the 'Clear' button.

Screenshot: FishEye custom chart settings

Screenshot: FishEye per-author LOC chart
**Screenshot: FishEye per-author LOC chart showing multiple authors**

**Screenshot: FishEye LOC chart by file extension**
Per-Author Lines of Code Statistics

You can view per-author statistics for lines of code as a chart. This allows you to see how many lines of code were contributed to your project by each author, over time. You can easily view this information on the charts page. Note, if you are upgrading from a previous version of FishEye, you will need to re-index the repository in order to show the per-author information.

Code Metrics

A number of built-in reports are also provided:

Screenshot: Commit Time/Volume

Screenshot: Top Committers
Using Favourites

This page contains instructions on using the 'Favourites' feature in FishEye to select, view and manage items of interest.

On this page:

- Favourites Overview
- Adding Items 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
- Viewing Your Favourite Items
- Renaming an Item In Your Favourites
- Removing an Item From Your Favourites

Favourites Overview

FishEye allows you to tag certain items as your favourites. You can select changesets, files, people and repositories to be added to your favourites. Once your favourites list is created, you can view it or see a stream of all activity relating to your favourite items. We suggest you select items that you are currently working on as favourites, to create a more relevant personalised view.

If you are using Crucible, you can also add code reviews to your favourites.

Adding Items to Your Favourites
To add an item to your favourites, follow one of the steps below.

**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.

**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, 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*

![Screenshot: Adding a Changeset to Your Favourites](image)

**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, 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.

**Adding a Repository to Your Favourites**

To add a repository to your favourites, click the 'Source' tab. 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*

![Screenshot: Adding a Repository to Your Favourites](image)

**Viewing Your Favourite Items**

To view your favourite items, click 'Dashboard' tab at the top left of the page and then the 'Favourites' second level tab, beneath that.

*Screenshot: Viewing Your Favourites*
Renaming an Item In Your Favourites

To rename an item in your favourites, open the Favourites drop-down menu (the gold star icon located at the top centre of the FishEye screen, next to your user menu). Select the option called ‘Manage favourites’. The Dashboard favourites page opens, showing all of your favourites in the system. To rename any item (changing its favourite display name only — not the name of item itself), simply click the yellow star to the left of its name. A small pop-up menu will appear, the ‘Update Favourites’ menu. Type the desired name into the ‘Name’ field and click the ‘Save label’ button. The label will be updated for the favourites view.

Screenshot: Renaming an Item in Your Favourites

Removing an Item From Your Favourites

To remove an item from your favourites, open the Favourites drop-down menu (the gold star icon located at the top centre of the FishEye screen, next to your user menu). Select the option called ‘Manage favourites’. The Dashboard favourites page opens, showing all of your favourites in the system. To remove any item, simply click the
yellow star to the left of its name. A small pop-up menu will appear, the 'Update Favourites' menu. Click the 'Remove' button. The star will turn grey, showing that it has been removed from your favourites.

Screenshot: Removing an Item From Your Favourites

Viewing People's Statistics

To see charts and activity of everyone who commits code to your FishEye repositories, click the People tab at the top of the page. The list of all People appears.

Screenshot: List of all People in FishEye

The list shown is comprised of all users that have accounts on the system. 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.

You can click on a person's name to see detailed information about their additions to the repository, and issue updates and reviews if using FishEye with Crucible and JIRA integration is set up.

Screenshot: Statistics on a Person in FishEye
Some users may not appear to have the correct 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 this directory will be considered trunk.

Avatars

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 FishEye supports. See the page on Changing your User Profile.

If you are using Crucible, statistics on each person's code reviews are also available.

Viewing File Content

When browsing a repository, clicking the 'Source' sub-tab in the right-hand column will display the contents of a file.

Screenshot: Viewing File Content in FishEye
Arbitrary Diffs

The numbered drop-down options in the toolbar allow you to request a diff between any two revisions of the selected file. You can use revision numbers or tag names.

Annotation Highlighting

Annotation highlighting can be configured via a drop-down menu button on the toolbar. You can choose from one of the following three options:

- Age,
- Author,
- None.

Columns

You can configure the amount of information displayed on the page's columns. You can toggle the following items on and off:

- Author(s),
- Revision(s),
- Line Numbers.

Changing your User Profile

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

To change these settings, log into FishEye and click the User Menu (labelled with your username) at the top of the screen, then select 'Settings'.

Below is a description of each tab and its contents.

- Display Settings Tab
- Profile and Email Tab
- Change Password Tab
- Author Mapping Tab
- Watches Tab
- Reviews tab
Customising Your User Avatar

Screenshot: Settings

Display Settings Tab

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

General

<table>
<thead>
<tr>
<th>Option</th>
<th>Default Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length of tag list</td>
<td>'Medium'</td>
<td>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>Show Linecount History Graph</td>
<td>'Yes'</td>
<td>This is the graph that appears on the left hand side of the Browse and Changelog screen.</td>
</tr>
<tr>
<td>Show hidden directories</td>
<td>'No'</td>
<td>Do not show the hidden directories within any folder lists.</td>
</tr>
<tr>
<td>Show empty directories</td>
<td>'Yes'</td>
<td>The option to see any empty directories within any folder lists.</td>
</tr>
</tbody>
</table>
**File History View Mode**

Default is 'Logical'.

In Subversion repositories, FishEye 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', FishEye 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.

**Timezone**

Default is the timezone of the FishEye server.

---

**Changelog**

<table>
<thead>
<tr>
<th>Changesets per page</th>
<th>The default is 30 per page.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum files shown in a changeset</td>
<td>Default is 5.</td>
</tr>
</tbody>
</table>

---

**Diff View**

<table>
<thead>
<tr>
<th>Truncate long diffs</th>
<th>Default is 'Yes'. Only show part of the diff, if the diff contains many lines of code.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diff mode</td>
<td>Default is 'Unified'. Can be changed to 'Side-by-side' diffs.</td>
</tr>
<tr>
<td>Line wrapping</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>Highlighting Colours</td>
<td>The default colour scheme uses bright colours for highlighting diffs in the code. If you prefer more muted colours, select <strong>Classic (muted)</strong>.</td>
</tr>
</tbody>
</table>

---

**Source View**

<table>
<thead>
<tr>
<th>Default annotation mode</th>
<th>Default is 'Age'. It can be changed to 'Author' or 'None'.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab width</td>
<td>Default is 8. Can be changed to a number between 1 and 10.</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>Display Name</th>
<th>Name displayed for the user currently logged in.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email Address</td>
<td>The address all email notifications will be sent to.</td>
</tr>
<tr>
<td>Email Format</td>
<td>Default is text. Can be changed to be sent as HTML.</td>
</tr>
<tr>
<td>Send Watch Emails</td>
<td>Default is 'Immediately'. Can be changed to 'Daily'.</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**

This functionality is used by Crucible. Refer to the Crucible documentation.
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
This functionality is used by Crucible. Refer to the Crucible documentation.

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.

Re-setting your Password
If you need to reset your password, FishEye has an integrated mechanism to generate a new password and send it to the email address in your profile.

To reset your password:

1. On the log in screen, click the 'Forgot your password?' link. The 'Request New Password' screen opens.
2. Fill out your username or email address and the Captcha step. That is, click in the form field labelled 'Please enter the word as shown below' and type the graphical letters shown above the 'Submit' button.
3. An email is then sent to the email address specified in your profile. When it arrives, click the link supplied to complete the password reset.
4. On the resulting web page, you will receive the message 'A new password has been sent to your account.'
5. An email will arrive in your inbox, containing your new password.

If you receive a password-reset email that you did not request, simply disregard it to continue using your current password.

Screenshot: The Log In dialog
FishEye contains a powerful query language called **EyeQL**. EyeQL is an intuitive SQL-like language that allows you to write your own specific queries. See examples.

EyeQL allows you to perform complex searches either within the Advanced Search or incorporated in scripts from the FishEye API.

query:

```plaintext
select revisions (from (dir|directory) word)? (where clauses)? (order by date (asc | desc))? 
Notes: asc produces 'ascending order'. desc produces 'descending order'. (group by (file|dir|directory|changeset))? (return return-clauses)? (limit limit-args)?

clauses:

clause (|(or|and|) clause)*
```
Notes:
* and binds more tightly than or.
'.' (comma) means 'and'.

clause:
(clauses)

not clause

path (not)? like word
Notes:
word is an Antglob.

path = word
Notes:
Defines an exact path without wildcards or variables. path must represent a complete (hard-coded) path.

path ! = word
Notes:
Defines an exact path exclusion without wildcards or variables. path must represent a complete (hard-coded) path.

date in ( ( [ ] dateExp, dateExp ( ) | ] )
Notes: The edges are
  inclusive if [ or ] is used.
  exclusive if ( or ) is used.

date dateop dateExp
Notes:
dateop can be <, >, <=, >=, =, == or !=.

author = word

author in (word-list)

comment matches word
Notes:
Does a full-text search.

comment = string
Notes:
Matches string exactly.
Most comments end in a new line, so remember to add \n at the end of your string.

comment =~ string
Notes:
string is a regular expression.

content matches word
Notes:
Does a full-text search.
At this time searches are restricted to HEAD revisions.

(modified|added|deleted)? on branch word
Notes:
Selects all revisions on a branch.
modified excludes the branch-point of a branch.
added selects all revisions on the branch if any revision was added on the branch.
deleted selects all revisions on the branch if any revision was deleted on the branch.
**tagged** `op? word` Notes:
`op` can be `<`, `>`, `<=`, `>=`, `=`, `==` or `!=`.
`op` defaults to `==` if omitted.
These operators are 'positional' and select revisions that appear on, after, and/or before the given tag.

**between tags** `tag-range`

**after tag** `word`

**before tag** `word`

**is head (on word)?**
Notes:
This selects the top-most revision on any branch, if no branch is specified.

**is ( dead | deleted )**
Notes:
Means the revision was removed/deleted.

**is added**
Notes:
Means the revision was added (or re-added).

**csid = word** Notes:
Selects all revisions for the given changeset ID.

**p4:jobid = word** Notes: finds revisions whose Perforce jobid is word.

**p4:jobid =~ word** Notes: finds revisions whose Perforce jobid matches regex word.

**reviewed**
Notes: *(applies to Crucible reviews)* alias for in or before any closed review.

**(in | before | in or before) review word (in | before | in or before) any (review states)? review**
Notes:
`word` is a review key.
in selects reviewed revisions. If a review contains a diff, then only the most recent revision is considered in the review.
before selects all revisions in a file prior to the revision in the review.
review states is a comma-separated list of open, closed, draft.

**tag-range:**

```
(( | [ ] T1:word, T2:word ( ) | [ ] )
```
Notes:
A range of revisions between those tagged T1 and T2.
The edges are:
inclusive if `[ ]` is used.
exclusive if `( )` is used.
You can mix edge types. These are all valid: `(T1,T2), [T1,T2], (T1,T2) and [T1,T2).

**Having trouble with Subversion tags?** See How Tags Work in Subversion for more information.

**word:**

Any *string*, or any non-quoted word that does not contain white space or any other separators.
string:
A sequence enclosed in either " (double quotes) or ' (single quotes).
The following escapes work: \\ " \n \r \t \b \f.
Unicode characters can be escaped with \uXXXX.
You can also specify strings in 'raw' mode like r"foo". (Similar to Python's raw strings. See Python's own documentation).

dateExp:
See our Date Expressions Reference Guide for more information on date formats.

return-clauses:
return-clause (, return-clause)*
A return clause signifies that you want control over what data is returned/displayed.

return-clause:
( path | revision | author | date | comment | csid | isBinary | totalLines | linesAdded | linesRemoved | isAdded
| isDeleted | isCopied | isMoved | tags | reviews)
( as word )?
The attribute to return, optionally followed by a name to use for the column.

Notes: reviews applies to Crucible reviews.

limit-clause:
( duration | offset, duration | duration offset offset )
Notes: Limits the number of results to return. offset specifies the starting point of the truncated result set and duration specifies the set length. offset is zero-based.

Examples
The following examples demonstrate using EyeQL to extract information from your repository.

Find files removed on the Ant 1.5 branch:
select revisions where modified on branch ANT_15_BRANCH and is dead group by
changeset

As above, but just return the person and time the files were deleted:
select revisions where modified on branch ANT_15_BRANCH and is dead return path,
author, date

Find files on branch and exclude delete files:
select revisions where modified on branch ANT_15_BRANCH and not is deleted group by
changeset

Find changes made to Ant 1.5.x after 1.5FINAL:
select revisions where on branch ANT_15_BRANCH and after tag ANT_MAIN_15FINAL group
by changeset

Find changes made between Ant 1.5 and 1.5.1:
select revisions where between tags (ANT_MAIN_15FINAL, ANT_151_FINAL] group by
changeset

As above, but show the history of each file separately :
select revisions where between tags (ANT_MAIN_15FINAL, ANT_151_FINAL] group by file
Find Java files that are tagged ANT_151_FINAL and are head on the ANT_15_BRANCH: (i.e. files that haven't changed in 1.5.x since 1.5.1)
select revisions from dir /src/main where is head and tagged ANT_151_FINAL and on branch ANT_15_BRANCH and path like *.java group by changeset

Find changes made by Conor to Ant 1.5.x since 1.5.0
select revisions where between tags (ANT_MAIN_15FINAL, ANT_154] and author = conor group by changeset

Find commits that do not have comments:
select revisions from dir / where comment = "" group by changeset

Find the 10 most recent revisions:
select revisions order by date desc limit 10

Find the 5th, 6th & 7th revisions:
select revisions order by date limit 4, 3

Find commits between two dates:
select revisions where date in [2008-03-08, 2008-04-08]

Antglob Reference Guide

FishEye supports a powerful type of regular expression for matching files and directories (same as the pattern matching in Apache Ant).

These expressions use the following wildcards:

<table>
<thead>
<tr>
<th>Wildcard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>Matches one character (any character except path separators)</td>
</tr>
<tr>
<td>*</td>
<td>Matches zero or more characters (not including path separators)</td>
</tr>
<tr>
<td>**</td>
<td>Matches zero or more path segments.</td>
</tr>
</tbody>
</table>

Remember that Antglobs match paths, not just simple filenames.

- If the pattern does not start with a path separator i.e. / or \, then the pattern is considered to start with /**/. |
- If the pattern ends with / then ** is automatically appended. |
- A pattern can contain any number of wildcards.

Also see the Ant documentation.

Examples

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.txt</td>
<td>/foo.txt and /bar/foo.txt but not /foo.txty or /bar/foo.txty/</td>
</tr>
<tr>
<td>/*.txt</td>
<td>/foo.txt but not /bar/foo.txt</td>
</tr>
<tr>
<td>/**dir1/file.txt</td>
<td>Same as above.</td>
</tr>
</tbody>
</table>
Date Expressions Reference Guide

FishEye supports a wide variety of date expressions. A date has the two possible general forms:

- DATE[+-]TIMEZONE[+-]DURATION, or
- DATECONSTANT[+-]DURATION.

The TIMEZONE and DURATION parts are both optional.

TIMEZONE can be an offset from GMT HHMM or HH:MM, or simply the letter Z to denote GMT. If no timezone is given, the FishEye server's configured timezone is used.

DATE can be either of the following:

<table>
<thead>
<tr>
<th>YYYY-MM-DDThh:mm:ss</th>
<th>Specifies a time and date (separated by a T). The seconds part may contain a fractional component. A / can be used instead of - as a separator.</th>
</tr>
</thead>
<tbody>
<tr>
<td>YYYY-MM-DD</td>
<td>Specifies 00:00:00 on the given date. A / can be used instead of - as a separator.</td>
</tr>
</tbody>
</table>

DATECONSTANT can be any of:

<table>
<thead>
<tr>
<th>now</th>
<th>This very instant (at the time the expression was evaluated).</th>
</tr>
</thead>
<tbody>
<tr>
<td>today</td>
<td>The instant at 00:00:00 today. (server-time* or GMT)</td>
</tr>
<tr>
<td>todaygmt</td>
<td>The instant at 00:00:00 today. (server-time* or GMT)</td>
</tr>
<tr>
<td>thisweek</td>
<td>The instant at 00:00:00 on the first day of this week. Sunday is considered the first day. (server-time* or GMT)</td>
</tr>
<tr>
<td>thisweekgmt</td>
<td>The instant at 00:00:00 on the first day of this week. Sunday is considered the first day. (server-time* or GMT)</td>
</tr>
<tr>
<td>thismonth</td>
<td>The instant at 00:00:00 on the first day of this month. (server-time* or GMT)</td>
</tr>
<tr>
<td>thismonthgmt</td>
<td>The instant at 00:00:00 on the first day of this month. (server-time* or GMT)</td>
</tr>
<tr>
<td>thisyear</td>
<td>The instant at 00:00:00 on the first day of this year. (server-time* or GMT)</td>
</tr>
<tr>
<td>thisyeargmt</td>
<td>The instant at 00:00:00 on the first day of this year. (server-time* or GMT)</td>
</tr>
</tbody>
</table>

* The timezone used for server-time is part of the FishEye configuration

The syntax for DURATION is similar to the XML Schema duration type. It has the general form PnYnMnDTnHnMnS. See Section 3.2.6 of the XML Schema Datatypes document for more details.

Examples

<table>
<thead>
<tr>
<th>2005-01-02</th>
<th>The start of the day on January 1, 2005 (server's timezone)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005-01-02-0500</td>
<td>The start of the day on January 1, 2005 at GMT offset -0500 (New York)</td>
</tr>
<tr>
<td>2005-01-02T12:00:00Z</td>
<td>Midday, January 1, 2005 GMT</td>
</tr>
<tr>
<td>today-P1D</td>
<td>Yesterday (start of day)</td>
</tr>
</tbody>
</table>
FishEye 2.1 Documentation

<table>
<thead>
<tr>
<th>today+P1D</th>
<th>Start of tomorrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>thismonth-P1M</td>
<td>Start of last month</td>
</tr>
<tr>
<td>thisyear+P1Y</td>
<td>Start of next year</td>
</tr>
<tr>
<td>now-PT1H</td>
<td>One hour ago</td>
</tr>
<tr>
<td>now+PT1H2M3S</td>
<td>One hour, two minutes and three seconds from now</td>
</tr>
</tbody>
</table>

**FishEye Administrator's Guide**

Once you have installed and configured FishEye, you can access the Administration pages at http://HOSTNAME:8060/admin/.

The FishEye 'Admin Menu' allows you to administer your FishEye instance and manage your repositories, as shown in the screenshot below.

You will also want to read about the command-line options for controlling FishEye.

You can disable FishEye's Administration pages by setting `admin-hash=""` in the `<config>` element of `config.xml` before starting FishEye.

**Screenshot: FishEye Admin Menu and Repository List**

![Screenshot: FishEye Admin Menu and Repository List](image)

Information in this Administrator's Guide:

- Managing your Repositories
- Setting up a Repository Client
- Configuring ViewVC Compatibility
- Setting up your Web Server
- Configuring SMTP
- Setting up your Users and Security
- Backing Up and Restoring FishEye Data
- Advanced Administration Options
Managing your Repositories

You can see a summary of your current repositories at the top of the first screen of the FishEye 'Administration' options.

Screenshot: The Repository List in the FishEye Administration Screen

See the links below for documentation on the FishEye repository options.

FishEye Repository Options

- Adding a Repository
  - ClearCase
  - CVS
  - Git
  - Perforce
  - Subversion
    - SVN fisheye.access
    - SVN Tag & Branch Structure
    - How Tags Work in Subversion

- Repository Options
  - Configuring Repository Details
  - Store Diff Info
  - Operations
  - Symbolic
  - Indexer
  - Updater
  - Linkers
  - Permissions
  - Watches
  - Allow (Process)
  - Hidden Directories
  - Tarball Settings
  - Properties
• Commit Message Syntax

Adding a Repository

Adding a repository to FishEye is a simple matter. Further configuration options are available once a repository has been added, depending upon the repository type.

Note that FishEye needs to build an index and cache of your repository. This begins when you first enable a repository, and may take some time to complete.

To add a repository,

1. From the 'Admin Menu', click the 'New' link next to 'Repository List', OR, click through to the 'Repository List' page and then click 'Add Repository'.
2. Select a 'Repository type' from the dropdown list.
3. Specific fields will appear on the 'Add Repository' screen, depending on the chosen repository type. Enter the repository details as prompted. You will find more information in the specific sections listed below.

FishEye currently supports the following repository types:

• ClearCase
• CVS
• Git
• Perforce
• Subversion

Screenshot: Adding a CVS Repository

Add Repository

| Name:          | [Input Field] |
| Description:   | [Input Field] |
| Repository type: | Concurrent Versions System (CVS) |
| CVS dir:       | [Input Field] |
| Charset:       | [Input Field] |
| Enable immediately: | Yes/No |

ClearCase

⚠️ The information on this page relates to FishEye's Early Access Program support for IBM ClearCase. Do not use in production.

This page contains instructions for how to set up a ClearCase repository in FishEye, a configuration reference and a list of known issues.
If you also have Crucible, once configured you will be able to run Crucible reviews on code from your ClearCase repository.

On this page:

- Requirements
- Setting up a ClearCase Repository
- ClearCase Repository Details
- Inclusion/Exclusion Settings
- View Creation
- Indexing Logic
  - UCM ClearCase
  - Base ClearCase
- Allocating Time for Repository Scanning
- Changes from previous Alpha releases
- Known Issues
- Feedback and Support

Requirements

The instructions on this page require the following applications:

- IBM ClearCase 2003.06.10 or later
- FishEye 2.1.0 Milestone 2 (click to download) or Crucible 2.1.0 Milestone 2 (click to download)

Setting up a ClearCase Repository

When adding or managing a ClearCase repository, carry out the following steps:
1. Open FishEye's 'Add Repository' dialog, by choosing 'Administration' > 'Repository List' > 'New'.
2. Set your repository details, as described below.

ClearCase Repository Details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Allowed values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A name chosen by you to be displayed in the list of FishEye repositories.</td>
<td>Free text</td>
</tr>
<tr>
<td>Description</td>
<td>A description for this repository, if required.</td>
<td>Free text</td>
</tr>
<tr>
<td>Repository Type</td>
<td>Defines the repository type; these instructions apply to ClearCase, so select ClearCase.</td>
<td></td>
</tr>
<tr>
<td>Enable Immediately</td>
<td>Defines whether the repository will be accessible in FishEye right away.</td>
<td>Yes / No</td>
</tr>
<tr>
<td>UCM</td>
<td>Indicates whether the underlying ClearCase repository uses UCM or Base ClearCase.</td>
<td>Yes / No</td>
</tr>
<tr>
<td>Integration Streams Only</td>
<td>Specifies whether FishEye should index content that has been delivered to development and streams or only integration streams. It is recommended that users choose 'Yes' for this option.</td>
<td>Yes / No</td>
</tr>
<tr>
<td>View Location</td>
<td>The location of a directory accessible to the FishEye instance where views can be created.</td>
<td>A system path</td>
</tr>
<tr>
<td>View Storage Location</td>
<td>The location where view storage files are stored.</td>
<td>A system path</td>
</tr>
</tbody>
</table>
### VOB to Include
- A drop down list displaying all the non-UCM VOBs found in the ClearCase installation. If users only require that FishEye index a single VOB, they should select the VOB to index from this drop down list.  
  - Auto populated

### VOB Includes
- Specifies the pattern that should be used to determine whether a VOB should be included in the indexing logic. Multiple inclusion patterns can be separated with a comma.  
  - Free text

### VOB Excludes
- Specifies the pattern that should be used to determine whether a VOB should be excluded from the indexing logic. Multiple exclusion patterns can be separated with a comma.  
  - Free text

### UCM Project To Include
- A drop down list displaying all the UCM Projects found in the ClearCase installation. If users only require that FishEye index a single UCM Project, they should select the Project to index from this drop down list.  
  - Auto populated

### UCM Project Include Patterns
- Specifies the pattern that should be used to determine whether a UCM Project should be included in the indexing logic. Multiple inclusion patterns can be separated with a comma.  
  - Free text

### UCM Project Exclude Patterns
- Specifies the pattern that should be used to determine whether a UCM Project should be excluded from the indexing logic. Multiple exclusion patterns can be separated with a comma.  
  - Free text

*Screenshot: Adding a ClearCase Repository*
Inclusion/Exclusion Settings

The following points provide guidelines for the settings which may need to be applied in order to restrict the number of ClearCase Projects/VOBs indexed by FishEye.

- If you want all the VOBs/UCM Projects within your environment to be indexed, then you don't need to add any additional information on the Edit Repository screen. This is the default behaviour.
- If you want several VOBs/UCM Projects to be included (but not all), then you should include appropriate details in the VOB Includes/Excludes fields.
- If you only wish for a single VOB/UCM Project to be indexed, then you should select the specific VOB/UCM Project from the 'VOB to Include' or 'UCM Project to Include' drop down list. This will force FishEye to only index the selected VOB/UCM Project.

View Creation

As part of the repository scanning logic, FishEye will create a view for each Project (for ClearCase UCM environments) or VOB (for Base ClearCase) using the locations defined in the 'View Location' and 'View Storage Location' fields. This is required in order for the underlying 'cleartool' commands to be executed in the correct context. Please note that FishEye will not perform updates on these views - it is intended that these views will remain unpopulated.

Indexing Logic
It may be helpful to understand how FishEye's ClearCase support carries out indexing.

**UCM ClearCase**

The ClearCase support will attempt to index all the available content within a ClearCase environment. The logic works as follows (ClearCase specific terms are underlined see definitions):

- All PVOBs that are available are identified.
- For each PVOB, find all the Projects contained within the PVOB.
- For each Project, find all the Activities that have been delivered to the project.
- Find the Versions that were included in each Activity and index the Version information.
- Any Labels attached to Versions are also indexed.

* PVOB stands for *Project Versioned Object Base.*

**Base ClearCase**

The logic for the Base ClearCase support is similar to the UCM ClearCase support,

- All non-UCM VOBs that are available are identified.
- Find the check-ins for each VOB and index the version information.
- Any Labels attached to Versions are also indexed.

**Allocating Time for Repository Scanning**

The initial scan of a repository is a time and resource intensive operation, more so if the ClearCase repository being indexed is large (both in terms of the number of ClearCase projects and the number of change sets included in each project). In the Atlassian test environment (running in a virtual machine), each commit included in a change set would take approximately one second to complete (the time taken in a non-VM environment seems to be slightly faster at approx 700ms). You can use these numbers to estimate the time it will take to scan your repository; it could take many hours or possible days to complete.

**Changes from previous Alpha releases**

In the first Alpha release, the include/exclude rules for VOBs and Projects were handled by the 'Include/Exclude' rules item on the administration page. Based on feedback received during Alpha testing, this has been updated to provide additional flexibility:

- The VOBs which are indexed can be controlled via the 'VOB to Include' and 'VOB Include/Exclude Patterns' fields.
- Similarly, the UCM Projects which are indexed can be controlled via the 'UCM Project to Include' and 'UCM Project Include/Exclude Patterns' fields.
- The Include/Exclude rules on the Administration page now apply to files/directories that are indexed within a ClearCase VOB/Project. The values entered into these fields perform the matching logic as defined on the Allow (Process) page

**Known Issues**

There are a number of known issues with the ClearCase Alpha support in FishEye. These are listed below:

- Currently XML files cannot be viewed as 'Annotated' source. By default, ClearCase using a specific type manager to store XML files. This type manager does not support the 'cleartool annotate' command, which is used by the logic in FishEye that displays the Annotated source.

Further to this, by default ClearCase treats any files not defined in the 'default.magic' file as 'compressed' (for instance, property files are not included in the default.magic file). Only text-based type managers can be annotated (and hence, can be displayed via the 'Annotated Source' link). The type manager can be updated by performing the
following steps:

1. Update the default.magic file to include appropriate rules that specify the type manager to use for files of a given naming convention (this will take effect for newly created elements)
2. Modify the type manager for existing elements through the 'cleartool chtype' command.

Further information on the ClearCase type manager is available on the following pages:

Type Manager white paper
cleartool chtype command reference
cc.magic reference

- There is a known bug with earlier versions of ClearCase that limit the cleartool output to 64K of data. This may affect projects that contain a large amount of changes included in a changeset. This bug can be fixed by upgrading ClearCase — see this page for more information.

Feedback and Support

Please visit the FishEye forums to seek assistance with the alpha release of FishEye ClearCase support.

CVS

When adding or configuring a CVS (Concurrent Versions System) repository, you can:

- Define repository details, as described below.
- Set FishEye's repository options.

There are also the following CVS-specific actions:

- Updater.
- Indexer.

⚠️ Known Limitations

- To add a CVS repository, FishEye must have file system access to the repository. If you cannot install fisheye on the same server as CVS, then use rsync to mirror the repository.
- Currently, FishEye does not handle the $Log RCS expansion keyword correctly. Some diff results (and line numbers in diffs) may appear incorrect in files where $Log is used.

CVS Repository Details

<table>
<thead>
<tr>
<th>Name</th>
<th>A name for this repository. The name may contain alphanumeric, underscore, '-' or '.' characters. Use 'cvs' if you can't think of a better name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A short description of this repository.</td>
</tr>
<tr>
<td>CVS dir</td>
<td>The path to the CVS repository. This is often /usr/local/cvsroot. This is a path in the server's file system.</td>
</tr>
<tr>
<td>Charset</td>
<td>The character set used to interpret and display text files.</td>
</tr>
<tr>
<td>Enable immediately</td>
<td>Controls whether FishEye will immediately enable this repository, which starts the initial scan. If you wish to do some further configuration before the scan starts, then select 'No'. You can enable a repository later from the Repository List.</td>
</tr>
</tbody>
</table>
Screenshot: Adding a CVS Repository

Add Repository

Name: ____________________________
Description: ____________________________
Repository type: Concurrent Versions System (CVS)
CVS dir: ____________________________
Charset: default (ISO-8859-1)
Enable immediately: Yes No

Add Cancel

Git

⚠️ The information on this page relates to FishEye’s Early Access Program support for Git. Do not use in production.

When adding or managing a Git repository, you can do the following:

- Define repository details, as described below.
- Set FishEye’s repository options.

Git Repository Details

<table>
<thead>
<tr>
<th>Name</th>
<th>A name for this repository. The name may contain alphanumeric, underscore, ‘.’ or ‘.’ characters.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A short description of this repository.</td>
</tr>
<tr>
<td>Location</td>
<td>The URL describing the git repository location. FishEye will clone this repository for indexing purposes.</td>
</tr>
<tr>
<td>Path</td>
<td>The path within the Git repository that you wish to have FishEye index.</td>
</tr>
<tr>
<td>Block Size</td>
<td>Controls how many commits FishEye will process in one batch.</td>
</tr>
</tbody>
</table>

Screenshot: Adding a Git Repository
When adding or managing a Perforce repository, you can:

- Define repository details, as described below.
- Set FishEye's repository options.
- Set up a Perforce client.

### Perforce Repository Details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A name for this repository. The name may contain alphanumeric, underscore, '-' or '.' characters. Use 'perforce' if you can't think of a better name.</td>
</tr>
<tr>
<td>Description</td>
<td>A short description of this repository.</td>
</tr>
<tr>
<td>Perforce Host</td>
<td>The name of the server which provides the Perforce repository.</td>
</tr>
<tr>
<td>Port</td>
<td>The port the server is listening on. This field is optional. FishEye will default to the standard Perforce port (1666) if you do not specify a value here.</td>
</tr>
<tr>
<td>Path</td>
<td>The path within the Perforce depot that you wish to have FishEye index. You would normally put the depot path here, e.g. //depot/ but you may also use a more specific path to restrict FishEye to a subset of the depot.</td>
</tr>
<tr>
<td>Block Size</td>
<td>Controls how many changelists FishEye will fetch from the depot in one batch. Larger values can reduce the time it takes for FishEye to scan your repository for changes, but use more memory. The default is 400.</td>
</tr>
<tr>
<td>Filelog limit</td>
<td>FishEye uses the P4 filelog command to gather information about the files in changesets. The list of files generated can be very large. Setting a limit here will cause FishEye to batch up filelog operations into groups. This is useful with some versions of the Perforce client which may have trouble with large output. In general you should only set this field if you have a 2005 client or earlier. Lower values will degrade scanning performance.</td>
</tr>
<tr>
<td><strong>P4 Operation</strong></td>
<td><strong>Sets the timeout value that FishEye imposes on P4 operations. Operations which exceed this value are terminated. The default for most operations is 10 minutes.</strong></td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Timeout</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Throttle</strong></td>
<td><strong>If set, this allows FishEye to throttle how many connections it makes per second to the Perforce server. The default is blank (do not throttle). You may enter fractional values such as 2.5.</strong></td>
</tr>
<tr>
<td><strong>connections-per-sec</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Charset</strong></td>
<td><strong>The character set used to interpret and display text files.</strong></td>
</tr>
<tr>
<td><strong>Unicode Server</strong></td>
<td><strong>This field indicates whether the Perforce Server is running in internationalised mode.</strong></td>
</tr>
<tr>
<td><strong>Skip Labels</strong></td>
<td><strong>When true, FishEye will not scan Perforce Labels for FishEye tag information.</strong></td>
</tr>
<tr>
<td><strong>Case Sensitive</strong></td>
<td><strong>This field indicates whether the Perforce Server metadata is case sensitive. You should set this to 'false' for servers running on Windows platforms.</strong></td>
</tr>
<tr>
<td><strong>Disable Multiple Print</strong></td>
<td><strong>When FishEye needs file content from Perforce, it uses a p4 print operation. Normally FishEye will request multiple files in one operation but this can cause problems in some Perforce instances. Set this value to true to disable printing multiple files in one operation.</strong></td>
</tr>
<tr>
<td><strong>Start Revision</strong></td>
<td><strong>If you wish to set this, please enter a changelist number. If set, the revision number from which FishEye will start indexing the repository. The default is to start scanning from the first revision in the repository.</strong></td>
</tr>
<tr>
<td><strong>Initial Import</strong></td>
<td><strong>When a Start Revision is set, this setting controls how FishEye establishes the initial state of the repository. When true, FishEye will import the repository content as it existed one revision prior to the start revision. FishEye will create a single synthetic revision to hold the initial state. The comment associated with this revision will be 'Created by FishEye for initial repository import'. False means that FishEye will only process the revisions from the start revision onwards. The repository state prior to this revision is ignored.</strong></td>
</tr>
<tr>
<td><strong>Username/Password</strong></td>
<td><strong>The credentials to use if your repository requires authentication.</strong></td>
</tr>
<tr>
<td><strong>Store Diff Info</strong></td>
<td><strong>Enable this option if you are using the Subversion or Perforce SCM systems and making use of per-author line counts. Otherwise, enabling this option is not necessary. Read more information</strong></td>
</tr>
<tr>
<td><strong>Enable immediately</strong></td>
<td><strong>Controls whether FishEye will immediately enable this repository, which starts the initial scan. If you wish to do some further configuration before the scan starts, then select 'No'. You can enable a repository later from the Repository List.</strong></td>
</tr>
</tbody>
</table>

*Screenshot: Adding a Perforce Repository*
Subversion

When adding or managing a SVN (Subversion) repository, you can:

- Define repository details, as described below.
- Set FishEye's repository options.
- Set up a Subversion client.
- Grant permission to FishEye to scan your repository.
- Set up the correct branch and tag structure.

There are also the following SVN-specific actions:

- Updater.
- Indexer.
- Store Diff Info.
It is particularly important that you set up the correct branch and tag structure for your Subversion repositories. If FishEye does not know which files are tags and branches, it will treat all files as trunk files. This can significantly increase the effective size of your repository. This will increase initial scan time and impact runtime performance. Please refer to the instructions on tag and branch configuration.

In the majority of cases, indexing a small repository shouldn't take hours, and certainly not days. However, if you have a giant repository, have a slow remote host, you're using HTTP or HTTPS protocols, or if there is a problem with the symbolic setup of your repository, it could potentially take hours or even days. If in doubt, schedule the indexing to run over a weekend or extended maintenance period.

Using the 'file:///' protocol to access your Subversion repository can be much faster than the other network protocols. We recommend using the 'file:///' protocol if possible.

Knowledge Base
You may find some useful information in the Knowledge Base too.

<table>
<thead>
<tr>
<th>SVN Repository Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Detail</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Repository type</td>
</tr>
<tr>
<td>Store Diff Info</td>
</tr>
<tr>
<td>SVN URL</td>
</tr>
<tr>
<td>Path</td>
</tr>
<tr>
<td>Block Size</td>
</tr>
<tr>
<td>Svn Operation Timeout</td>
</tr>
</tbody>
</table>
Throttle connections-per-sec

If set, this allows FishEye to throttle how many connections it makes per second to the SVN server. Many systems use `inetd/xinetd` to service the `svnserve` protocol. `xinetc` has, by default, an incoming connection limit which can cause FishEye to disrupt other `svnserve`-based connections. The default is blank (do not throttle).

Charset

The character set used to interpret and display text files.

Access Code

The access code for the `fisheye.access` property on the server. See also Subversion `fisheye.access`.

MD5 Access Code

The MD5 sum of the above Access Code. See also Subversion `fisheye.access`. (This field only appears if Access Code is set.)

Set Access Property Command

The Subversion command to set the `fisheye.access` property to grant FishEye access if necessary. See also Subversion `fisheye.access`. (This field only appears if Access Code is set.)

Start Revision

If set, the revision number from which FishEye will start indexing the repository. The default is to start scanning from the first revision in the repository.

Initial Import

When a Start Revision is set, this setting controls how FishEye establishes the initial state of the repository.

- **Do not import** means that FishEye will only process the revisions from the start revision onwards. The repository state prior to this revision is ignored.
- **Import without tag information** means that FishEye will import the repository content as it existed one revision prior to the start revision. FishEye will create a single synthetic revision to hold the initial state. The comment associated with this revision will be 'Created by FishEye for initial repository import'. Tags created prior to the start revision are ignored.

Username/Password

The credentials to use if your repository requires authentication.

Follow Base Moves

If you have changed the location of your repository in your Subversion server over time, set this option to true. Doing so will cause FishEye to index your repository’s full history across all its locations within your server.

trunk/branch/tag structure

Determines how FishEye attempts to understand the tag and branch structure of your Subversion repository. Read more information.

Enable immediately

Controls whether FishEye will immediately enable this repository, which starts the initial scan. If you wish to do some further configuration before the scan starts, then select ‘No’. You can enable a repository later from the repository list.

⚠️ If FishEye returns the error `Authentication Cancelled`, this may simply indicate an incorrect username or password.

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.

*Screenshot: Adding a SVN Repository*
The `fisheye.access` property allows an administrator/committer to control FishEye access to a directory in the repository. FishEye queries this property to decide whether it will continue to access the repository. If the property exists, but does not match that configured in FishEye, FishEye will immediately disconnect from the repository.

By default, FishEye will have access to your repository and you do not need to configure this access mode. It is only necessary if you identify a need to restrict FishEye access to your repository (this is generally not required).

### Setting FishEye Access Mode

FishEye can operate in one of three access modes:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Access</th>
<th>Subversion repository property:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><code>fisheye.access</code></td>
</tr>
</tbody>
</table>
If you would like to restrict FishEye access to your repository, you must set the `fisheye.access` property. This property must be set on the 'URL + path' you have configured in FishEye.

### Setting an Access Code

The repository must be configured with the MD5 sum of the Access Code that is configured in FishEye. The MD5 sum and even the `svn` command to set the property will be generated for you by FishEye when you configure the repository using the FishEye Administration page. See Subversion repository details.

For example, if you have configured FishEye with a URL of `svn://foo.com/` and a path of . and an Access Code of 'fisheye', then you would need to do something like this:

```
$ svn checkout -N svn://foo.com/
   tmpworkspace
$ cd tmpworkspace
$ svn propset fisheye.access
   "md5:4d0c5db8382f80c58e7b0619ae5767a7" .
$ svn commit -m "grant fisheye access"
```

### Denying Access to all FishEye Instances

To deny all FishEye instances access to the repository, it must be configured with the `fisheye.access` property of 'deny'.

For example, if you have configured FishEye with a URL of `svn://foo.com/` and a path of . (or you have left path empty), then you would need to do something like this:

```
$ svn checkout -N svn://foo.com/
   tmpworkspace
$ cd tmpworkspace
$ svn propset fisheye.access "deny" .
$ svn commit -m "disable fisheye access"
```

If you configured a path of `some/dir` then use:
SVN Tag & Branch Structure

Since tags and branches in Subversion are implemented via directory copies, they are not really first-class concepts. You can describe what your tag/branch structure looks like, and FishEye will display the related branch and tag information for your files. These settings can be edited on the 'Add Repository' or 'Edit Repository' pages in the FishEye Administration pages.

The symbolic setup tells FishEye how to classify each path it encounters in the repository. Each path is classified as either a trunk, branch, tag or root path. The root category is used when a path does not match any of the given trunk/branch/tag settings and is mostly treated in the same way as trunk paths.

- The symbolic settings do not exclude any paths from consideration by FishEye. To exclude paths you should set up appropriate 'allow' rules. If your symbolic setup does not match a path, that path will be classified as a root path and processed by FishEye accordingly.

- If you change these trunk/branch/tag settings, you would normally perform a complete re-scan of the repository to ensure FishEye's index is consistent with the settings. FishEye will suggest this when you make changes and you can also do this manually from the Indexer option.

For more information on tag/branch layout, see Repository Layout in the Subversion documentation, or How Tags Work in Subversion if you are having trouble.

In-built Symbolic Rules

FishEye provides In-built symbolic rules which cover most common Subversion conventions. If your branch and tag structure is straightforward and you use the common conventions for directory names, etc. you should use this option. If you are using a non-standard structure then you will not be able to use this option. You should set this to false and use the custom layout option.

- If you set "Use in-built Symbolic rules" to true then DO NOT define a custom layout below. Only define the custom layout if you have set this to false.

Common layouts

There are two common repository layouts that you can choose from in FishEye. These layouts are described in Repository Layout in the Subversion documentation. In most cases you will no longer need to use these options directly as you can use the In-built symbolic rules option described above. These options can be useful to use as the starting point of a custom layout if you only need to support a minor variation to the standard layouts.

```
$ svn checkout -N svn://foo.com/some/dir tmpworkspace
$ cd tmpworkspace
$ svn propset fisheye.access "deny" .
$ svn commit -m "disable fisheye access"
```
The first is where there are top level trunk, branches and tags directories. This is called '/trunk/..., /branches/NAME/..., /tags/NAME/...' in FishEye.

The second is where the trunk, branches and tags directories are one level down, under each top-level project directory. This is called '/project/trunk/..., /project/branches/NAME/..., /project/tags/NAME/...' in FishEye.

**Custom layouts**

You can describe to FishEye any custom tag/branch structure you have. If you want to use one of the common layouts as a basis, first select it from the dropdown, then select 'Custom' to edit/add rules.

When looking at a file on a branch, or a file that was tagged, FishEye needs to determine a name for the branch/tag. FishEye does this by matching a regular expression against the file's path, and extracting the name based upon the match. FishEye also needs a name for files on the trunk. In effect, this is the name of the trunk 'branch'.

For any file that matches a trunk/branch/tag regular expression, a logical path is calculated. Two different files with the same logical path are considered to be related. For example, using the second type of common repository layout:

- The file `project1/trunk/dir1/foo.txt` would have a logical path of `project1/dir1/foo.txt`.
- The file `project1/tags/BUILD123/dir1/foo.txt` would have a logical path of `project1/dir1/foo.txt` and the name of the tag would be `project1-BUILD123`.
- Both these files have the same logical path, and so are considered related. By looking at the revision where the directory-copy for `project1/tags/BUILD123/dir1/foo.txt` occurred, FishEye can determine to what revision the tag `project1-BUILD123` applies.

You can add as many rules as you need. For any given file, the first rule that matches is used.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regex</td>
<td>The regular expression used to match against the start of the path. The trailing part of the path that does not match the regex is called the tail.</td>
</tr>
<tr>
<td>Name</td>
<td>An expression used to extract a tag or branch name from the regex.</td>
</tr>
<tr>
<td>Logical Path Prefix</td>
<td>This is an expression used to construct the logical path. The logical path is the concatenation of the result of this expression, and the tail of the regex.</td>
</tr>
</tbody>
</table>

**How Tags Work in Subversion**

If you are having trouble finding out the correct Tag or Branch names to use under Subversion (especially when searching or creating EyeQL queries), the steps on this page will help.

**Discovering your Subversion Tag & Branch Structure**

1. Open a file in your repository using FishEye. Look for a file that has a long lifetime (for example, your build script).
2. Look at the 'Tags' section to see examples of tags in your system. There are a few different conventions, but the tags shown will give you clues as to how your repository custodians do things. A typical tag looks like this:

   ```
   fisheye-build-82
   ```

3. Copy the tag(s) you may need from the examples you can see in FishEye's view.
4. Enter the tags exactly as they appear into your EyeQL queries or (FishEye Search).
5. Test the outcome.

**Screenshot: Subversion Tags viewed in FishEye**

<table>
<thead>
<tr>
<th>Branch fisheye-fE皤nmBranch</th>
</tr>
</thead>
<tbody>
<tr>
<td>6267 annotated / raw</td>
</tr>
<tr>
<td>Created: 2005-08-20 01:29:05 +1000 (2 years 7 months ago)</td>
</tr>
<tr>
<td>NONE: handle files with no logical path. Pass config around to have access to username/password</td>
</tr>
<tr>
<td>Tags: fisheye-build-91 fisheye-build-90 fisheye-build-89 fisheye-build-88 fisheye-build-87 fisheye-build-82</td>
</tr>
<tr>
<td>Properties</td>
</tr>
<tr>
<td>svn:executable = *</td>
</tr>
<tr>
<td>svn:keywords = Author Date Id Revision</td>
</tr>
</tbody>
</table>

**How tag names are constructed**

Hyphens are the default method for separating the tag elements, but tags in your FishEye instance may be different.

Your tag structure depends on several things:

- The symbolic structure of your repository
- The way your FishEye instance was set up
- Your organisation's convention for naming things in the repository
- The configurable character that separates parts of the tag name.

This can sometimes make it confusing to guess what the tag structure could be, when you are searching in FishEye or using an EyeQL query. Additionally, the separator between tag elements can be configured as hyphen, colon, or other punctuation marks.

Note that slashes cannot be used in a tag name – these are converted to colons by default. For example if your symbolic setup would give you a tag name like the following:

```
path/project/fisheye-build-82
```

then your tag should look like this:

```
path:project:fisheye-build-82
```

For more information, see SVN Tag & Branch Structure.

**Repository Options**

FishEye has configuration options for each repository and default settings that will affect all repositories.

- To access the settings for a specific repository, click the name of the repository in the 'Admin Menu', or click 'Repository List' and then click 'View' next to the repository name.
- To change settings that will affect all repositories, click 'Repository Defaults' in the 'Admin Menu'.
Some changes will require the repository to be restarted, while others will require the repository to be re-indexed. FishEye will advise you if this is the case when you make the change. You can restart a repository from the 'Repository List'.

Repository options:

- Configuring Repository Details
- Store Diff Info
- Operations
- Symbolic
- Indexer
- Updater
- Linkers
- Permissions
- Watches
- Allow (Process)
- Hidden Directories
- Tarball Settings
- Properties
- Commit Message Syntax

Screenshot: Repository Options
Configuring Repository Details

When adding or managing a repository, you can:

- Define repository details, as described below.
- Set FishEye's repository options.

To access the details of a repository,

1. From the 'Admin Menu', choose one of the following:
   - Click 'Repository List' then click 'View' next to the repository name.
   - Or just click the repository name in the 'Admin Menu'.
2. Then click 'Edit Details' on the 'View Repository' page.

Repository details differ depending on the repository type. FishEye currently supports the following repository types:

- ClearCase
- CVS
- Git
- Perforce
- Subversion

Store Diff Info

This page explains the 'Store Diff Info' setting in FishEye and how to switch it on and off.

On this page:

- About the Store Diff Setting
- Turning 'Store Diff Info' on and off
- Considerations for the Store Diff Info Setting
- Required for Per-author Line Graphs
- Required for Displaying Context of Search Results
- Re-index Required for Per-author Line Graphs
- Perforce Repository Indexing Performance Impact

About the Store Diff Setting

Store diff info means that FishEye is caching the summary of what lines are added and removed between subsequent versions of the same file in its database. In other words, we are storing the info of a diff, not the diff itself (thus you will still be able to view diffs if this value is set to off).

After setting this value to on, a full re-index of your repository is required so that FishEye can collect diff info for all revisions in your repository.

Turning 'Store Diff Info' on and off

To find this setting, open the FishEye Administration Screen, then click 'Repository List' under 'Repository Settings'.

For the desired repository, click 'View'. The 'View Repository' screen opens. In the section labelled 'Repository Details', click 'Edit Details'. The 'Edit Repository Details' screen opens.

Screenshot: The View Repository Screen

In the 'Edit Repository Details' screen, 'Store Diff Info' is a value that can be toggled on or off. On is the default setting for new repositories.

Screenshot: The Edit Repository Details Screen
Considerations for the Store Diff Info Setting

Required for Per-author Line Graphs

Leaving this option off will disable per-author line graphs.

Required for Displaying Context of Search Results

Leaving this option off will disable the display of context of found terms in search results.

Re-index Required for Per-author Line Graphs

Diff info is always stored for CVS repositories. Note that a full re-index is required to enable per author charts after upgrading from FishEye 1.4.3 or earlier.

Perforce Repository Indexing Performance Impact

Turning ‘Store Diff Info’ on for Perforce repositories requires FishEye to make extra requests to your depot in order to collect the diffs. This may substantially increase the time it takes to scan your repository.

Repositories created before FishEye 1.5 will default to ‘Store Diff Info’ off.

CVS and Subversion repository scan times are not affected by this setting.

Operations

You can carry out various operations on a FishEye repository in the Repository Options screen.

To get to this page:

1. Open FishEye ‘Administration’,
2. Find the desired repository on that page and click ‘View’ in the repository controls. The Repository Options screen opens.
3. Scroll down to the ‘Operations’, section in the repository options. A summary of the current status is shown.
4. Here, you have a number of operation actions you can carry out by clicking:

- **Browse**: This will allow you to browse the repository.
- **Stop**: This will stop the repository scan.
- **Restart**: This will restart the repository scan.
- **Disable**: This will disable the repository.
- **Delete**: This will delete the repository profile from FishEye.

Screenshot: Operations Panel in FishEye Repository Options
## Symbolic

In the Repository Options screen, you can view the symbolic settings for the current repository. You can click 'Edit' to adjust these settings.

For more information, see SVN Tag & Branch Structure.

![Screenshot: Symbolic Settings in FishEye Repository Options](image)

### Indexer

To get to this page, go to 'Administration' > 'Repository List' > 'View' (next to your repository name) > 'Indexer' > 'Maintenance'

The 'Indexer Maintenance' screen opens, where an administrator can manually trigger the following actions:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh</td>
<td>Refreshes the status string, in the Status box. This will show the name of any repository task that FishEye is currently carrying out (if any). This may be helpful to monitor the progress of an initial indexing task or to diagnose problems (such as your repository server being out of action).</td>
</tr>
<tr>
<td>Re-index Repository</td>
<td>Delete the current cache and re-index the repository from the beginning. This action will also restart the repository.</td>
</tr>
<tr>
<td>Re-index Crucible Data</td>
<td>(applies only when using Crucible with FishEye) Re-index all the Crucible review data associated with the current repository.</td>
</tr>
<tr>
<td>Re-index Linecount Data</td>
<td>Re-index the linecount data used to generate the LOC (Lines Of Code) charts. The linecount data will be recalculated in daily buckets based on the server timezone.</td>
</tr>
<tr>
<td>Scan Now</td>
<td>Run a repository scan (also referred to as indexing) now. If the repository has already been indexed, it will be an incremental scan, otherwise an &quot;initial&quot; scan. This is especially useful if you have not set the repository to poll automatically, or it is set with a long poll (interval) period.</td>
</tr>
</tbody>
</table>
Rescan Revisions

For a **Subversion** repository, you can rescan non-versioned properties (revprops). In Subversion it is possible to enable non-versioned properties (e.g., the check-in comments) to be updated by committers. When this happens, FishEye will not automatically pick up the updates. By rescanning specific revisions, FishEye will rescan the non-versioned properties and amend the entry in FishEye accordingly.

For a **Perforce** repository, you can rescan changelists. This will rescan changelist metadata (author, date, commit message) and any job fixes associated with changelists in the given changelist range.

**Screenshot: FishEye Index Maintenance menu**

![FishEye Index Maintenance menu](image)

**Updater**

The **Updater** repository option allows an administrator to manually trigger the actions described below, depending upon your repository type.

- Updater (Affects all version-control repositories)
- Updater (CVS)

Updater (Affects all version-control repositories)
Poll Period
How often FishEye will check to see if there have been any new commits into the SVN or Perforce repository. The default is 60 seconds. It is possible to set the period by units. For example: 10second, 1week. Valid units are 'second', 'minute', 'hour', 'day', 'week', 'month', 'year'. The default unit is days if only a number is added.

⚠️ You can also set the value 'never', which creates a situation where scanning is purely manually controlled via the command line, or the Scan Now option on the Indexer Maintenance page, accessed by clicking 'Maintenance' on the 'View' page for each repository.

Updater (CVS)

FishEye will monitor your CVS history file `CVSROOT/history` to determine what has changed in your repository. FishEye will also periodically scan the whole repository.

CVS is not always configured to create a history file. Talk to your CVS administrator.

The default values should be fine for most repositories. Leave a value blank to use the default value.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>History file</td>
<td>The location of the CVS history file. If relative, then it is relative to the CVS directory specified for this repository. Defaults to <code>./CVSROOT/history</code>.</td>
</tr>
<tr>
<td>Full scan period</td>
<td>How often FishEye will do a full scan of the repository. Defaults to 15 minutes. Specify an interval, such as '15 min', '2 hours', etc. A value of '0' disables the periodic full scan. (You can still use <code>fisheyectl fullscan</code> to cause a full scan to occur.)</td>
</tr>
<tr>
<td>Strip prefix</td>
<td>Prefix to strip off files found in the history file, to make them relative to this repository's CVS directory. Necessary if the CVS directory specified is not the root of the CVS repository. For example, your CVS is located at <code>/usr/local/cvsroot</code>, but you specified <code>/usr/local/cvsroot/foo/bar</code> as the CVS directory of this repository. You will need to give the history file as <code>../../CVSROOT/history</code> and set a strip prefix of <code>foo/bar</code>.</td>
</tr>
</tbody>
</table>

⚠️ Once you have changed the value, you will need to restart fisheye. The period begins from when the initial index completes, i.e. when you restart, your repos will be scanned in order (depending on the number of threads you have configured) and when this scan completes this is the start of the period.

For example, if you set one hour, then your next scan will begin one hour after your initial scan is complete.

Linkers

FishEye can detect special substrings in commit messages, and hyperlink those substrings to other systems. The result is, when browsing commit messages or comments in FishEye, any issue IDs or Bug IDs that appear will be turned into hyperlinks, allowing you to easily click to see those referenced issues or pages.

This is particularly useful if you use an issue tracking system, and put the issue identifiers into your commit messages. The 'Linkers' repository option (Administration > Repository Defaults > Linkers, or Administration > View Repository List > View (next to your REPO) > Linkers) allows you to define the substrings and their related URLs.

Any linkers defined in the repository defaults are added to each individual repository.

Example Linkers
Here are some examples of how to create simple linkers.

**JIRA examples**

If you have already set up JIRA integration, ignore the following instructions.

- To link any occurrence of a JIRA-style issue to JIRA:

  ```
  Regex: [a-zA-Z]{2,}-\d+
  Href: http://jirahost:8080/browse/${0}
  ```

  The regular expression above matches any sequence of two or more alphabetical characters, followed by a dash, followed by a number, which comprise the format of JIRA issue IDs (such as AB-123 or ABC-123 or ABCDE-123). Replace `jirahost` with the hostname of the desired JIRA instance.

- To link a specific set of JIRA projects (e.g. JRA, CONF and CRUC) to a JIRA instance:

  ```
  Regex: (JRA|CONF|CRUC)-\d+
  Href: http://jirahost:8080/browse/${0}
  ```

  The regular expression above matches only specific JIRA issue keys with any number, like JRA-123 or CONF-123 or CRUC-123. Replace `jirahost` with the hostname of the desired JIRA instance.

**Bamboo examples**

- To link to specific Bamboo builds:

  ```
  Regex: (ABC)-[a-zA-Z]+-\d+
  Href: http://bamboohost/browse/${0}
  ```

  The regular expression above matches Bamboo build IDs like ABC-MAIN-123 or ABC-BRANCH-123. These will then be made links to the build reports in your Bamboo instance. Replace `bamboohost` with the hostname of the desired bamboo instance.

**Bugzilla examples**

- To link bug numbers that occur at the start of a line to Bugzilla:

  ```
  Regex: ^BUG: (\d+)
  Href: http://bugzilla/bugzilla/show_bug.cgi?id=${1}
  ```

- To link bug numbers that occur after the word bug and optionally whitespace, ":" or ":" (e.g. Bug123, bug:123, or BUG #123):
The regular expressions above matches Bugzilla bug IDs. These will then be made links to build reports in your Bamboo instance.

About FishEye Regular Expressions

FishEye uses the Java regular expression language, which is based on Perl 5 regular expressions.

Note: If you want your regex to be case insensitive, put (?i) at the start of the regex.

To try out your regular expressions, you can use this online test page.

Permissions

For each repository in FishEye, you can adjust per-repository permission settings.

To do this:

1. Open FishEye 'Administration',
2. Find the desired repository on that page and click 'View' in the repository controls. The Repository Options screen opens.
3. Scroll down to the 'Permissions', section in the repository options.
4. Click 'Edit' to change the permissions. You can enable or disable anonymous access, as well as adding or removing groups from those that are available. For more information, see Associating a Group with a Repository.

Screenshot: Per-Repository Permissions Summary in FishEye

Permissions 🎉

<table>
<thead>
<tr>
<th>Allow anonymous access:</th>
<th>NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built-in groups:</td>
<td>atlassian-dev atlassian-staff crucible-users-non-crowd KLDDevToolsSupport</td>
</tr>
</tbody>
</table>

Edit

Screenshot: Editing Per-Repository Permissions
Watches

FishEye has a watch notification system that allows users to receive email notifications when commits are detected. The 'Watches' repository option allows you to disable this functionality in the Repository Defaults and on a per-repository basis.

⚠️ Watch functionality requires a valid SMTP server to be configured.

Allow (Process)

By default, FishEye will cache and index your whole repository, and present all of this information to users. You can control what parts of your repository FishEye will access, by setting the 'Allow (Process)' repository option.

Includes

The 'Includes' subsection defines what subtrees of your repository FishEye will index. FishEye defaults to including 'everything'. If you specify some 'include' directories, then FishEye will process only those directories (and all their subdirectories). For instance, you might want to do this to limit FishEye to the subset of active projects in your repository. Each include specifies the path to a subtree to be processed. Paths are expressed relative to the repository root configured in the repository configuration.

⚠️ By default Includes are case sensitive but they can be configured to be case-insensitive. If your repository is set to be case insensitive (Perforce) then Includes will also be case insensitive regardless of how the individual Includes are configured.

Examples:

- Including directories:

  ```
  /PROJECT1
  ```

  The code above includes /PROJECT1 and all its children (sub-directories and their contents). You could specify /PROJECT1/ and /PROJECT2/ to include both of these directories in FishEye's indexing.

Excludes
The 'Excludes' subsection allows you to specifically exclude files and directories from those which have been included in indexing. FishEye will not process these files and directories. Each exclude is an Antglob Pattern. Examples:

- Excluding directories:

```plaintext
/PROJECT2/
```

The code above excludes `/PROJECT2` and all its children (sub-directories and their contents).

- Excluding file types:

```plaintext
**/*.OBJ
```

The code above excludes any OBJ (object) files.

⚠️ Changes to Includes and Excludes do not take effect until the repository is restarted. If you do not re-index when changing the includes and excluded, files and directories which have been indexed prior to the update will remain visible in FishEye.

### Include/Exclude Processing

Currently FishEye will process include definitions first and then excludes. If there are any includes defined a path must match at least one of those includes to be considered. If there are no include patterns defined, all paths are considered to be included. Once includes have been processed, a path which is a candidate for processing is tested against any defined excludes. If the path matches any of the exclude patterns, the path is excluded and not included in FishEye.

⚠️ Includes do not support Antglobs. They are also defined relative to where FishEye connects to your repository. To match the 'tags' subtree, simply use `/tags`. Please note that Excludes can still make use of Antglobs.

### About Setting the Repository Root

When you are setting the Allow (Process), you should be aware that the options on this page only act on the parts of the repository that lie under the level of the repository root, which you configure as a directory location in your repository. In other words, FishEye can only access directories "lower" than the repository root. For example, consider a repository with the following structure:

```plaintext
/CORE/2007/LEGACY/
/CORE/2008/PROJECT1/
/CORE/2008/PROJECT2/
```

In this case, you could set the repository root (or 'Path') to be `/CORE/2008/`. In that situation, you would be able to
include or exclude the `/PROJECT1/` and `/PROJECT2/` directories, but the `/CORE/2007/LEGACY/` directory would not be available. To have FishEye index all of the directories in this repository, you would need to set the repository root path to be `/CORE/`. Then, you could use the includes and excludes to add and remove directories under `/CORE/` from FishEye's scan. For more information, see the Subversion configuration page and read the 'Path' options.

**Hidden Directories**

The 'Hidden Dirs' repository option allows you to mark unused (deprecated) directories as 'hidden'. Hidden directories will not appear in the FishEye user interface unless the user has specifically toggled 'Show hidden directories'. FishEye will still index and cache these directories.

This can be useful if you have old directories that you don't want cluttering the screens by default.

**Tarball Settings**

FishEye contains a feature that will build an archive of a directory tree. This feature is disabled by default. The 'Tarball Settings' repository option allows you to customise tarball settings in the Repository Defaults and on a per-repository basis.

You can set a limit on the number of files that a tarball can contain.

You can selectively disable the creation of tarballs for certain directories or directory trees.

**Properties**

The 'Properties' repository option allows you to customise the behaviour of FishEye. Specifically, you can remove the graph and calendars from certain screens.

A property may be set either per repository or globally as a repository default. A repository default property is inherited by all repositories. A default property may be overridden at the repository level.

The following properties are supported:

<table>
<thead>
<tr>
<th>Name</th>
<th>Possible Values</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>show-changelog-calendar</td>
<td>true, false</td>
<td>false</td>
<td>If set to false, the calendar is disabled on the Changelog page. This may be required for performance reasons. The revision totals displayed per calendar day, month and year may be expensive to calculate. For repositories with a lot of historical data, disabling the calendar can result in significant performance improvements when viewing the Changelog page.</td>
</tr>
<tr>
<td>enable-line-history</td>
<td>true, false</td>
<td>true</td>
<td>Allows you to disable (hide) the line-count history graph on the Browse and Changelog pages. This may be desirable if you have a large repository and generating the line graphs takes a long time.</td>
</tr>
</tbody>
</table>

**Commit Message Syntax**

FishEye can render Wiki Markup in commit messages. This page contains instructions for turning this setting on and off. You can set this option as a global default, as well as individually per-repository.

ℹ️ When using Crucible, you can also have Wiki Markup rendering in review comments and review descriptions.

**Adjusting the Global Default Setting**
1. Go to 'Administration' > 'Repository Defaults', then scroll down to the 'Commit Message Syntax' option.
2. Click 'Edit' to make changes. You can choose from the following options:

- **Wiki Markup**: Wiki Markup is rendered in Commit messages.
- **Plain text**: Wiki Markup is not rendered in FishEye commit messages.

⚠️ The factory setting is 'plain', i.e. FishEye will not render commit messages.

**Screenshot: Setting Global Wiki Rendering for Commit Messages**

Adjusting the Per-Repository Setting

To adjust this, go to 'FishEye Administration' > 'Repository Options' > 'Commit Message Syntax'.

You can choose from the following options:

- **Wiki Markup**: Wiki Markup is rendered in Commit messages.
- **Plain text**: Wiki Markup is not rendered in FishEye commit messages.

**Screenshot: Setting Per-Repository Wiki Rendering for Commit Messages**

**Setting up a Repository Client**

- CVS Client
- Git Client
- Perforce Client
- Subversion Client

**CVS Client**

FishEye supports CVS repository access, but this does not require installation of a separate client.

As FishEye reads your CVS repository directly from the local file system, using a CVS repository is a 'zero-installation' scenario. Once you have set up FishEye on the system where your CVS repository resides, FishEye should be able to access it automatically once you have configured it.

See Configuring FishEye to access your CVS repository for more information.

**Git Client**

⚠️ The information on this page relates to FishEye's Early Access Program support for Git. Do not use in production.
Git 1.6 is supported at this stage.

Introduction and Disclaimer

Atlassian stresses that this is not final support for Git which means the following:

1. Subsequent updates to the technology are likely to require re-indexing (although Git indexing is reasonably fast).
2. Problems are to be expected. Do not use this on production instances.

However, feedback (see below) is appreciated and your comments will contribute to the quality of Git support in FishEye.

Configuration

FishEye requires Git 1.6.0 or later. At present, you need to have the Git executables in the path of the user used to run your FishEye instance.

Submitting Feedback

We're very interested in your feedback. The best place for submitting feedback is the FishEye forums.

Related Links

- Git Configuration
- Crucible Repository Configuration
- General Crucible Configuration

Perforce Client

FishEye can communicate with any Perforce server, but it needs to use the P4 command-line client to do so.

By default, FishEye looks for the `p4` executable in the current path. To specify the exact path of the `p4` executable, click 'Server Settings' in the FishEye 'Admin Menu'.

**Files incorrectly considered binary**

Some users have reported errors where FishEye considers some files to be binary when they are not. It appears this may be a limitation of earlier P4 clients. If you can upgrade to a recent P4 client (2006.1 onwards), this will fix this issue. You do not need to update the P4 Server.

If you are unable to upgrade to a recent P4 client, the Repository Details page in FishEye allows you to set a limit on the size of filelog commands sent to the server. Setting this to something around 100 will fix the issue. It will, however, also impact performance significantly.

Subversion Client

FishEye can communicate with any Subversion server running version 1.1 or later, but it needs to use a Subversion client to do so.

The SVNKit client is included in the current FishEye package (version 1.4.2 onwards). This is the default client for interfacing with Subversion, is generally the easiest to use, requires "zero-installation" and will be used automatically unless another client is specified. The SVNKit client is recommended for most users.
The alternative is the native client, which should only be used if the SVNKit client is unsuitable.

- Native Subversion Client
- SVNKit Client

⚠️ Using the 'file:///' protocol to access your Subversion repository can be much faster than the other network protocols. We recommend using the 'file:///' protocol if possible.

⚠️ Using Subversion 1.3 or later is strongly recommended. Versions prior to 1.3 are no longer supported by the Subversion project. They will work with FishEye, but you may want to consider upgrading to a supported version.

**Native Subversion Client**

Native Client

FishEye can use a native Subversion client installed on your system, but your client needs to be version 1.2 or later, and **must include the JavaHL bindings**. FishEye can use all of the protocols supported by your native client.

The JavaHL bindings include a Java .jar file, typically named jasvnhl.jar, and a dynamic library such as libsvnjavah-1.so or libsvnjavahl-1.dll. FishEye must be configured so it can find both the .jar and the dynamic library.

If the JavaHL dynamic library is in your library path (such as %PATH% on Windows), then FishEye will automatically find it. Otherwise you can tell FishEye where it is, or set the FISHEYE_LIBRARY_PATH environment variable before starting FishEye.

**Acquiring native Subversion libraries for your operating system**

Pre-compiled native clients are available for most platforms. The Subversion download page links to platform specific distributions. **Ensure you get the binary that includes JavaHL bindings**, as well as the standard package. Also ensure that the versions of the JavaHL and standard packages match.

- **Subversion for Windows:**
  To install Subversion for Windows, visit this page
  You need to download the standard package as well as the JavaHL version. The standard package is named `svn-X.Y.Z-setup.exe` and the JavaHL installer file is named `svn-win32-X.Y.Z_javahl.zip` where `X.Y.Z` refers to the version number (for example, `svn-win32-1.4.6_javahl.zip` at the time of writing).

- **Subversion for Fedora Linux:**
  For Linux systems using the yum package manager (such as Fedora Core 3 and above) you can type the following:

  ```sh
  yum install subversion-javahl
  ```

  at the Linux command line interface to install the JavaHL bindings for Subversion. Note that this will also install the standard Subversion library, which is required.
**Subversion for Ubuntu and Debian Linux:**
For Linux systems using the apt-get package manager (such as Debian and Ubuntu) you can type the following:

```
apt-get install libsvn-javahl
```

at the Linux command line interface to install the JavaHL bindings for Subversion. Note that this will also install the standard Subversion library, which is required.

### Native Client Configuration

There are two ways you can configure the path to your Subversion client: Via the FishEye user interface, or by editing the `config.xml` configuration file.

#### Configuring your Native Client in the FishEye User Interface

You can configure your Subversion client in the FishEye Administration screens, under **Admin > Server Settings > Subversion Client**.

| JAR | The path to the JavaHL `.jar`. |
| Dynamic library | The path to the dynamic library, if it is not already on your system's library path. |

⚠️ Due to a bug in earlier versions of the JavaHL bindings, setting this value is ineffective unless you are using a Subversion client 1.2.3 or later.

#### Configuring your Native Client in the FishEye Configuration File

An alternative method to using the FishEye user interface is editing the `<svn-config>` section of your `config.xml`. If you change these settings, you need to restart FishEye.

**Windows Platform Example (change path locations as required)**

```
<svn-config jar="C:\subversion\lib\svn-javahl.jar" jnilib="C:\subversion\lib\libsvnjavahl-1.dll"/>
```

**Mac OS X Platform Example (change path locations as required)**
SVNkit Client

SVNkit is included in the current FishEye package, and is the default library for interfacing with Subversion. It is generally the easiest to use, and will be used automatically unless another library is specified.

See Configuring Subversion repositories for more information.

⚠️ SVNKit supports the "file://" protocol for FSFS repositories only.

SVNKit sometimes has problems working with Subversion servers which are running older versions, such as 1.1.x. If you see exceptions such as those listed below in FishEye's log file, you will need to either swap to use the native client or upgrade your Subversion server to version 1.3 or later.

Example exceptions:

- SEVERE: assert #B
- Checksum mismatch while reading representation:

Other Subversion Clients

Native Subversion Clients are also supported by FishEye as an alternative to SVNkit. This process requires additional configuration. Read the Native Subversion Client instructions for more information.

Configuring ViewVC Compatibility

For backwards-compatibility and legacy system support, FishEye contains a URL-compatibility mode with the
ViewVC (formerly known as ViewCVS) and CVSWeb tools. This allows FishEye to supplant or extend ViewVC, making use of the URLs set up for ViewVC.

To configure ViewVC compatibility, click ‘ViewCVS URL Mappings’ on the ‘Admin Menu’.

For example, a ViewVC URL of the form

http://host/viewcvs.cgi/x/y/z

can be viewed in FishEye at

http://fisheye_host/viewcvs/x/y/z

FishEye can be configured to determine exactly how it provides this compatibility mode. In particular, you can configure how to map ViewVC repository names (cvsroot or root in the query parameter) to FishEye repository names.

The Default Mapping can be used to configure which repository to use if no repository is specified in the URL. If a repository name is given in the URL, you can tell FishEye how to translate that to the name of a FishEye repository. Otherwise, FishEye will attempt to use the repository name given in the URL directly.

Screenshot: Configuring ViewVC Compatibility

Once you have entered the mappings you will need to restart Fisheye.

Setting up your Web Server

To configure the server settings, click ‘Server Settings’ on the ‘Admin Menu’.

- Configuring the FishEye Web Server
- Integrating with Other Web Servers

Screenshot: Configuring Server Settings
Configuring the FishEye Web Server

To configure the server settings, click 'Server Settings' on the 'Admin Menu'.

⚠️ Some settings on this page require you to restart FishEye for changes to take effect.

<table>
<thead>
<tr>
<th>Setting</th>
<th>HTTP Bind</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The hostname the FishEye web server will bind to. This can take the form of a host name and port number, or you can leave the host name blank. If no host name is specified, then FishEye will bind to all available interfaces.</td>
</tr>
</tbody>
</table>
### Examples

You can use bindings like the following:

- **Host name and port number**:

  ```
  hostname:8060
  ```

- **port number only** (requires a leading colon):

  ```
  :8060
  ```

- **IP address and port number**:

  ```
  10.0.0.11:60
  ```

(At least one of 'AJP13 Bind' or 'HTTP Bind' must be set.)

> Do not add http:// to the front i.e. do not add http://hostname:8060 or http://10.0.0.11:60. Simply define hostname:port or IpAddress:port.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Web context</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>By default, the FishEye application can be accessed via <a href="http://HOST:PORT/">http://HOST:PORT/</a>, where HOST and PORT are defined as above. You can force the FishEye application to be hosted on a different 'context' or 'path' by specifying a value here.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>If you specify a web context of 'fisheye' then FishEye will be accessible from <a href="http://HOST:PORT/fisheye/">http://HOST:PORT/fisheye/</a> instead of <a href="http://HOST:PORT/">http://HOST:PORT/</a>.</td>
</tr>
<tr>
<td><strong>Restart required</strong></td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setting</th>
<th>Proxy scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Can be set if you are forwarding through to FishEye from another web server.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>Valid settings are http and https.</td>
</tr>
<tr>
<td><strong>Restart required</strong></td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setting</th>
<th>Proxy host</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Can be set if you are forwarding through to FishEye from another web server.</td>
</tr>
</tbody>
</table>
A valid setting would be www.example.com, where 'example' is the domain name of your web server.

**Proxy port**

Can be set if you are forwarding through to FishEye from another web server.

The port number of the web server, an integer from 0 through 32,765.

Yes

**AJP13 Bind**

The bind host name for ajpv13. If no host name is specified, then FishEye will bind to all available interfaces.

You can use bindings like the following:

- **Host name and port number**:
  
  `hostname:8009`

- **port number only** (requires a leading colon):
  
  `:8009`

- **IP address and port number**:
  
  `10.0.0.11:8009`

(At least one of 'AJP13 Bind' or 'HTTP Bind' must be set.)

Do not add http:// to the front i.e. do not add http://hostname:8060 or http://10.0.0.11:60. Simply define hostname:port or IpAddress:port

Yes

**Remote API**

Enables/disables FishEye's Remote API. Clicking on the help link will take you to the API doc.

'On' enables the Remote API.
## Server timezone

<table>
<thead>
<tr>
<th>Setting</th>
<th>Server timezone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The time zone to use within FishEye. This time zone is used when displaying dates and parsing EyeQL date expressions. If blank, then the time zone of the server running FishEye is used.</td>
</tr>
<tr>
<td>Example</td>
<td>This defaults to the FishEye server's time zone, but you can select another zone from the drop-down list.</td>
</tr>
<tr>
<td>Restart required</td>
<td>No</td>
</tr>
</tbody>
</table>

## Site URL

<table>
<thead>
<tr>
<th>Setting</th>
<th>Site URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The base URL for this FishEye instance. If not specified, FishEye will attempt to determine this value.</td>
</tr>
<tr>
<td>Example</td>
<td>This is used whenever you have set FishEye up in a sub-folder on the web server, for example <a href="http://www.example.com/apps/fisheye">www.example.com/apps/fisheye</a>.</td>
</tr>
<tr>
<td>Restart required</td>
<td>No</td>
</tr>
</tbody>
</table>

See also Subversion Client settings.

### Integrating with Other Web Servers

FishEye has a built-in web server, but commonly runs in an environment that has its own web server. You can easily proxy through to FishEye from this primary web server, so that it appears as if FishEye is part of the primary web server.

In most situations, FishEye can determine the host and port of the primary web server automatically. This is useful when you have multiple virtual hosts proxied through to the one FishEye instance.

If it appears FishEye is having trouble automatically detecting the primary web server’s host and port, you will need to set the **Proxy host** and **Proxy port** parameters. If the primary web server is running on `WEBHOST:80` and FishEye is running on `FEHOST:8060`, then you can set FishEye’s **Proxy host** and **Proxy port** parameters to `WEBHOST` and `80`.

If the primary web server is using SSL, add the parameter `proxy-scheme="https"` to the connecting server as in the following example:

```xml
<http bind=":8080" proxy-host="extranet.example.com" proxy-port="443" proxy-scheme="https"/>
```

You will probably want FishEye to appear in a subdirectory of the primary server. In that case, you need to set FishEye’s **web context** parameter. The rest of the page assumes you have set this value to `fisheye`.

You will need to restart FishEye before any of the above parameters take effect.
Then, configure your primary web server as described below.

**Apache**

The easiest way to proxy through to FishEye is using the `ProxyPass` directive, which requires the `mod_proxy` module. Add this section to your Apache configuration:

```conf
ProxyPass /fisheye http://FEHOST:8060/fisheye
```

If you want Apache to serve FishEye's static content, then you can do something like this instead:

```conf
<Directory "/FISHEYE_HOME/content/static" >
  Allow from all
  AllowOverride None
</Directory>
Alias /fisheye/static /FISHEYE_HOME/content/static
ProxyPass /fisheye/static/ !
ProxyPass /fisheye http://FEHOST:8060/fisheye
```

⚠️ An alternative to using `ProxyPass` is to use `mod_rewrite` with the `[P]` flag.

**AJP**

FishEye also supports AJPv13 connectivity. For more information, please see AJPv13 Authentication.

### Configuring SMTP

To configure SMTP settings, click 'Server Settings' on the 'Admin Menu'.

You can enter the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>From Address</strong></td>
<td>The from email address used when FishEye sends an email, e.g. '<a href="mailto:fisheye-noreply@example.com">fisheye-noreply@example.com</a>'</td>
</tr>
</tbody>
</table>
### Send mail from
Selects either the 'Server Address' (default, as above) or the 'User Address', which selects the email address in the user's profile.  
(Note: this only applies to Crucible notifications. FishEye will always use the Server Address.)

<table>
<thead>
<tr>
<th>SMTP Host Name</th>
<th>The host name of the SMTP server.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable debug</td>
<td>Optional. Turn this on to enable debug logging from the mail server. Useful in tracking down mail server connectivity problems.</td>
</tr>
<tr>
<td>SMTP Port</td>
<td>Optional, defaults to 25. The port to connect to on the SMTP host.</td>
</tr>
<tr>
<td>Use SSL/TLS</td>
<td>Optional, defaults to 'False'. This turns on Secure Sockets Layer/Transport Layer Security security for mail servers that require it, or use it by default.</td>
</tr>
<tr>
<td>Username &amp; Password</td>
<td>Optional. Username and password for authenticated SMTP access.</td>
</tr>
</tbody>
</table>

Once you have configured SMTP, you can use the 'Send test email' link on the 'Server Settings' page to confirm the SMTP connectivity.

**Screenshot: Configuring SMTP**

---

### Setting up your Users and Security

You can implement access control using a set of 'built-in' users stored in the FishEye database, or you can have FishEye look in an external authentication source for users, passwords and permissions.

Anonymous access to FishEye is allowed by default. You can disable anonymous access at a global level and per repository.

For an overview of FishEye security, please see Understanding security.

To configure your authentication settings, click 'Users/Security' on the 'Admin Menu'.
FishEye provides a pluggable architecture to allow arbitrary forms of authorisation and authentication.

Screenshot: Authentication Settings

<table>
<thead>
<tr>
<th>Permissions Summary</th>
<th>Allow anon access</th>
<th>Groups</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>YES (No)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crucible</td>
<td>NO (Yes)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repository Default</td>
<td>YES</td>
<td>none set</td>
<td>Edit</td>
</tr>
<tr>
<td>Per-repository</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>svn</td>
<td>YES (default)</td>
<td>default</td>
<td>Edit</td>
</tr>
<tr>
<td>website</td>
<td>YES (default)</td>
<td>default</td>
<td>Edit</td>
</tr>
</tbody>
</table>

Built-in

Public Signup: OFF (On)

Authentication settings

- Setup host authentication
- Setup LDAP
- Setup AJP13 authentication
- Setup Custom authentication

Adding a User to a Group

There are two types of FishEye user groups:

- 'Built-in' groups — these are stored in the FishEye database.
- 'External' groups — these are stored in an external directory (e.g. Crowd), if any are configured. See Configuring External Authentication Sources.

Note that for external directories and external groups, users can only be added to external groups from within the directory, so the method described below will not apply. However if you want to add external users to internal (built in) groups, then you can using the below.

To add a user to a 'built-in' group,
1. Click 'Users' on the 'Admin Menu'.
2. The 'User Browser' screen will be displayed (see screenshot below), showing a list of users.
3. Locate the user and click the corresponding 'Edit' link.
   - If the user doesn't initially appear on the screen, type part of their email address and/or select a group to which they belong, and click the 'Filter' button.
4. The 'Edit User' screen will be displayed (for more information please see Editing a User's Details). Click the 'Edit Groups' link.
5. The 'Edit User Groups' screen will be displayed as shown in the screenshot below.
   - To add the user to a group, select the group in the 'Available Groups' column at left and click the 'Join' button.
   - To remove the user from a group, select the group in the 'Groups' column at right and click the 'Leave' button.
6. Click the 'Back to user' button.

**Screenshot: Edit User Groups**

<table>
<thead>
<tr>
<th>Edit User Groups: matt</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Available Groups</strong></td>
</tr>
<tr>
<td>lean-2</td>
</tr>
<tr>
<td><strong>Groups</strong></td>
</tr>
<tr>
<td>lean-1</td>
</tr>
<tr>
<td>lean-3</td>
</tr>
</tbody>
</table>

**Associating a Group with a Repository**

Associating a group with a repository means that only members of that group will be able to access the repository. For more information, please see Understanding security.

To associate a group with a repository,

1. Click 'Security' on the 'Admin Menu'.
2. The 'Authentication Settings' screen will be displayed (see screenshot below), showing a list of existing groups.
3. In the 'Permissions Summary' section, under 'Per-repository', click the 'Edit' link corresponding to the repository with which you wish to associate a group.
   - Or, to associate a group with all repositories by default, click the 'Edit' link in the 'Repository Default' row.
4. The 'Edit Security' screen will be displayed. Select the relevant group and click the 'Join' button.
5. Click the 'Update' button to return to the 'Authentication Settings' screen.
6. Your new group will now appear in the 'Groups' column for your chosen repository.
Configuring Anonymous Access

Anonymous access to FishEye is allowed by default. You can disable anonymous access at the following levels:

- Global.
- Repository default.
- Per repository.

Note: in Crucible, anonymous access is also subject to individual project permissions (see Creating a Permission Scheme).

To configure anonymous access,

From the 'Admin Menu', choose one of the following options:

- 'Security' - Allows you to change anonymous access at all levels.
- 'Repository Defaults' - Allows you to change the default setting for repositories.
- A repository name - Allows you to change the setting for the specific repository.

Configuring Avatar Settings

For each user who has a FishEye/Crucible account, an avatar image is displayed next to their name. By default, each user has a unique avatar that is randomly formed from the text in their email address.
Alternatively, you can allow users to upload their own avatar image by uploading an image to an external service, such as Gravatar.

To allow avatars from an external server,

1. Go to the FishEye 'Admin Menu'.
2. Click the 'Avatar Settings' link in the left navigation column.
3. The 'Edit Avatar Server Settings' page will be displayed (see screenshot below).
4. Change 'Select Type', to 'External'.
5. In the 'Url' field, type the address of the external service, e.g. http://gravator.com/avatar/
   For additional security, it is recommended that you specify the following:

   https://secure.gravatar.com/avatar/

   When using an external avatar service, you should use the same protocol as is used to access FishEye. i.e. if FishEye uses the secure HTTP protocol then your Gravatar URL would include that prefix, as in the previous example. Conversely, if your FishEye instance uses regular HTTP, then your Gravatar URL should do the same.

6. In the 'Suffix' field, you can optionally specify parameters as supported by the external server. For example, if you wish to control the type of images to be displayed, Gravatar accepts "r=g" ("rating=general").
   The suffix must be in the form of a url query string, starting with an ampersand. For example "&r=g"
7. Click the 'Save' button.
8. Users can now upload their own images to the external server.
   The user's login name to the external server must be the email address that is specified in their FishEye/Crucible User Profile.

Screenshot: 'Edit Avatar Server Settings'
Configuring External Authentication Sources

Although FishEye always maintains a list of users internally, you can have FishEye authenticate and authorise users against one or more external authentication sources.

Be aware that you can force usernames to become lowercase on import.

Single External Authentication Source

To set an external authentication source, click 'Users/Security' on the 'Admin Menu'. Only one external authentication source can be setup and each repository has the choice of authenticating against that or the internal login. To change authentication sources, you will need to remove the settings that are already configured, by clicking the 'Remove' link. You will then be presented with the option to add a different authentication. FishEye currently supports:

- LDAP Authentication
- Host-Based Authentication - Implemented using PAM on Linux/Solaris/OS-X and Local/Domain Accounts on Windows
- AJPv13 Authentication
- Custom Authentication

Multiple External Authentication Sources

The recommended approach to authenticating against more than one authentication source is to implement single signon (SSO). You can integrate FishEye with an existing SSO solution, or add Atlassian Crowd integration to combine users and authentication from multiple external user repositories. Crowd connectors are already available for all Atlassian products, LDAP, Active Directory and Subversion:

- Crowd Authentication
- Custom Authentication

AJPv13 Authentication

AJP authentication expects requests to be pre-authenticated via an external server before arriving at FishEye.

Typically, this would be a web server (e.g. Apache) configured to perform password and role checking for a given URL. If successful, the server forwards the request to the FishEye server via the AJPv13 protocol.

FishEye Configuration

For FishEye to use AJP authentication, the following two values must be configured:

- The AJP Bind Address must be set per FishEye instance. See also Server Settings.
- The user's Auth Type must be set to 'ajp'.

Apache Configuration

Here is one example of how to configure Apache Httpd so that all requests to Apache Httpd for the path /fisheye are forwarded to a FishEye instance on the same machine with an AJP Bind Address of localhost:8009.

Add these lines to your apache configuration:
LoadModule jk_module modules/mod_jk.so

JkWorkersFile /path/to/workers.properties
JkLogFile /var/log/mod_jk.log
JkLogLevel debug
JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "
JkMount /fisheye/* worker1

Then create a file under /path/to/workers.properties and add:

worker.list=worker1
worker.worker1.type=ajp13
worker.worker1.host=localhost
worker.worker1.port=8009

Crowd Authentication

Atlassian's Crowd identity management system can be integrated with FishEye. Please see the document Integrating Crowd with FishEye in the Crowd Administrator's Guide.

Note:

- In FishEye versions 1.4 and later, support for Crowd is built in and configuration is greatly simplified.
- In FishEye versions 1.3.x and earlier, using Crowd required installing a custom authentication plugin on your FishEye server. In version 1.4, the custom authentication plugin is no longer required or supported.

⚠️ FishEye is bundled with the Crowd 1.3 client library, and therefore is intended to operate with Crowd 1.3 or later.

Custom Authentication

To implement an arbitrary form of authentication and authorisation for FishEye you need to provide a class which extends com.cenqua.fisheye.user.plugin.AbstractFishEyeAuthenticator. You can find more information about custom FishEye authorisation in the online javadocs and the library jar.

For FishEye to use the authenticator, it must be compiled, placed in a jar archive and then put in the $FISHEYE_INST/lib directory. If other third-party libraries are required by your authenticator, they must also be in the $FISHEYE_INST/lib directory.

Global Configuration
After implementing a custom authenticator, the next step is to configure FishEye to use it.

Click the 'Setup Custom authentication' link on the FishEye 'Admin' -> 'Users/Security' page.

You will be presented with a form containing the following fields to be set:

<table>
<thead>
<tr>
<th>Classname</th>
<th>The fully qualified class name of your AbstractFishEyeAuthenticator, e.g. com.cenqua.fisheye.user.plugin.ExampleFishEyeAuthenticator.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache TTL (positive)</td>
<td>How long FishEye should cache permission checks. Example values are: 0 secs, 5 mins.</td>
</tr>
<tr>
<td>Auto-add</td>
<td>FishEye can automatically create a user it has not previously encountered if the user can successfully authenticate against your authenticator.</td>
</tr>
<tr>
<td>Properties</td>
<td>Any properties your authenticator requires. These will be passed to its init() method. This field should comply with the java.util.Properties format. Example:</td>
</tr>
</tbody>
</table>

```
# comments
name1=value1
date=value2
```

**Per-Repository Constraint Configuration**

You may also require a per-repository constraint to restrict access to specific repositories using your custom authenticator. If a custom authenticator is set, then the Permissions Summary table will display the constraint per repository and a link to enable you to edit it.

The 'Authentication Test' page allows you to enter a user's credentials and to test the user's authentication. It will also test which repositories the user is authorised to access.

**Forcing Imported Usernames to be Lowercase**

When importing users from external authentication sources, you can set FishEye to force the usernames to become lowercase. This solves an issue with some sources adding duplicate users to the FishEye database.

To force lowercase usernames, carry out the following steps:

1. Log into FishEye's Admin Interface.
2. Under **Authentication settings**, the option **Force Lowercase Username** can be toggled on and off.
3. With this setting switched **On**, when new users are added to FishEye from an external authentication source, all usernames will be converted to lowercase.

**Screenshot: Forcing Lowercase Usernames**

```
Username Option

Force Lowercase Username: OFF (On)
```

**Host-Based Authentication**

Host-based authentication uses the user account mechanism of the underlying operating system on which FishEye
is running. FishEye currently supports PAM-based authentication on Linux/Solaris/OS-X, and NT-based authentication on Windows.

Group Restrictions

FishEye can be configured to check if a user belongs to a group (or groups) before allowing access.

You can list one group name, or join several group names into a boolean expression like `group1 & (group2 | group3)`.

If your group name contains spaces or non-ASCII characters, then you need to use quotes. For example: "Power Users" | Administrators.

Windows

⚠️ If you are using Active Directory, you can configure FishEye to use LDAP as an alternative to host-based authentication.

If the computer FishEye is running on is not a member of a domain, then the Domain attribute is ignored.

When the computer is a member of a domain, you need to enter the full DNS name of the domain (e.g. corp.example.com). If you enter the short version of the domain (e.g. corp), then group-based restrictions may fail.

Once you have configured your settings, we recommend you use the 'Test' function to ensure your access control behaves correctly.

PAM

On Linux, Solaris and OS-X, host-based authentication uses PAM (Pluggable Authentication Modules) to check users' passwords.

FishEye needs to be configured with the service name to use when conversing with PAM. You can create a new service name in the PAM configuration (typically `/etc/pam.conf` or `/etc/pam.d/`), or configure FishEye to use an existing service name (such as `other`, `login` or `xscreensaver`).

Some general operating-system specific tips are given below, but you should consult the PAM documentation for your operating system.

Once you have configured your settings, we recommend you use the 'Test' function to ensure your access control behaves correctly.

Linux

On many Linux distributions, you may need to create a `/etc/pam.d/fisheye` file containing:

```
auth required pam_stack.so
service=system-auth
```

Mac OS-X

On a default OS-X installation, you may need to create a `/etc/pam.d/fisheye` file containing:
Solaris

If you are using the default `pam_unix_auth` PAM configuration on Solaris, then you may need to add a line like this to your `/etc/pam.conf` file:

```
fisheye auth requisite  
pam_authtok_get.so.1
fisheye auth required  
pam_unix_auth.so.1
```

If you test this and it does not work, it is probably because when using `pam_unix_auth` on Solaris, the process doing the password check needs read access to `/etc/shadow`.

Giving the FishEye process read access to this file may solve this problem, but using permissions other than `0400` for `/etc/shadow` is not recommended. You should discuss this with your system administrators first, and possibly change to a PAM module other than `pam_unix_auth`.

Global Settings

Global settings are:

<table>
<thead>
<tr>
<th>Domain/Service name</th>
<th>Windows: the name of the domain. Leave blank to use the local computer.</th>
<th>PAM: The service name in your PAM configuration to use. If blank, <code>fisheye</code> is used.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required group:</td>
<td>The group or groups a user must belong to in order for them to be able to log in.</td>
<td></td>
</tr>
<tr>
<td>Cache TTL (positive)</td>
<td>How long FishEye should cache permission checks. Example values are: 0 secs, 5 mins.</td>
<td></td>
</tr>
<tr>
<td>Auto-add</td>
<td>FishEye can automatically create a user it has not previously encountered if the user can successfully authenticate with the host.</td>
<td></td>
</tr>
</tbody>
</table>

Per-Repository Settings

You can give FishEye a group restriction that will be used to check if a user has access to individual repositories. You can specify this per repository, or just specify it in the repository defaults:

<table>
<thead>
<tr>
<th>Required Group</th>
<th>A group (or groups) used to check if a given user can access a given repository. For example:</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cvsusers</code> &amp; <code>cvs${REP}</code></td>
<td>The <code>${REP}</code> variable is replaced with the name of the repository in question.</td>
</tr>
</tbody>
</table>
**LDAP Authentication**

This page explains the settings for LDAP authentication and their parameters.

*On this page:*

- Global Settings
- Per-Repository Settings
- Active Directory

### Global Settings

Global LDAP settings are:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>URL</strong></td>
<td>The URL of the LDAP server, e.g. ldap://localhost:389.</td>
</tr>
<tr>
<td><strong>Base DN</strong></td>
<td>The base search space for users, e.g. dc=example,dc=com</td>
</tr>
<tr>
<td><strong>User Filter</strong></td>
<td>The LDAP search for locating users, e.g. uid=${USERNAME}. The ${USERNAME} variable is</td>
</tr>
<tr>
<td></td>
<td>expanded to the username of the individual being authenticated. You can use a more complicated</td>
</tr>
<tr>
<td></td>
<td>LDAP filter to allow only a subset of users, such as: (&amp;(uid=${USERNAME})(group=fisheye)).</td>
</tr>
<tr>
<td><strong>UID Attribute</strong></td>
<td>The name of the username attribute in objects matching the filter.</td>
</tr>
<tr>
<td><strong>Email attribute</strong></td>
<td>Optional. The name of an attribute giving the user's email address.</td>
</tr>
<tr>
<td><strong>Cache TTL (positive)</strong></td>
<td>How long FishEye should cache permission checks. Example values are: 0 secs, 5 mins.</td>
</tr>
<tr>
<td><strong>Auto-add</strong></td>
<td>FishEye can automatically create a user it has not previously encountered if the user can</td>
</tr>
<tr>
<td></td>
<td>successfully authenticate against LDAP.</td>
</tr>
<tr>
<td><strong>Initial bind DN and password</strong></td>
<td>Optional. If your LDAP server does not allow anonymous bind, then you need to specify a user</td>
</tr>
<tr>
<td></td>
<td>FishEye can use to do its initial bind.</td>
</tr>
<tr>
<td><strong>Synchronise users with Crowd</strong></td>
<td>Optional. Sets whether users will be loaded from an external directory.</td>
</tr>
</tbody>
</table>

### Per-Repository Settings

You can give FishEye an LDAP filter that will be used to check if a user has access to individual repositories. You can specify this per repository, or just specify it in the Repository Defaults to have it apply to all repositories where not specified for the individual repository:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LDAP restriction</strong></td>
<td>An LDAP filter used to check if a given user can access a given repository, e.g.</td>
</tr>
<tr>
<td></td>
<td>(&amp;(uid=${USERNAME})(group=${REP})). The ${REP} variable is replaced with the name of the repository.</td>
</tr>
</tbody>
</table>
### Match Type

One of 'user' (default) or 'any'. This setting modifies the meaning of LDAP restriction.

If set to 'user', then FishEye expects the filter to match the exact DN of the current user. If it does match, the user has access to the repository. Commonly, if your user object contains the list of groups the user has access to, you would use a 'user' match.

If set to 'any', then the filter just needs to match one result for the user to have access to the repository. If your group object contains the list of UID members, then you would use an 'any' match. In such a case, the restriction filter may look like this:

```
(&(uniquemember=${USERNAME})(cn=${REP},ou=groups,ou=com)(objectClass=groupofuniques))
```

That is, return the group of which the current user is a member.

### Active Directory

To have FishEye connect to an Active Directory server, use settings such as the following:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>ldap://HOSTNAME:389</td>
</tr>
<tr>
<td>Base DN</td>
<td>DC=corp,DC=example,DC=com</td>
</tr>
<tr>
<td>User Filter</td>
<td>sAMAccountName=${USERNAME}</td>
</tr>
<tr>
<td>UID Attribute</td>
<td>sAMAccountName</td>
</tr>
<tr>
<td>Email attribute</td>
<td>mail</td>
</tr>
<tr>
<td>Initial bind DN</td>
<td>corp.example.com/Users/SomeUser</td>
</tr>
</tbody>
</table>

### Configuring Public Signup

If you enable public signup for your FishEye system, visitors can create their own FishEye user accounts via the 'Signup' link on the login screen:

```
Login Required

Username: 
Password: 
Login

Not a member? Signup for an account.
```

Public signup is disabled by default.

To enable public signup,

1. Go to the FishEye 'Admin Menu'.
2. Click the 'Security' link in the left navigation column.
3. The 'Authentication Settings' page will be displayed (see screenshot below).
4. Next to 'Public Signup', click the 'On' link.
5. Log out of FishEye and verify that the login screen now contains a 'Signup' link.
Creating a Group

There are two types of FishEye user groups:

- ‘Built-in’ groups — these are stored in the FishEye database.
- ‘External’ groups — these are stored in an external directory (e.g. Crowd), if any are configured. See Configuring External Authentication Sources.

Note that ‘external’ groups can only be created in your external directory.

To add a 'built-in' group,

1. Click ‘Groups’ on the ‘Admin Menu’.
2. The ‘Groups’ screen will be displayed (see screenshot below), showing a list of existing groups.
3. Type the name of your new group into the ‘Name’ field and click the ‘Add Group’ button at the bottom of the screen.
4. Your new group will now appear in the list of groups.

To add users to your new group, see Adding a User to a Group.
Creating a User

There are two types of FishEye user accounts:

- 'Built-in' user accounts — these are stored in the FishEye database.
- 'External' user accounts — these are stored in an external directory (e.g. LDAP), if any are configured. See Configuring External Authentication Sources.

**Note re external directories:**
- new users can only be added in FishEye if they already exist in the external directory. FishEye does not modify your external directory.
- if you have enabled 'auto-add' for your external directory, users who don't exist in FishEye will be automatically added the first time they login to FishEye.

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.
   - The user can easily change their own password later.
9. Click the ‘Add’ button.
Deleting or deactivating a User

To revoke a user’s access to FishEye, you need to delete the user’s account.

To revoke a user’s access to Crucible, you can either:

- delete the user’s account; or
- deactivate the user’s account (this has the advantage of being easy to undo if required).

Note that the number of users granted by your FishEye license may be different from the number of users granted by your Crucible license.

To deactivate a user’s Crucible account,

1. Click ‘Users’ on the ‘Admin Menu’.
2. The ‘User Browser’ screen will be displayed (see screenshot below), showing a list of users.
3. Locate the user and click the corresponding ‘Edit’ link.
   - If the user doesn’t initially appear on the screen, type part of their email address and/or select a group to which they belong, and click the ‘Filter’ button.
4. The ‘Edit User’ screen will be displayed. Deselect the ‘Active Crucible user’ check-box.
5. Click the ‘Apply’ button.

To delete a user’s FishEye (and Crucible) account,

1. Click ‘Users’ on the ‘Admin Menu’.
2. The ‘User Browser’ screen will be displayed (see screenshot below), showing a list of users.
3. Locate the user and click the corresponding ‘Delete’ link.
   - If the user doesn't initially appear on the screen, type part of their email address and/or select a group to which they belong, and click the ‘Filter’ button.
There are two types of FishEye user accounts:

- 'Built-in' user accounts — these are stored in the FishEye database.
- "External" user accounts — these are stored in an external directory (e.g. LDAP), if any are configured. See Configuring External Authentication Sources.

To edit a user's details,
1. Click 'Users' on the 'Admin Menu'.
2. The 'User Browser' screen will be displayed (see screenshot below), showing a list of users.
3. Locate the user and click the corresponding 'Edit' link.
   - If the user doesn't initially appear on the screen, type part of their email address and/or select a group to which they belong, and click the 'Filter' button.
4. The 'Edit User' screen will be displayed. You can update the following fields:
   - 'Display Name' — type the user's display-name.
     - To change the user's login name, see Renaming a User.
   - 'Email' — type the user's email address. This address is where the user will receive notifications.
   - 'Auth Type' — select either 'Built-in' or the name of the appropriate external directory where the user will be stored.
     - By changing the 'Auth Type', you are moving the user to a different directory.
5. Click the 'Apply' button.

**Screenshot: User Browser**

![User Browser Screenshot]

**Changing a User's Password**

There are two types of FishEye user accounts:

- 'Built-in' user accounts — these are stored in the FishEye database.
‘External’ user accounts — these are stored in an external directory (e.g. LDAP), if any are configured. See Configuring External Authentication Sources.

⚠️ You can only change the passwords of ‘built-in’ users.

To change a user’s password,

1. Click ‘Users’ on the ‘Admin Menu’.
2. The ‘User Browser’ screen will be displayed (see screenshot below), showing a list of users.
3. Locate the user and click the corresponding ‘Edit’ link.
4. If the user doesn't initially appear on the screen, type part of their email address and/or select a group to which they belong, and click the ‘Filter’ button.
5. The ‘Edit User’ screen will be displayed. Click the ‘Change Password’ link.
6. The ‘Change Password’ screen will be displayed. Type the new password and click the ‘Apply’ button.

Renaming a User

There are two types of FishEye user accounts:

- ‘Built-in’ user accounts — these are stored in the FishEye database.
- ‘External’ user accounts — these are stored in an external directory (e.g. LDAP), if any are configured. See Configuring External Authentication Sources.

⚠️ Renaming a user changes their login name. To change their Display Name, please see Editing a User’s Details.

To rename a user,

1. Click ‘Users’ on the ‘Admin Menu’.
2. The ‘User Browser’ screen will be displayed (see screenshot below), showing a list of users.
3. Locate the user and click the corresponding ‘Edit’ link.
4. If the user doesn't initially appear on the screen, type part of their email address and/or select a group to which they belong, and click the ‘Filter’ button.
5. The ‘Edit User’ screen will be displayed. Click the ‘Rename’ link.
6. The ‘Rename user’ screen will be displayed. Type the new username and click the ‘Rename’ button.

Granting Administrator Privileges

FishEye can now grant Admin privileges to users and groups. These exist in addition to the core ‘Admin’ account (in FishEye/Crucible 1.6 onwards).

⚠️ Be sparing in granting Admin privileges, as all Admin users have the ‘keys to the kingdom’. They can add or remove other’s user or group admin rights, and change the password of the core Admin account.

Once logged in as admin, you have the following option in the Admin screen:

Screenshot: The Administrators Menu Option
Setting Admin privileges for individual users

To set user's Admin rights, log in as the Admin user and select 'Global Settings' > 'Administrators' from the left navigation bar.

To grant a user admin rights:

- Select the username in the 'Non-Admin Users' column.
- Click the 'Add' button.
- The username appears in the 'Admin Users' column.

To remove a user's admin rights:

- Select the username in the 'Admin Users' column.
- Click the 'Remove' button.
- The username appears in the 'Non-Admin Users' column.

⚠️ Take care not to remove admin rights from your own account.

Screenshot: Setting User's Admin Settings

Setting Admin privileges for groups
To set a group's Admin rights, log in as the Admin user and select 'Global Settings' > 'Administrators' from the left navigation bar.

To grant a group admin rights:

- Select the group name in the 'Non-Admin Groups' column.
- Click the 'Add' button.
- The group name appears in the 'Admin Groups' column.

To remove a group's admin rights:

- Select the group name in the 'Admin Groups' column.
- Click the 'Remove' button.
- The group name appears in the 'Non-Admin Groups' column.

⚠️ Take care not to remove admin rights from your own account.

Screenshot: Setting Group Admin Settings

Load all users from Active Directory, LDAP or Atlassian Crowd

FishEye and Crucible can load an external user base stored in Active Directory, Atlassian Crowd, or any LDAP-compatible user repository.

1. Configuring your external source

You must firstly set up FishEye/Crucible to connect to your external user repository, either LDAP-based (including Active Directory) or Atlassian Crowd. Follow the steps in the documentation pages listed below and then return to Step 2 on this page.

- LDAP Authentication
- Crowd Authentication

⚠️ You can only have one external user repository connected to FishEye/Crucible. If you need multiple repositories,
you can use Atlassian Crowd to collate users from multiple sources and then connect FishEye/Crucible to Atlassian Crowd.

2. **Loading external users**

To load users from Atlassian Crowd:

1. From the 'Admin Menu', click 'Global Settings' then 'Security'.
2. Under Security, find the option 'Synchronise users with Crowd'. Select the 'Yes' option by clicking the radio button. Click 'Apply' to complete the process.
3. As soon as an option has been selected and 'Apply' has been clicked, users are immediately added to the user list.

⚠ Note that users with identical names to existing users will not be imported. In the Users list, you can check whether each user is from the local database or loaded externally.

3. **Setting the synchronise period**

The 'synchronise period' option allows you to set the time interval when Fisheye or Crucible will synchronise with the LDAP directory. You can use intuitive settings such as '1m' for one minute, '1h' for one hour, and so on. Simply enter the time interval and click 'Apply'.

**Understanding security**

The following flowchart shows how to determine whether a user is allowed to access a FishEye repository:

*Screenshot: FishEye Security Flowchart*
Backing Up and Restoring FishEye Data

FishEye 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 FishEye instance.

On this page:

- Backing Up FishEye Data
Back up FishEye Data

The FishEye Admin Interface Backup Process

1. Navigate to the FishEye 'Admin' screen (click the 'Administration' link in the footer of any FishEye 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 FishEye Backup Screen
The FishEye Command Line Backup Process

1. Your FishEye instance must be running during the backup.

   1. Open a command line interface on the FishEye 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 FishEye instance uses a custom FISHEYE_INST directory, make sure the environment variable is properly set when running the backup command.

Components of a FishEye Backup

The FishEye 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>
<tr>
<td>Plugins</td>
<td>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).</td>
<td>Configuration data for all plugins are backed up by default, as well as all plugins installed in FISHEYE_INST/var/plugins/user.</td>
</tr>
</tbody>
</table>
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 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. This item only applies when using Crucible with FishEye.

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 OR --quiet</td>
<td>Suppresses output</td>
<td>No</td>
</tr>
<tr>
<td>Output filename</td>
<td>-f OR --file</td>
<td>Specify a different path and filename to the FISHEYE_INST/backup/backup_YYYY-DD-MM_HHmm.zip file. When filename is omitted, the backup filename contains the date and time.</td>
<td>FISHEYE_INST/backup/backup_YYYY-DD-MM_HHmm.zip is the default</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>
</tbody>
</table>
Anonymise

-a OR --anonymise  
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).**

Cache Backup

--cache  
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.

<table>
<thead>
<tr>
<th>Command Line Examples</th>
</tr>
</thead>
</table>

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**

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

**Backup including cache data (also includes all default components)**

```bash
$ ./fisheyectl.sh backup --cache
```

**Restoring a backup with cache data (also restores all default components)**

```bash
$ ./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. This item only applies when using Crucible with FishEye.</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 FishEye 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><strong>Option name</strong></th>
<th><strong>Description</strong></th>
<th><strong>Allowed Values</strong></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>
<tr>
<td>Backup frequency</td>
<td>Sets how often the backup will take place.</td>
<td>Can be set to 'every day', 'every Sunday', 'Monday to Friday' and 'first day of the month'.</td>
</tr>
<tr>
<td>Backup time (HH:mm)</td>
<td>The time when the backup will take place.</td>
<td>Any valid 24-hour time in the format HH:mm (hours, minutes).</td>
</tr>
<tr>
<td>Include</td>
<td>Specifies which items must be included in the backups (these components are explained at the top of this page).</td>
<td>As per the options for regular on-demand backup (These components are explained at the top of this page).</td>
</tr>
</tbody>
</table>

**Screenshot: Scheduling Backups in FishEye**

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

The FishEye Data Command Line Restoration Process

There is currently no way to restore a backup from the web interface because FishEye 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.

1. Install FishEye 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 FishEye which are older than the version that created the backup.
2. Make sure the FishEye instance is not running.
3. Open a command line interface on the FishEye 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 FishEye 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</td>
<td>Restore the backup from PATH/Filename.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>PATH/Filename</td>
<td></td>
<td>(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>
</tbody>
</table>
FishEye 2.1 Documentation

**JDBC class**

| --driver OR -d | 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 `--dbtype`. (Not applicable for 'built-in'.) |

---

**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 FishEye 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 FishEye 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
```

---

**Advanced Administration Options**

- Command-Line Options
- Configuring Indexing
- Customising FishEye's Look & Feel
- Environment Variables
- Re-indexing your repository
- Tuning FishEye

**Command-Line Options**

A FishEye instance can be managed using the `fisheyectl` script. Before running this script you need to ensure that you have set the `JAVA_HOME` environment variable, or that `java` is on the path.
Unix usage:

/FISHEYE_HOME/bin/fisheyectl.sh command [options]

Windows usage:

\FISHEYE_HOME\bin\fisheyectl.bat command [options]

The command parameter can be one of run, start or stop (see below). You can also find convenience scripts for running each of these commands, such as run.sh or run.bat.

run
The run command starts FishEye. This command runs FishEye in the foreground. It does not fork a background process.

start
The start command has the same options as run, but starts FishEye in the background.

Windows: FishEye will be run in a separate cmd.exe window.
Unix: FishEye will be run with nohup and the console output will be redirected to $FISHEYE_INST/var/log/fisheye.out.

Options (for both run and start):

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--config path</td>
<td>Load configuration from the file at path. Default is $FISHEYE_INST/config.xml.</td>
</tr>
<tr>
<td>--quiet</td>
<td>Do not print anything to the console.</td>
</tr>
<tr>
<td>--debug</td>
<td>Print extra information to the debug log.</td>
</tr>
<tr>
<td>--debug-perf</td>
<td>Print performance-related information to the debug log.</td>
</tr>
</tbody>
</table>

The following options are currently available, but will be removed at a later date:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--Xtab-width nchars</td>
<td>Specifies the number of spaces to use to represent a tab character. The default is 8.</td>
</tr>
<tr>
<td>--Xdisable-dirtree-empty-checks</td>
<td>When rendering the directory tree on some pages, FishEye calculates if each directory subtree is empty. For massive repositories, this calculation can cause the page to take a long time to render. This option disables the calculation that determines emptiness.</td>
</tr>
<tr>
<td></td>
<td><strong>⚠️</strong> If you start fisheye with this flag, then empty directories will no longer show up as grey (i.e. deleted) and the option to hide them will also disappear</td>
</tr>
</tbody>
</table>
Disable the generation of a full-text index for file content. This prevents further indexing, but does not delete any existing full-text indexes. FishEye will not warn you if you specify this option but still try to do a content search. This option is useful if you do not use content search and you are finding FishEye is taking a long time to index your content.

**stop**

The `stop` command stops a running FishEye instance.

**Options:**

`--config path` Load configuration from the file at `path`. Default is `$FISHEYE_INST/config.xml`.

**fullscan**

Usage:

```
fisheyectl fullscan [options] [repname ...]
```

The `fullscan` command requests a full scan of the given repositories, or all repositories if no repository name is given.

**Options:**

`--config path` Load configuration from the file at `path`. Default is `$FISHEYE_INST/config.xml`.

**rescan**

Usage:

```
fisheyectl rescan [options] repname start end
```

Requests a rescan of the given repository between two specified revision ids.

**Note:** this operation is not supported by CVS repositories.

**Options:**

`--config path` Load configuration from the file at `path`. Default is `$FISHEYE_INST/config.xml`.

**reindex**

Usage:

```
fisheyectl reindex [options] [repname ...]
```

Requests a full reindex of the given repositories, or all repositories if no repository name is given.

**Options:**

`--config path` Load configuration from the file at `path`. Default is `$FISHEYE_INST/config.xml`. 
scannow

The `scannow` command requests an incremental scan from the command line.

Usage:

```
fisheyectl scannow -s [repname ...]
```

Options:

- `s`  Run synchronously; i.e. the command will not terminate until the requested indexing operation is complete.

  - You can set the poll period to 'Never' in which case the repo will be viewable from the web UI but the repository will not be polled.
  - Repositories may be polled in the demand using this code:

    ```
fisheyectl.sh scannow [repname ...]
    
    This will perform a one-time indexing operation.
    
    Note: scannow can also be used on a repository which has a poll period.
    
backup

Usage:

```
fisheyectl backup [filename]
```

The `backup` command creates a zip archive containing important FishEye configuration files.

Options:

- `filename`  Store the backup.zip to `filename`. Default is `$FISHEYE_INST/backup/backup_yyyyddMMHhmss.zip`.

### Configuring Indexing

#### Understanding How FishEye Indexes Repositories

FishEye's administration interface now allows you to configure the number of threads used for repository indexing. FishEye uses two types of indexing operation:

- **Initial indexing** - the complete scan of the repository that FishEye does when FishEye first accesses the repository, and
- **Incremental indexing** - the quick, background process that regularly scans repositories for changes.

The number of threads available for these indexing operations is now configurable.

Prior to FishEye 1.5, a transient error during processing the initial indexing operation would cause a repository to move to the incremental indexing thread, even when a large amount of indexing remained. This would impact the incremental indexing of other repositories in the FishEye instance. In FishEye 1.5 this has been corrected. A repository will remain on the initial indexing thread until it has successfully completed the initial indexing operation.

#### Setting FishEye's Total Threads

To configure the number of threads FishEye uses for indexing, open the FishEye 'Admin Screen', then click 'Server'
Settings'. Now, click 'Edit Details' under 'Resource Limits'. You can increase the number of threads available for each indexing phase independently. Enter the desired number of threads for each process and Click 'Update'.

Screenshot: FishEye Multi-threading Configuration

<table>
<thead>
<tr>
<th>Edit Server Resource Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Initial Index Threads:</td>
</tr>
<tr>
<td>Max Incremental Index Threads:</td>
</tr>
</tbody>
</table>

Choosing a Thread Setting

As changing the number of threads always involves a trade-off in performance, you should generally change these settings in increments of one, then observe the performance impact. A safe range is from 1-3 threads on both settings. Using more than three threads on either setting is not recommended; the more threads, the busier the server will become when indexing. There is certainly no benefit in having more threads than you have repositories. Adding a lot of threads may impact overall system performance.

By default, FishEye uses one thread each for the two indexing modes. This is equivalent to the functionality in FishEye prior to version 1.5.

For more information on managing FishEye's performance, see the FishEye Tuning page.

⚠️ This feature was introduced in FishEye 1.5.

Customising FishEye's Look & Feel

FishEye enterprise license users have access to the HTML/JSP source of FishEye and can customise FishEye's look and feel.

FishEye Source Edition

To use custom HTML/JSP content, you must be using a build of FishEye that contains the JSP source. These builds are named fisheye-1.x.y-jspsource.zip instead of the normal fisheye-1.x.y.zip bundle.

If you have a commercial license assigned to your account, you will see a 'source download' option on your download page.

Customising Content

You can modify any of the files in FISHEYE_HOME/content/. However we strongly recommend that you use separate FISHEYE_HOME and FISHEYE_INST directories (as described in the Installation Guide), and that you store your modified files in FISHEYE_INST/content (If you use FISHEYE_INST/content, you only need to keep your modified JSP/HTML files in that directory. This simplifies your upgrade process).

When you make changes to content, your changes should appear when you next refresh the page in your browser. If they do not, then log into the FishEye Administration screens, click 'Content' on the 'Admin Menu' and follow the instructions on the 'Content' page.

Screenshot: Content Page
Environment Variables

Environment variables are system-wide settings that are required for certain applications. Instructions on Setting Environment Variables are here.

The following is a list of the environment variables used by FishEye.

**JAVA_HOME**

The JAVA_HOME environment variable is used by FishEye to select the Java Virtual Machine (JVM) to be used to run FishEye. If this environment variable is not set, FishEye will use whatever Java executable is available on the path. In Linux systems, this may sometimes be GCJ-based which causes some problems running FishEye.

See the instructions on setting JAVA_HOME.

**FISHEYE_OPTS**

FishEye uses the FISHEYE_OPTS environment variable to pass parameters to the Java Virtual Machine (JVM) used to run FishEye. This is typically used to set the Java heap size available to FishEye (see Fix out of Memory Errors). With a Sun JVM, for example, you would use:

```
FISHEYE_OPTS=-Xmx1024m -XX:MaxPermSize=128m
```

This would give FishEye a max of 1024 MByte heap, a Max permanent generation size of 128m. See Tuning FishEye for more information.

After having set the FISHEYE_OPTS and restarting your server, go to Administration > Sys Info/Support > System Info, and check your JVM Input Arguments to ensure that your server is picking up your FISHEYE_OPTS as expected.

**FISHEYE_ARGS**

FISHEYE_ARGS are the arguments which will be passed to FishEye when it is started. You can set this to --debug, for example, or --debug-perf if you always want to have FishEye debugging put into the FishEye log files.

**FISHEYE_LIBRARY_PATH**

The FISHEYE_LIBRARY_PATH environment variable tells FishEye where it should look to load any additional native libraries.

**FISHEYE_INST**

The FISHEYE_INST variable tells FishEye where to store its data. If you wish to separate FishEye's data from its application files in FISHEYE_HOME, you should use this variable.

Read more about using FISHEYE_INST in the Installation Guide.
Setting Environment Variables under Windows XP

(If you are running Fisheye as a windows service you need to refer to the instructions here)
(Linux instructions are here)

1. Click 'Start' > 'Control Panel' > 'System'.

Screenshot: System Properties under Windows XP Control Panel

2. Click the 'Advanced' tab.
3. Click the 'Environment Variables' button.

Screenshot: Environment Variables under Windows XP Control Panel
4. Click ‘New’.
5. In the ‘Variable name’ field, enter the name of the environment variable, for example

   FISHEYE_OPTS

6. In the ‘Variable value’ field, enter the setting as required. This may be quite cryptic, for example the default value for FISHEYE_OPTS is this:

   -Xmx256m
7. Restart the computer.

**Setting Environment Variables for Windows Services**

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.

```plaintext
# JDK 1.5 Additional Parameters for jmx
wrapper.java.additional.4=-Dcom.sun.management.jmxremote
wrapper.java.additional.5=-Dcom.sun.management.jmxremote.port=42...
wrapper.java.additional.10=-Dwrapper.mbean.name="wrapper:type=Java Service Wrapper Control"
```

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:

```plaintext
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:

```plaintext
# Initial Java Heap Size (in MB)
wrapper.java.initmemory=32

# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=256
```

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).

**Setting Environment Variables under Linux or UNIX based platforms**

There are a number of ways to set environment variables on Linux or UNIX based systems (including Mac OS X). Here are just two:

**For your current user:**

1. Open up a shell or terminal window
2. Type this command:
For all users in the system:
1. Open up a shell or terminal window
2. vi /etc/profile (replace vi with your favourite text editor)
3. Add export (variable name)=(variable value) on its own line at the end of the file
4. Save, and restart your shell

If you are using a GUI, you may not need to open up the shell. Instead, you might be able to open the file directly in a graphical text editor.

If you are experiencing memory errors in FishEye, see Fix Out of Memory errors by increasing available memory.

Re-indexing your repository
This page contains instructions on how to re-index your repository in FishEye.

Re-indexing involves FishEye doing an intensive scan of the repository contents, so it can quickly show the repository data via FishEye's web interface. Re-indexing may be required in a number of situations such as server maintenance, changes in your repository, major setting changes and upgrades to FishEye.

Due to the complexity of SCM repositories, an indexing scan may take many hours or even days to complete. During this time, your FishEye users may be inconvenienced as they will not by able to view the repository being indexed. As a result, Crucible users will not be able to carry out reviews unless you have Crucible set to store all revisions offline.

Performing a Manual Re-index

vi ~/.profile

export (variable name)=(variable value)

export FISHEYE_OPTS=-Xmx256m

Add this command on its own line at the end of the file.
4. Save, and restart your shell.
To re-index your repository,

1. Open the 'Admin Menu', by clicking the 'Administration' link in the footer of any FishEye page.
2. The 'Repository List' screen opens by default. In the main window, the list of configured repositories is shown. Click 'View' next to the desired repository, then click on maintenance.
3. Select one of the re-indexing options discussed below.
4. The repository will be re-scanned, a process that may take hours or possibly days, depending on the size of the repository, network speed, machine performance and other factors.

How do I mitigate long re-index times when upgrading? See the FAQ page on this topic.

**Indexing Settings**

Settings and administrative operations for repository indexing are located on the Indexer page.

To get to this page, go to 'Administration' > 'Repository List' > 'View' (next to your repository name) > 'Indexer' > 'Maintenance'

The 'Indexer Maintenance' screen opens, where an administrator can manually trigger the following actions:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh</td>
<td>Refreshes the status string, in the Status box. This will show the name of any repository task that FishEye is currently carrying out (if any). This may be helpful to monitor the progress of an initial indexing task or to diagnose problems (such as your repository server being out of action).</td>
</tr>
<tr>
<td>Re-index Repository</td>
<td>Delete the current cache and re-index the repository from the beginning. This action will also restart the repository.</td>
</tr>
<tr>
<td>Re-index Crucible Data</td>
<td>(applies only when using Crucible with FishEye) Re-index all the Crucible review data associated with the current repository.</td>
</tr>
<tr>
<td>Re-index Linecount Data</td>
<td>Re-index the linecount data used to generate the LOC (Lines Of Code) charts. The linecount data will be recalculated in daily buckets based on the server timezone.</td>
</tr>
<tr>
<td>Scan Now</td>
<td>Run a repository scan (also referred to as indexing) now. If the repository has already been indexed, it will be an incremental scan, otherwise an &quot;initial&quot; scan. This is especially useful if you have not set the repository to poll automatically, or it is set with a long poll (interval) period.</td>
</tr>
<tr>
<td>Rescan Revisions</td>
<td>For a Subversion repository, you can rescan non versioned properties (revprops). In Subversion it is possible to enable non-versioned properties (e.g. the check-in comments) to be updated by committers. When this happens, FishEye will not automatically pick up the updates. By rescanning specific revisions, FishEye will rescan the non-versioned properties and amend the entry in FishEye accordingly. For a Perforce repository, you can rescan changelists. This will rescan changelist metadata (author, date, commit message) and any job fixes associated with changelists in the the given changelist range.</td>
</tr>
</tbody>
</table>

*Screenshot: FishEye Index Maintenance menu*
Tuning FishEye

This page contains information about improving FishEye's performance. On this page:

- Configurable Indexing Threads
- Java Heap Size
- Improving Initial Scan Performance
- Background Information
- General Improvements
- Improve Update Performance during Initial Scan
- Improving Initial Scan performance for an SVN Repository
- Performance Support

Configurable Indexing Threads

FishEye is now multi-threaded, allowing you to control the number of threads dedicated to the repository indexing process. See the page on Configuring Indexing.

Java Heap Size
The heap size of the FishEye Java Virtual Machine is controlled by the FISHEYE_OPTS environment variable. The best heap size to use is dependent on a number of factors including:

- The source code management (SCM) system being used. Subversion scanning typically uses more memory than CVS, for example.
- The complexity of operations in the repository. Processing changesets which affect many files will use more memory.
- The amount of physical RAM in the system. If the Java heap is too large, it may induce swapping which will impact performance.

FishEye will reserve a portion of the available heap for caching of database data. So in general, the more memory you can supply, the better.

For Subversion repositories, it is also possible to reduce FishEye's memory footprint by reducing the BlockSize parameter.

If you do run into 'Out of Memory' errors, see Fix out of Memory Errors.

*Improving Initial Scan Performance*

This page contains information about improving the performance of FishEye repository scans.

*Background Information*

When you add a repository, FishEye needs to perform a once-off scan through the repository to build up its initial index and cache. This scan can take some time. Until this scan is complete, you may find that some data is not displayed. As a guide, FishEye should be able to process about 100KB-200KB per second on an averaged-size PC. If FishEye is accessing the repository over the network (e.g. over a NFS mount), then you should expect the initial scan to take longer.

*General Improvements*

You can increase the speed of your scans using the following options:

- If your repository is non-local, set up a local repository mirror on the FishEye server. This will provide a major speed boost for anyone scanning a repository across a network.
- Exclude unused file types, unused directories and specific large files from FishEye.

*Improve Update Performance during Initial Scan*

One option is break large repositories into multiple smaller repositories. While this technique will not improve the overall initial scan time, it allows for all fully scanned repositories to be updated while the initial scan is still being performed on those remaining.

In FishEye 1.3.4 and later, the initial and incremental scans happen in separate, single threads. So, splitting the repositories will allow incremental scans to run concurrently alongside the initial scans. You may also wish to split projects into separate repositories, since permissions in FishEye are applied on a per-repository basis.

*Improving Initial Scan performance for an SVN Repository*

The http/s protocol has the slowest performance during the initial scan. The svn protocol (svn://) is faster and the file protocol (file:///) is the fastest.

Therefore if you find your initial scan takes an extended amount of time (in some cases weeks), you should consider switching over from the http/s protocol to the svn or file protocol to define the location of your SVN repository. (Use svnsync to mirror the repository onto the fisheye server, so that you can access it with the file protocol.)
E.g. Switch from
https://example.com/svn/project/
to
svn://example.com/svn/project/
or
file:///home/user/some/location/svn/project

In order for SVN protocol to work you need to have set up an svnserv based server.

**Performance Support**

If you have implemented at least one of the suggestions above but are still experiencing slow performance, ask us for help:

1. First read the Tuning Fisheye document.
2. Turn on debug logging using the command line debug flag.
3. Allow FishEye to continue its initial scan overnight.
4. Create a new support request in the FishEye project, including your server environment and log files with the problem description.

**Managing Plugins**

A FishEye plugin is a program that provides an extra piece of FishEye functionality.

You can create your own plugins (see the FishEye Development Hub), then install them into your FishEye system as follows:

**To add a plugin,**

1. Copy the JAR file to the /FISHEYE_INST/var/plugins/user directory. You can see the absolute path name of this directory under the plugins list.
2. Access the 'Admin Menu' and click 'Plugins'.
3. The 'Plugins' screen will be displayed (see screenshot below). Click the 'Check for new plugins in...' link.
4. Your newly added plugin will now appear in the list on the 'Plugins' screen.

**To enable a plugin,**
1. Click 'Plugins' on the 'Admin Menu'.
2. The 'Plugins' screen will be displayed (see screenshot below).
3. Locate the plugin you wish to enable, and click the 'Enable' link.
   - To view, enable or disable individual modules within a plugin, click the '+' sign at the left of the plugin's name.

To configure a plugin,

1. Click 'Plugins' on the 'Admin Menu'.
2. The 'Plugins' screen will be displayed (see screenshot below).
3. Locate the plugin you wish to enable, and click the 'Configure' link. This will show the configuration page for that plugin. If the plugin does not require configuration there will not be a 'Configure' link.
4. Read the documentation for that particular plugin to understand the process of configuring the plugin.

Some plugins require further configuration after they are enabled.

**Screenshot: Plugin Management Screen**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>State</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>File System SCM</td>
<td>An SCM provider for the local file system</td>
<td>Enabled</td>
<td>All modules enabled</td>
</tr>
<tr>
<td>Confluence SCM</td>
<td>An SCM provider for Confluence instances</td>
<td>Enabled</td>
<td>All modules enabled</td>
</tr>
</tbody>
</table>

Trusted Applications

A 'trusted application' is an application that FishEye will allow to access specific functions in FishEye, on behalf of any user — without the user logging in to FishEye. Trusted applications is a new feature in Crucible 1.2.2 and FishEye 1.4.2. At this time, JIRA 3.12 and Confluence 2.7 can be configured as trusted applications.

Note: FishEye and Crucible share the same trusted applications — an application trusted by FishEye is also trusted by Crucible.
Before you begin, note that configuring a trusted application requires the transmission of sensitive data. To prevent 'man-in-the-middle attacks', it is recommended that you use an encrypted SSL connection while configuring a trusted application.

Adding a Trusted Application

To add a trusted application to FishEye:

1. Access the 'Administration Screen'.
2. Click 'Trusted Applications' under 'Global Settings' on the left navigation bar.
3. Click 'Add a Trusted Application'. The 'Trusted Application' screen opens.

On this page, there are two areas, the 'Identification' area and the 'Access Permissions' area.

Configuring Identification Settings

Under the 'Identification' heading, there are two fields, 'URL' and 'Id'.

URL field

In this field is where you will enter the Trusted Application Public Key URL of the application you wish to trust. For example, if your application's base URL is;

'http://www.mycompany/jira/

you would enter that into the URL field. Once you've done this, click the 'Get Id' button. FishEye will then retrieve the
Trust Certificate Id from the other application and display it in the Id field. If this step fails, you may not have not entered the correct URL for the other application.

**Id field**

This field contains the Trust Certificate ID, once you have filled out the URL field correctly (see above) and clicked the Get Id button. The contents of this field are not editable.

(Note: The application you are trusting must support Trusted Applications also. JIRA 3.12 and Confluence 2.7 support this.)

**Configuring Access Permissions**

Under the Access Permissions heading, there are three fields, URL Patterns, IP Address Patterns and Certificate Timeout. These allow you to further restrict requests from a trusted application.

**URL Patterns field**

With this field, you can limit the access a trusted application has to FishEye. It is not necessary to specify anything for this field; in fact a blank value is a sensible default. The default behaviour is no restriction.

The text that you specify should not include your hostname, IP address or port number, rather it relates to folders on the server, that start with the text you provide.

For example, if you use this setting:

```
/foo
```

then FishEye will trust only the requests to FishEye URLs starting with /foo, e.g. /foo/bar, /foobaz and /foo/bar/baz/x. You can specify multiple URLs by separating them with a comma.

URL Patterns do not support wildcard characters or regular expressions in FishEye.

**IP Address Patterns field**

With this field, you can limit the trusted network addresses for other applications. You can use wildcards to specify a number range, and multiple addresses can be separated with commas. For example, if you use this setting:
then FishEye will only trust requests from machines with the IP addresses 192.168.*.*.* anything.anything (a group of network addresses) and 127.0.0.0 (a single host). The default is no restriction.

Certificate Timeout field

With this field, you can set the number of milliseconds before the certificate times out. This feature's purpose is to prevent 'replay attacks'. For example, if an attacker intercepts a request, they may attempt to extract the certificate and send it again independently. With the certificate timeout, the application will be able to tell that this is no longer a valid request. The default value is 1000 (one second).

- A shorter time out is more secure, but if set too short, it may cause valid requests to be rejected on slower networks.

Once you've finished entering the settings for the Trusted Application, click Save to confirm and activate the trust relationship.

Editing Trusted Application Settings

Once you have configured your trusted application(s), you can view the settings on the main 'Trusted Applications' page.

Screenshot: Trusted Applications list

<table>
<thead>
<tr>
<th>Name</th>
<th>Id</th>
<th>Url Patterns</th>
<th>IP Address Patterns</th>
<th>Edit</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://localhost:8080">http://localhost:8080</a></td>
<td>C0A88FE614C7B635G4648A5D01359D8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a Trusted Application</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

From this screen, you can click 'Edit' to make changes to the trusted application settings, or click 'Delete' to remove the trust relationship for that application.

Customising the Welcome Message

To customise the welcome message, access the administration page, and click 'Customize Welcome & Support Message' under 'Global Settings' on the left navigation bar.

The 'Customize Welcome & Support Message' page opens.

On this page, you can provide your own custom text for the FishEye 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: FishEye Customize Welcome and Support Messages

Restoring the default messages

To revert to the default Welcome or Support messages, simply delete the text shown in the text box and click the respective '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 /FISHEYE_HOME/ folder.

To do this, simply add the following XML tags to config.xml:
## Customising Email Notifications

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

### Editing FishEye Email Templates

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

They templates are only for changing the appearance and order of certain content inside the messages.

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

### Editing the Subject Line

1. Open the `fisheye-mail-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. `fisheye-mail-subject.ftl` is used as the subject template for all FishEye email notifications.
3. Save and close the file.
4. Restarting FishEye will activate the new templates.

### Editing the Header

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

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

**Editing the Footer**

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

1. Open the 'fisheye-mail-footer.ftl' template file from FISHEYE_HOME/templates/ in a text editor. 'fisheye-mail-footer.ftl' is used as the footer template for all FishEye email notifications.
2. Add your new footer content.
3. Save and close the file.
4. Restarting FishEye 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 FishEye 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 FishEye 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.

**FishEye Email Template File List**

The following template files for FishEye notifications are stored in the FISHEYE_HOME/templates/ folder.

<table>
<thead>
<tr>
<th>Template filename</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>fisheye-mail-subject.ftl</td>
<td>Subject template</td>
</tr>
<tr>
<td>fisheye-mail-header.ftl</td>
<td>Header template</td>
</tr>
<tr>
<td>fisheye-mail-footer.ftl</td>
<td>Footer template</td>
</tr>
<tr>
<td>changeset-mail-html.ftl</td>
<td>HTML email template</td>
</tr>
<tr>
<td>changeset-mail-plain.ftl</td>
<td>Plain text email template</td>
</tr>
</tbody>
</table>

See also Customising Crucible Email Notifications.

**Freemarker Data Model for Email Templates**

*Customising FishEye email templates with Freemarker*

This page lists the Freemarker data-model for FishEye email templates. See the Freemarker documentation for instructions on Freemarker syntax. Use the templates that ship with FishEye as a guide to the properties available on each object.
These templates are used to send both batch (e.g. daily) and immediate emails. The template has access to the `changesets` variable which contains the list of changesets to send.

The default FishEye email templates make use of various data model objects, listed below.

Here is a simple example that prints out each revision in each changeset.

```
[#list changesets as cs]
${cs.id}
Author: ${cs.author}
Comment: ${cs.comment}
Files:
[#list cs.revisionInfos as rev]
   ${rev.path} ${rev.revision}
[/#list]
[/#list]
```

**Primary Data Model Objects**

<table>
<thead>
<tr>
<th>Object name</th>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>repname</td>
<td>The name of the repository</td>
<td></td>
</tr>
<tr>
<td>siteurl</td>
<td>Base URL of the FishEye instance</td>
<td></td>
</tr>
<tr>
<td>timezone</td>
<td>The time zone as configured in admin</td>
<td></td>
</tr>
<tr>
<td>watchpath</td>
<td>The path for this watch</td>
<td></td>
</tr>
<tr>
<td>changesets</td>
<td>A list of changesets</td>
<td></td>
</tr>
</tbody>
</table>

The syntax to use the data model object `repname` as an example, is as follows:

```
${repname}
```

**Changeset objects**

The `changesets` list will contain multiple changesets for batch (e.g. daily) notifications and one element for immediate notifications.

These changeset objects have the following properties:

<table>
<thead>
<tr>
<th>Object name</th>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>comment</td>
<td>The commit comment</td>
<td>Belongs to a changeset</td>
</tr>
</tbody>
</table>
For example, to iterate through all the changesets notifications, you would use the following:

```
[#list changesets as cs]
 ${cs.id}  ${cs.author}
[/#list]
```

### Revision objects

<table>
<thead>
<tr>
<th>Object name</th>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>The path of the file</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>revision</td>
<td>The revision number</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>binary</td>
<td>Boolean indicating whether file is binary</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>dead</td>
<td>Boolean indicating whether file is deleted</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>move</td>
<td>Boolean indicating whether file is moved</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>copy</td>
<td>Boolean indicating whether file is copied</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>added</td>
<td>Boolean indicating whether file is added</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>linesAdded</td>
<td>Number of lines added</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
<tr>
<td>linesRemoved</td>
<td>Number of lines removed</td>
<td>Belongs to a revisionInfo, under a changeset</td>
</tr>
</tbody>
</table>

### Software Update Notifications

FishEye (and Crucible) can detect and notify you of new versions of the program. In the Admin screen, you can set a background thread to poll Atlassian's servers and report when a new version of FishEye or Crucible has been released.

To use the Update Notification,
1. From the 'Admin Menu', click 'Server Settings' from the left navigation bar.
2. The 'Software Status' line displays either 'Up to Date', 'New Version Available' or 'Unknown'.
3. The 'Last Checked' line displays the date when the feature last checked for a new version.
4. Clicking the 'Check Now' link will immediately check for a new version.
5. Clicking the 'Enable' link allows you to switch on regular polling of the Atlassian servers, which will do periodic checking for the release of new versions. The default interval between checks is one day.
6. Once enabled, click 'Edit config' to adjust the settings. You can set an interval of your choosing. The minimum interval allowed is one minute.

⚠️ A status of 'Unknown' is displayed when the feature has not yet been activated.

**Screenshot: Settings for Update Notification**

![Update Notifications](image)

**Screenshot: Changing the Update Notification Interval**

![Edit Software Update Notifications](image)

**Contacting Support directly via FishEye**

You can contact Atlassian support directly from the FishEye Admin interface.

On the left navigation bar, click 'Sys-Info/Support'.

**Screenshot: The Sysinfo/Support Menu Option**

![System Info/Support](image)

The System Info/Support page loads. On this page, you can fill out a web form which will automatically send an email to Atlassian Support, attaching your FishEye logs and configuration file (if you wish).

⚠️ This functionality requires that the FishEye web server is already set up and capable of sending email.

**Screenshot: The System Info/Support page**
Fields in the Support Request form

On the **Support Request** form there are a number of fields to fill out and options to select.

- **Subject**: Enter a one line summary of the problem.
- **Priority**: Choose from Low, Medium, High or Critical.
- **Description**: Type a detailed description of the problem you are trying to solve.
- **Existing Support Request**: Leave this field blank to create a new support request. If you have an existing support ticket open at support.atlassian.com, enter the issue key here. This will append this request to the existing ticket.
- **Contact Name**: Your contact name.
- **Contact Number**: Your contact number.
- **Attach FishEye Logs**: Tick this box to send Atlassian your FishEye log file.
- **Attach Config File**: Tick this box to send Atlassian your FishEye configuration file.

When you have filled out the required fields, click **Send Support Request** to finish.

⚠️ Note that the form may take several minutes to fully submit, as it takes some time to export and process the log file data.

**Running Scheduled Events**
JIRA Integration in FishEye

This page contains instructions for setting up JIRA integration.

JIRA is Atlassian's issue tracking product, which can be used to manage projects and associated work. JIRA issues can be viewed in the main Dashboard view. This requires you to enter details on the required JIRA server(s) via the administration screens.

⚠️ Before you begin: Ensure that you configure your JIRA instance to enable sub-tasks and allow Remote API access; it is also recommended that you enable unassigned issues.

The instructions on this page have been tested with JIRA 4.0.0.

On this page:

- Opening the Administration Screen for JIRA Integration
- Adding a New JIRA Server
- 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
- Known Issues
- See Also

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 described below.

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 base URL of the JIRA server, e.g. {{ <a href="http://jira.atlassian.com">http://jira.atlassian.com</a> }}</td>
<td>Yes</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 in the Activity Streams.</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 the FishEye documentation and JIRA documentation on trusted applications. Known Issues.</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'.

**Editing Default JIRA Server Mappings**

This setting has two effects; it enables the feature that shows JIRA information in a dynamic window when you hover the mouse over a JIRA issue key; it will also turn every issue key into a hyperlink to that issue.

To enable this feature, click 'Edit Default JIRA Server Mappings' from the View JIRA Servers page. The 'Map JIRA Project Default' page opens.

**Screenshot: The Default JIRA Server Mappings Page**

On this page, select the FishEye Repositories or Crucible Projects that you wish to associate with all the JIRA servers that you have configured for use in FishEye/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**

![Screenshot](image)

**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.

**Known Issues**

If you elect to use Trusted Applications for authentication with your JIRA server, activity streams 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 JIRA Server configuration page in the FishEye administration section.

We are working towards supporting Trusted Applications for issue key linking and project mapping. If this issue is important to you, please vote for CRUC-1910.

**See Also**

- The JIRA documentation on Integrating JIRA with FishEye, which enables you to view FishEye data from within JIRA.
- The Crucible documentation on JIRA Integration in Crucible, which enables you to view JIRA data from within Crucible.

**Migrating to an External Database**

This page contains instructions on migrating your FishEye database from its default embedded form to an external database. This may be useful for the following reasons:

- **Improved Protection Against Data Loss**: The FishEye 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 FishEye instance, running the database on the same server as FishEye may slow it down. When using the embedded database, the database will always be
FishEye and Crucible offer alternatives to the built-in HSQLDB database for storing its relational data. At the time of writing, MySQL and PostgreSQL are supported (see System Requirements for version numbers). This page outlines the steps required for switching to an external database.

**Migrating to MySQL**

To switch from the built-in HSQLDB database to MySQL, install MySQL and follow the steps below.

1. Download the MySQL JDBC driver .JAR file from the MySQL website and copy the .JAR file to your FISHEYE_INST/lib directory (create the lib/ directory if it doesn’t already exist). Restart FishEye or Crucible to have it pick up the driver.

2. Create a UTF-8 Database:

   ```
   CREATE DATABASE fisheye CHARACTER SET utf8 COLLATE utf8_bin;
   ```

3. You will also 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 [mysqlld]:

   ```
   [mysqlld]
   default-character-set=utf8
   ```

4. Use the status command to verify database character encoding information:

   *Screenshot: Using the MySQL Status Command*
5. 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 FishEye and MySQL run on the same machine (accessible through `localhost`), issue the following commands (replacing username and password with the appropriate values):

```
mysql> grant all on fisheye.* 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)
```

6. With the database prepared, navigate to the 'Database Configuration' section in the admin interface, select MySQL 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
If this fails, verify that you have the MySQL JDBC driver .JAR file in the classpath (by placing the .JAR file in FISHEYE_INST/lib). Also, ensure that the database user can log in to the database from the machine that Crucible or FishEye is running on and that all the required privileges are present.

7. 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, FishEye 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.

Screenshot: Migrating the Database
Migrating to PostgreSQL

To switch from the built-in HSQLDB database to PostgreSQL, install PostgreSQL and follow the steps below.

1. Download the PostgreSQL JDBC driver .JAR file from the PostgreSQL website and copy the .JAR file to your FISHEYE_INST/lib directory (create the lib/ directory if it doesn’t already exist). Restart FishEye or Crucible to have it pick up the driver.

2. Create a new database user (replacing 'username' and 'password' with the appropriate values):

```bash
$ psql
> create user username password 'password';
```

3. Create a UTF-8 database and make the newly created user the owner:

```bash
> create database fisheye ENCODING 'UTF-8'
OWNER username;
```

4. Make sure the user has full access to the database:

```bash
> grant all on database fisheye to username;
```

5. With the database prepared, navigate to the 'Database Configuration' section in the admin interface, select PostgreSQL from the drop down and fill in the details for database URL, username and password.

Then click 'Test Connection' to verify that Crucible or FishEye can log in to the database.
If this fails, verify that you have the PostgreSQL JDBC driver .JAR file in the classpath (by placing the .JAR file in FISHEYE_INST/lib). Also, ensure that the database user can log in to the database from the machine that Crucible or FishEye is running on and that all the required privileges are present.

6. 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, FishEye 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.

Support for other Databases

FishEye and Crucible currently ship with support for MySQL and PostgreSQL as external databases (see System Requirements 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.

FishEye Release Notes

⚠️ FishEye 2.1 has now been released. Read the Release Notes.

FishEye Release Notes and Changelogs

- FishEye Release Summary
- FishEye 2.1 Release Notes
  - FishEye 2.1 Changelog
- FishEye 2.0 Release Notes
  - FishEye 2.0 Upgrade Notes
  - FishEye 2.0 Changelog
- FishEye 2.0 Beta Release Notes
  - Upgrading to the FishEye 2.0 Beta
  - JIRA Integration in FishEye 2.0 Beta
  - Git Alpha in FishEye 2.0 Beta
  - FishEye 2.0 Beta Reviewer’s Guide
- FishEye 1.6 Release Notes
  - FishEye 1.6 Changelog
- FishEye 1.5 Release Notes
  - FishEye 1.5 Changelog
- FishEye 1.4 Release Notes
  - FishEye 1.4 Changelog
- FishEye 1.3 Release Notes
  - FishEye 1.3 Changelog
• For changes prior to 1.3, see:
  • 1.2.x Changelog
  • 1.1.x Changelog
  • 1.0.x Changelog

Installation
You can now download FishEye from here. Information on installing FishEye can be found here.
If upgrading from a previous version, please follow the Upgrade Guide.

FishEye 2.1 Release Notes
12 November 2009

For details on minor releases since FishEye 2.1, see the FishEye Changelog.

Atlassian presents FishEye 2.1
FishEye 2.1 adds new search options, streamlined JIRA integration and runs significantly faster.

Highlights of this release:

• Wiki Markup in Commit Messages
• Streamlined JIRA Integration
• FishEye Admin API
• History Page Performance Increases
• ClearCase Support Now in Beta
• Numerous improvements and bug fixes

Thank you for your interest in FishEye 2.1.

See the documentation on Upgrading to this version.

Installing FishEye 2.1
You can now download FishEye 2.1 from here. See the documentation on Upgrading to this version.

Highlights of FishEye 2.1

1

Wiki Markup in Commit Messages
You can now use Atlassian Wiki Markup in your commit messages — **bold text**, *italics*, `code`, links, images and much more — FishEye will render it. Crucible now has Wiki Markup Rendering as well.

**Screenshot: Wiki Markup in Activity Stream**

<table>
<thead>
<tr>
<th>Seb Ruiz</th>
<th>committed 47351 to FE.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUC-1203: Rework from CR-FF-2708. Nic suggested checking against the <code>dom</code> object, not the <code>jquery</code> wrapper.</td>
<td></td>
</tr>
</tbody>
</table>
| We should remove all of `global.js`:

1. This is old code, most of which needs to be replaced to meet our newer javascript standards:
   1. Namespaceing
   2. Needs to be `jQueryified`
   3. Doesn’t do proper error reporting |

### Streamlined JIRA Integration

The JIRA hover feature has been improved, now resolving JIRA user names as they appear in FishEye. The integration has also been updated to support JIRA 4.

**Screenshot: Streamlined JIRA Integration**

### FishEye Admin API

The FishEye administration API has been improved. It is now possible to let plugins control repositories, for example adding new repositories, starting, stopping and so on. This work will be followed by a REST layer to also make this
available to remote clients and scripts. We also added Atlassian Plugin SDK support to FishEye and Crucible, 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: New Development Documentation Space*

**A map of FishEye and Crucible technology**

This image shows how the various components of FishEye and Crucible interact with each other:

- **Web Item Plugins**: display extra links and tabs in the UI
- **Plugins**: interact with FishEye/Crucible via our Java API
- **Gadgets**: extend the UI to OpenSocial Gadget containers like JIRA and iGoogle
- **External Systems**: your external systems use our REST API to query and control FishEye/Crucible

*Key: In this diagram, Green items are your code. White items are FishEye/Crucible components.*

**History Page Performance Increases**

The History page’s now uses “lazy loading” - getting only those features, such as revisions and commit log messages, that can initially be seen by the viewer and then building out the rest of the page contents as they’re required. New pagination options were added giving access to the first and last pages of information, in addition to the previous and next. This makes things work a lot faster.

*Screenshot: FishEye's History Page*
ClearCase Support Now in Beta

FishEye’s support for IBM ClearCase is now in beta, following extensive feedback from our brave alpha testers.

Numerous improvements and bug fixes
Visit our issue tracker to see the full list of improvements and bug fixes.

FishEye 2.1 Changelog

On this page:

- From 2.1.1 to 2.1.2
- From 2.1.0 to 2.1.1

From 2.1.1 to 2.1.2

19th November 2009
This is a bugfix release.
The following issues are addressed by this release:
JIRA Issues (3 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-2308</td>
<td>Do not creat tags when files are copied into a tag area from outside the repository scope</td>
<td>📫</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2307</td>
<td>Can't set the use built-in symbolic rules for Subversion repositories</td>
<td>📫</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2303</td>
<td>revision icons not inline on p4 repo activity streams</td>
<td>📫</td>
<td>Closed</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:

JIRA Issues (8 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-2302</td>
<td>File History Diff navigation broken in 2.1</td>
<td>📫</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2301</td>
<td>Close LDAP NamingEnumerations</td>
<td>📫</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2300</td>
<td>Column headers don't display correctly in IE7 on Repositories tab</td>
<td>📣</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2298</td>
<td>out of bounds error?</td>
<td>📣</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2296</td>
<td>Attempt to continue when errors occur in the content retrieval logic</td>
<td>📫</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2271</td>
<td>Exceptions generated when users attempt to test ClearCase Repository connection</td>
<td>📣</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2255</td>
<td>Activity details are being duplicated on the Activity tab</td>
<td>📣</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-2176</td>
<td>Enable users to specify if they want the includes/excludes rule to be exact matches</td>
<td>📫</td>
<td>Closed</td>
</tr>
</tbody>
</table>

FishEye 2.0 Release Notes

⚠️ FishEye 2.1 has now been released. Read the Release Notes.

30 June 2009

Atlassian presents FishEye 2.0

FishEye 2.0 adds enhanced JIRA integration and a brand new user interface.

Highlights of this release:

- Activity Streams
- People
- Favourites, bookmarks & saved search
• Enhanced JIRA Integration
• Projects
• New User Interface
• Quick Navigation & Improved Quick Search
• Git Beta
• Numerous improvements and bug fixes

Thank you for your interest in FishEye 2.0.

See the documentation on Upgrading to this version.

Installing FishEye 2.0

You can now download the FishEye 2.0 from here. See the documentation on Upgrading to this version.

Highlights of FishEye 2.0

1

Activity Streams

FishEye 2 provides an activity stream showing commits, JIRA issues, and Crucible review activities. Activity streams are available across all repositories, from any directory, person, project and even individual files. Your home page provides a custom feed generated from the people, projects, and source you have selected as favorites.

Screenshot: Activity for a directory
People

Each user and committer in FishEye 2 has a page showing their statistics and activity. You can also see activity grouped by user from all repositories down to an individual file. You can sort by latest activity, commits, line count, even reviews using Crucible.

Screenshot: Users on trunk sorted by activity
**Favourites, bookmarks & saved search**

Clicking on the star icon on almost any artifact in FishEye adds that artifact to the your bookmark menu, and adds any downstream activity for that artifact to your personal dashboard. It also makes them searchable in quicknav. Charts and searches can be saved and given custom names for simpler reuse.

*Screenshot: New Favourite Menu*
Enhanced JIRA Integration

Multiple JIRA servers can be configured for your FishEye instance. Projects and repositories can be mapped to one, several, or all JIRA projects. FishEye retrieves summary issue information when you mouseover a JIRA key anywhere in FishEye. Optionally you can include issue activity in your FishEye activity streams as well.

Screenshot: Enhanced JIRA Integration
Projects

Multiple repositories or subsets of repositories can be combined into a project to provide a more focussed reporting and activity streams for teams that have source spread across repositories, or even different source control systems.

*Screenshot: A Project page*
New User Interface

The FishEye team has completely revamped the user interface. FishEye now uses an intuitive three pane view, with a rewritten file explorer and collapsible chart and information panes. There is better access to menus, user hovers, and lots more.

Screenshot: New annotation
Quick Navigation & Improved Quick Search

Typing into the search box on the top right of every page now gives you quick access to directories, files, changesets, even your personal favourites. Quick search now works across repositories and is significantly faster.

Screenshot: Quicknav
Git Beta

FishEye 2 adds beta support for Git repositories. There are still a few kinks to be worked out, but it provides full access to FishEye's functionality.

*Screenshot: The linux kernel's Git repository in FishEye*
Numerous improvements and bug fixes

Visit our issue tracker to see the full list of improvements and bug fixes.

FishEye 2.0 Changelog

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 Beta

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.

- **Support for new licenses**: Starter licenses are special offer, low-cost licenses that allow small teams to make use of Atlassian products.

The following issues are addressed by this release:

<table>
<thead>
<tr>
<th>JIRA Issues (12 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>FE-2163</td>
</tr>
<tr>
<td>FE-2157</td>
</tr>
<tr>
<td>FE-2143</td>
</tr>
<tr>
<td>FE-2095</td>
</tr>
<tr>
<td>FE-2054</td>
</tr>
<tr>
<td>FE-2029</td>
</tr>
<tr>
<td>FE-1933</td>
</tr>
<tr>
<td>FE-1851</td>
</tr>
<tr>
<td>FE-1425</td>
</tr>
<tr>
<td>FE-1367</td>
</tr>
<tr>
<td>FE-1277</td>
</tr>
<tr>
<td>FE-712</td>
</tr>
</tbody>
</table>

From 2.0.3 to 2.0.4

8th September 2009

This is a bugfix and improvement release. This release addresses the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (9 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>FE-2078</td>
</tr>
</tbody>
</table>

- **Support for new licenses**: Starter licenses are special offer, low-cost licenses that allow small teams to make use of Atlassian products.
### 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 (8 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>FE-1995</td>
</tr>
<tr>
<td>FE-1987</td>
</tr>
<tr>
<td>FE-1984</td>
</tr>
<tr>
<td>FE-1963</td>
</tr>
<tr>
<td>FE-1960</td>
</tr>
<tr>
<td>FE-1934</td>
</tr>
<tr>
<td>FE-1901</td>
</tr>
<tr>
<td>FE-811</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 (6 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>FE-1957</td>
</tr>
<tr>
<td>FE-1948</td>
</tr>
<tr>
<td>FE-1943</td>
</tr>
<tr>
<td>FE-1938</td>
</tr>
<tr>
<td>FE-1932</td>
</tr>
<tr>
<td>FE-778</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 (45 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>FE-1930</td>
</tr>
<tr>
<td>FE-1900</td>
</tr>
<tr>
<td>FE-1896</td>
</tr>
<tr>
<td>FE-1894</td>
</tr>
<tr>
<td>FE-1891</td>
</tr>
<tr>
<td>FE-1888</td>
</tr>
<tr>
<td>FE-1887</td>
</tr>
<tr>
<td>FE-1886</td>
</tr>
<tr>
<td>FE-1885</td>
</tr>
<tr>
<td>Ticket</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>FE-1882</td>
</tr>
<tr>
<td>FE-1881</td>
</tr>
<tr>
<td>FE-1875</td>
</tr>
<tr>
<td>FE-1874</td>
</tr>
<tr>
<td>FE-1866</td>
</tr>
<tr>
<td>FE-1864</td>
</tr>
<tr>
<td>FE-1863</td>
</tr>
<tr>
<td>FE-1857</td>
</tr>
<tr>
<td>FE-1849</td>
</tr>
<tr>
<td>FE-1845</td>
</tr>
<tr>
<td>FE-1841</td>
</tr>
<tr>
<td>FE-1839</td>
</tr>
<tr>
<td>FE-1836</td>
</tr>
<tr>
<td>FE-1823</td>
</tr>
<tr>
<td>FE-1821</td>
</tr>
<tr>
<td>FE-1801</td>
</tr>
<tr>
<td>FE-1798</td>
</tr>
<tr>
<td>FE-1797</td>
</tr>
<tr>
<td>FE-1796</td>
</tr>
<tr>
<td>FE-1788</td>
</tr>
<tr>
<td>FE-1786</td>
</tr>
</tbody>
</table>
### FishEye 2.1 Documentation

**From 2.0 Beta3 to 2.0**

**30th June 2009**

Full list of issues in this release:

| FE-1865 | removing a favourite and then clicking on it again causes an error | | Closed |
| FE-1854 | Activity stream on project page doesn't filter by mapped JIRA projects | | Closed |

| FE-1785 | 'All Activity' and 'Reviews' tabs on /committer/REPO/NAME appear active on mouseover | | Closed |
| FE-1783 | Deleting a repo doesn't delete committer mappings | | Closed |
| FE-1781 | Legend and Commit activity accordion segments don't add value when expanded | | Closed |
| FE-1780 | Breadcrumbs on /foo/admin/editAvatarType-default.do are wrong | | Closed |
| FE-1779 | gravitar admin page table has poor layout | | Closed |
| FE-1777 | Latest Activity on projects page should have time as well | | Closed |
| FE-1775 | Author/Moderator name at the top of the review should be a link | | Closed |
| FE-1767 | not selecting a repo in "Settings: User Mapping" and clicking "add" leads to empty user drop down | | Closed |
| FE-1764 | JIRA change aggregation shows same issue id twice when two files attached to the same issue | | Closed |
| FE-1741 | In the browse view, the 'show/hide deleted files' toggle is not respected when clicking on a node in the dir tree | | Closed |
| FE-1663 | hover popups' shadows dont expand with the popups, | | Closed |
| FE-1514 | Document disk space usage changes | | Closed |
| FE-1416 | CR-FE-1499 Rework | | Closed |
| FE-1402 | handle blank commiter names | | Resolved |
| FE-1189 | SVN initialscahn does not create changeset index until end of scan | | Closed |

<table>
<thead>
<tr>
<th>JIRA Issues (200 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>FE-1865</td>
</tr>
<tr>
<td>FE-1854</td>
</tr>
<tr>
<td>FE-1847</td>
</tr>
<tr>
<td>FE-1818</td>
</tr>
<tr>
<td>FE-1815</td>
</tr>
<tr>
<td>FE-1809</td>
</tr>
<tr>
<td>FE-1808</td>
</tr>
<tr>
<td>FE-1804</td>
</tr>
<tr>
<td>FE-1802</td>
</tr>
<tr>
<td>FE-1799</td>
</tr>
<tr>
<td>FE-1784</td>
</tr>
<tr>
<td>FE-1778</td>
</tr>
<tr>
<td>FE-1771</td>
</tr>
<tr>
<td>FE-1770</td>
</tr>
<tr>
<td>FE-1768</td>
</tr>
<tr>
<td>FE-1760</td>
</tr>
<tr>
<td>FE-1754</td>
</tr>
<tr>
<td>FE-1703</td>
</tr>
<tr>
<td>FE-1698</td>
</tr>
<tr>
<td>FE-1519</td>
</tr>
<tr>
<td>FE-1515</td>
</tr>
<tr>
<td>FE-1513</td>
</tr>
<tr>
<td>FE-1506</td>
</tr>
<tr>
<td>Issue</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>FE-1505</td>
</tr>
<tr>
<td>FE-1504</td>
</tr>
<tr>
<td>FE-1500</td>
</tr>
<tr>
<td>FE-1492</td>
</tr>
<tr>
<td>FE-1489</td>
</tr>
<tr>
<td>FE-1485</td>
</tr>
<tr>
<td>FE-1484</td>
</tr>
<tr>
<td>FE-1483</td>
</tr>
<tr>
<td>FE-1479</td>
</tr>
<tr>
<td>FE-1477</td>
</tr>
<tr>
<td>FE-1473</td>
</tr>
<tr>
<td>FE-1472</td>
</tr>
<tr>
<td>FE-1467</td>
</tr>
<tr>
<td>FE-1465</td>
</tr>
<tr>
<td>FE-1464</td>
</tr>
<tr>
<td>FE-1463</td>
</tr>
<tr>
<td>FE-1455</td>
</tr>
<tr>
<td>FE-1453</td>
</tr>
<tr>
<td>FE-1452</td>
</tr>
<tr>
<td>FE-1451</td>
</tr>
<tr>
<td>FE-1449</td>
</tr>
<tr>
<td>FE-1448</td>
</tr>
<tr>
<td>FE-1447</td>
</tr>
<tr>
<td>FE-1441</td>
</tr>
<tr>
<td>FE-1435</td>
</tr>
<tr>
<td>FE-1427</td>
</tr>
<tr>
<td>FE-1418</td>
</tr>
<tr>
<td>FE-1417</td>
</tr>
<tr>
<td>FE-1415</td>
</tr>
<tr>
<td>FE-1414</td>
</tr>
<tr>
<td>FE-1412</td>
</tr>
<tr>
<td>FE-1411</td>
</tr>
<tr>
<td>FE-1410</td>
</tr>
<tr>
<td>FE-1408</td>
</tr>
<tr>
<td>FE-1401</td>
</tr>
<tr>
<td>FE-1392</td>
</tr>
<tr>
<td>FE-1391</td>
</tr>
<tr>
<td>FE-1390</td>
</tr>
<tr>
<td>FE-1389</td>
</tr>
<tr>
<td>FE-1388</td>
</tr>
<tr>
<td>FE-1387</td>
</tr>
<tr>
<td>FE-1386</td>
</tr>
<tr>
<td>FE-1385</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>FE-1383</td>
</tr>
<tr>
<td>FE-1365</td>
</tr>
<tr>
<td>FE-1361</td>
</tr>
<tr>
<td>FE-1360</td>
</tr>
<tr>
<td>FE-1357</td>
</tr>
<tr>
<td>FE-1356</td>
</tr>
<tr>
<td>FE-1355</td>
</tr>
<tr>
<td>FE-1354</td>
</tr>
<tr>
<td>FE-1353</td>
</tr>
<tr>
<td>FE-1352</td>
</tr>
<tr>
<td>FE-1351</td>
</tr>
<tr>
<td>FE-1349</td>
</tr>
<tr>
<td>FE-1348</td>
</tr>
<tr>
<td>FE-1347</td>
</tr>
<tr>
<td>FE-1346</td>
</tr>
<tr>
<td>FE-1345</td>
</tr>
<tr>
<td>FE-1344</td>
</tr>
<tr>
<td>FE-1343</td>
</tr>
<tr>
<td>FE-1342</td>
</tr>
<tr>
<td>FE-1341</td>
</tr>
<tr>
<td>Issue</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>FE-1340</td>
</tr>
<tr>
<td>FE-1339</td>
</tr>
<tr>
<td>FE-1338</td>
</tr>
<tr>
<td>FE-1337</td>
</tr>
<tr>
<td>FE-1336</td>
</tr>
<tr>
<td>FE-1335</td>
</tr>
<tr>
<td>FE-1334</td>
</tr>
<tr>
<td>FE-1333</td>
</tr>
<tr>
<td>FE-1332</td>
</tr>
<tr>
<td>FE-1331</td>
</tr>
<tr>
<td>FE-1330</td>
</tr>
<tr>
<td>FE-1329</td>
</tr>
<tr>
<td>FE-1322</td>
</tr>
<tr>
<td>FE-1321</td>
</tr>
<tr>
<td>FE-1320</td>
</tr>
<tr>
<td>FE-1313</td>
</tr>
<tr>
<td>FE-1312</td>
</tr>
<tr>
<td>FE-1311</td>
</tr>
<tr>
<td>FE-1310</td>
</tr>
<tr>
<td>FE-1307</td>
</tr>
</tbody>
</table>

Changeset page should show longer extract of Crucible review title -- there's plenty
<table>
<thead>
<tr>
<th>Issue Key</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-1306</td>
<td>of room</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1305</td>
<td>Changeset page doesn't show spinner while loading diffs</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1296</td>
<td>EyeQL results which include &quot;group by&quot; clause disconnect when serving second page</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1294</td>
<td>-&gt; ACTION</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1293</td>
<td>-&gt; ACTION</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1291</td>
<td>refactor hover popup (cru/jira) linker</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1289</td>
<td>turn global.js for fisheye into the jquery equivalent, and to split it up into proper modules.</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1288</td>
<td>New UI treatment for extra change set page features</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1287</td>
<td>UI Rework</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1286</td>
<td>NPE viewing user-committer manager mappings in admin</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1285</td>
<td>Integrate new ui for file history page</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1279</td>
<td>fisheye code pointers should highlight the line they are pointing to</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1276</td>
<td>Add permission checks to /fe/ ajax actions</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1272</td>
<td>consolidate scripts into a single place, into head tag.</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1271</td>
<td>remove inline event handlers, replace with event binds</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1265</td>
<td>make tree open at current path</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1264</td>
<td>Integrate resizable column layout with dirlist.jsp</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1262</td>
<td>Layout in Repository plugin page is broken</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1259</td>
<td>just do it</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1258</td>
<td>plugin finangling</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1257</td>
<td>View review blockers report</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1256</td>
<td>plugin finangling</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1255</td>
<td>most active developers / directories</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1254</td>
<td>punch card chart</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1253</td>
<td>expose data via API</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1252</td>
<td>View code metrics report</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1251</td>
<td>create reports page</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1250</td>
<td>create webitem + decorator</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1249</td>
<td>Reports tab</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1247</td>
<td>jsp wrangling</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1246</td>
<td>action</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1245</td>
<td>Add a project page</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1244</td>
<td>update admin page</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1243</td>
<td>db schema change to store proj info</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1242</td>
<td>Add a collection of &quot;content roots&quot; across one or more repositories.</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1241</td>
<td>refactor javascript - remove prototype</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1240</td>
<td>Javascript refactor</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1239</td>
<td>jsp</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1238</td>
<td>new action to handle fetching / sorting</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1237</td>
<td>lucene searcher</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1236</td>
<td>List of users at /users and committers at /committers/REPO</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1235</td>
<td>UI integration</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1234</td>
<td>render chart table thingy</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1233</td>
<td>searchy extractor thingy</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1232</td>
<td>Activity calendar</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1231</td>
<td>UI integration</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1230</td>
<td>render chart</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1229</td>
<td>implement searchy extractor thingy</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1228</td>
<td>Activity histograms commits vs hour of day and day of week</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1227</td>
<td>UI integration</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1226</td>
<td>render the chart</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1225</td>
<td>implement searchy extractor thingy</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1224</td>
<td>Recent activity sparkline &amp; chart</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1223</td>
<td>just do it</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1222</td>
<td>Show personal review summary on user home page</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1221</td>
<td>just do it</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1220</td>
<td>Show open review count on user pages</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1218</td>
<td>jiralinkspan span is created inside anchor tags</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1217</td>
<td>simple impl</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1216</td>
<td>Ubiquitous (cross repo) quicksearch/nav</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1212</td>
<td>just do it</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1211</td>
<td>repo dropdown in breadcrumb bar</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1209</td>
<td>build new JSP</td>
<td>Closed</td>
</tr>
<tr>
<td>Issue ID</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>FE-1207</td>
<td>Reload the “files pane” with ajax from a click in the “tree pane” and update the breadcrumbs</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1206</td>
<td>ajax remaining data</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1204</td>
<td>Use a full tree to navigate the new browse page</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1196</td>
<td>make action</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1195</td>
<td>make a tag / javascript function / standardise</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1194</td>
<td>Built in smart Crucible linker</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1183</td>
<td>make repository and crucible defaults for jira issues that are not mapped redux</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1140</td>
<td>Syntax Highlighting</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1138</td>
<td>Review design of CommitterUserMapping.hmb.xml</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1136</td>
<td>Add the ability to return unique results in EyeQL</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1038</td>
<td>svn-connection getting wrong values out of the repository configuration</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-962</td>
<td>allow execution contexts to be added to quartz jobs / triggers</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-935</td>
<td>Allow users to remove the &quot;Email Review&quot; button via the GUI or add tool tip it to make its use clearer.</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-928</td>
<td>Repository with long names does not wrap correctly in folder view listing</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-914</td>
<td>fix diff text caching</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-906</td>
<td>REST auth-tokens should be instance-wide</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-904</td>
<td>Display commits for a branched file before the trunk commits that were performed prior to the branch operation</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-854</td>
<td>add smart filename search to quicksearch (including CamelCase initals)</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-851</td>
<td>Show related JIRA issues in annotated file view</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-838</td>
<td>FE: No hyperlink for copies expander</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-835</td>
<td>minify &amp; combine javascript</td>
<td>Closed</td>
</tr>
<tr>
<td>Key</td>
<td>Summary</td>
<td>Priority</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>FE-831</td>
<td>p4 Issue to do with line endings causes logs to fill up with unexpected line errors</td>
<td></td>
</tr>
<tr>
<td>FE-800</td>
<td>Date Constraint query combined with checking empty directories takes forever to return</td>
<td></td>
</tr>
<tr>
<td>FE-796</td>
<td>'reviews' EyeQL return clause should return all reviews that the revision is included in</td>
<td></td>
</tr>
<tr>
<td>FE-792</td>
<td>fix images for quicksearch dropdown</td>
<td></td>
</tr>
<tr>
<td>FE-788</td>
<td>Improve annotation Colours</td>
<td></td>
</tr>
<tr>
<td>FE-787</td>
<td>refactor quicksearch jsp</td>
<td></td>
</tr>
<tr>
<td>FE-758</td>
<td>Upgrade to Atlassian-Plugins 2.1</td>
<td></td>
</tr>
<tr>
<td>FE-743</td>
<td>Introduce validation for Updater in fisheye or make configuration foolproof</td>
<td></td>
</tr>
<tr>
<td>FE-682</td>
<td>the &quot;commit history&quot; histogram should optionally show changeset counts, not just revision counts</td>
<td></td>
</tr>
<tr>
<td>FE-654</td>
<td>Setting show-changelog-calendar to true will cause the changelog to take forever to load.</td>
<td></td>
</tr>
</tbody>
</table>

*From 2.0 Beta2 to 2.0 Beta3*

**5th June 2009**

Full list of issues in this release:

### JIRA Issues (200 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-1865</td>
<td>removing a favourite and then clicking on it again causes an error</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1854</td>
<td>Activity stream on project page doesn't filter by mapped JIRA projects</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1847</td>
<td>'star' instead of 'favourite' in cog menu</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1818</td>
<td>Links from files in a changeset to the file history page don't include the revision</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1815</td>
<td>a con: XML namespace is being added to saved config.xml</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1809</td>
<td>Enhance README.html to include links to Evaluator Guide, Installation Guide, and Upgrade Guide</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1808</td>
<td>FishEye RC1 tarball contains Readme.HTML that links to Crucible docs</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td></td>
<td>we shouldn't show an empty star before a search has been performed - they throw</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1804</td>
<td>an exception when clicked</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1802</td>
<td>with the blue background on the comment form you can't tell which button is active when tabbing through them</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1799</td>
<td>Collapse all files in review doesn't work if there is a comment anchor in the URL</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1784</td>
<td>Soft wrapping always selected in prefs menu</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1778</td>
<td>admin project page doesn't render default and allowed reviewers</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1771</td>
<td>Commit by hour chart time axis numbering incorrect</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1770</td>
<td>Inconsistent and unsatisfactory highlighting colours on review page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1768</td>
<td>defect label on comments has no css, but draft does</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1760</td>
<td>Infinite loop in CalculatedBucketGraphXY</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1754</td>
<td>Cross-repo QS repository membership and ordering changes between pages</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1703</td>
<td>Typo error</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1698</td>
<td>Improve LDAP Authentication so it does not Abandon initial bind or do an CMP</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1519</td>
<td>Change include/exclude parameters for Restore</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1515</td>
<td>p4 calls labels command without supplying path when it needs to get the info on a particular label</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1513</td>
<td>Upgrade to latest AGSL-1</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1506</td>
<td>Truncate number of lines in scrolling log</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1505</td>
<td>Restore should not fail if run from a different directory</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1504</td>
<td>Update FE System Requirements - remove IE 6 from support browsers</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1500</td>
<td>Let BackupManager store the job data in config.xml</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1492</td>
<td>Anna's misc M7 non-ui tasks</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1489</td>
<td>Add user timezone to send request by support so that all support requests have a valid timezone.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>Ticket</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>FE-1485</td>
<td>chart tweaks and fixes</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1484</td>
<td>Add directory tree to other pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1483</td>
<td>Enhance directory tree to display file</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1479</td>
<td>Make scale/chart nicer on code-metrics-report</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1477</td>
<td>Fix code-metrics-report</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1473</td>
<td>Make report plugin urls pretty</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1472</td>
<td>Fix 'revisions' tabulation on code-metrics report</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1467</td>
<td>Problem accessing Admin Remote API if the global anonymous access is off</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1465</td>
<td>Refactor report plugin packages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1464</td>
<td>Report Mode Navigation (similar to Search Mode Navigation)</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1463</td>
<td>Root (No Repository Context) Search Page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1455</td>
<td>Date Pickers should be jQuery pickers (remove old script file when done)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1453</td>
<td>Fix Structure (make search query controls in 'fixed/hard' section up top)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1452</td>
<td>Remove blanks from Author box</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1451</td>
<td>Change &quot;Search All Directories&quot; to &quot;Searching... &lt;BREADCRUMB TRAIL&gt;&quot;</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1449</td>
<td>Remove .do from global quicksearch url</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1448</td>
<td>QuickSearch Improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1447</td>
<td>Search Mode Navigation</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1441</td>
<td>Make revision comments appear on just one line in revision table</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1435</td>
<td>No progress indicator on dir tree twiddle</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1427</td>
<td>Improve group membership management</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1418</td>
<td>Combine summarize and close emails</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1417</td>
<td>CR-FE-1480 Rework</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1415</td>
<td>fix up pagination in users/ page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1414</td>
<td>Fix it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1412</td>
<td>Fix it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1411</td>
<td>Fix QuickNav</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1410</td>
<td>View Changeset Stream Context Fixes</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1408</td>
<td>activity-streams plugin wrangling</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1401</td>
<td>add a comment as one of the return columns in /search/FE/</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1392</td>
<td>Change mset messages displayed in browse history table should not be truncated (except via css)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1391</td>
<td>quicksearch pages continuously displays progress indicator</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1390</td>
<td>Fix error handling when loading directory subtrees via ajax</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1389</td>
<td>Poor Javascript performance on file history page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1388</td>
<td>can sort by checkboxes in browse view</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1387</td>
<td>bad title name in the config servlet for light svn plugin.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1386</td>
<td>Make physical/logical link labels indicate change to different state, not the current state</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1385</td>
<td>Make tooltip content more descriptive</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1383</td>
<td>RSS improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1365</td>
<td>Page listing only source files to browse</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1361</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1360</td>
<td>Chart jsp refactorings</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1357</td>
<td>write tag</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------</td>
<td>-----------</td>
<td>--------</td>
</tr>
<tr>
<td>FE-1356</td>
<td>Consistent time and age formats</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1355</td>
<td>update stream jsps to render a changeset</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1354</td>
<td>update all stream actions to take a csid</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1353</td>
<td>Maintain stream context when viewing a changeset page</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1352</td>
<td>just do it</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1351</td>
<td>preference and toggle to exclude own activity from home page stream</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1349</td>
<td>just do it</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1348</td>
<td>Switch all hover popups to use ajs.hover</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1347</td>
<td>just do it</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1346</td>
<td>Open settings pages in ajs.dialog</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1345</td>
<td>gravatar servlet</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1344</td>
<td>config screen + jsps</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1343</td>
<td>Avatar config screens</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1342</td>
<td>make the committer page group by user if there are multiple committer with same user</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1341</td>
<td>User &amp; committer list rationalisation</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1340</td>
<td>just do it</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1339</td>
<td>One jsp for global quicksearch and repo qsearch</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1338</td>
<td>just do it</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1337</td>
<td>Switch all dropdowns to ajs.dropdown</td>
<td>![status]</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1336</td>
<td>i.e. the old style changelog or what you get after you hit expand all</td>
<td>![status]</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1335</td>
<td>preference to show files rather than summary in activity streams</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1334</td>
<td>legend unfucking</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1333</td>
<td>sparkline</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1332</td>
<td>line history sparklines and user line history chart improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1331</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1330</td>
<td>Integrate new html for diff/annotation pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1329</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1322</td>
<td>this should be done in a single query for all committers you're interested in, not once for every user or every committer</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1321</td>
<td>you could use Math.max/min in a few places here</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1320</td>
<td>you shouldn't need to set the commiters as a parameter, just set the BreakdownOption in ParameterSetQuery to AUTHOR</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1313</td>
<td>there is no real point paging if we are going to do these types of calculations for every user in the system. It's just not going to work imho. Is sorting by num-reviews a must have? If so do one query that counts reviews for all users, grouped by u</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1312</td>
<td>change the comparator BY_RECENT_ACTIVITY to sort nulls last -- remove have NoActivityItem</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1311</td>
<td>use a better UI than ^ and V, see craig/pete</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1310</td>
<td>Rework from CR-FE-1441: FE-1236: sortable and paged list of users at /users and committers</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1307</td>
<td>Star in Changeset page breadcrumbs doesn't appear in Safari 3.2.1</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1306</td>
<td>Changeset page should show longer extract of Crucible review title -- there's plenty of room</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1305</td>
<td>Changeset page doesn't show spinner while loading diffs</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1296</td>
<td>EyeQL results which include &quot;group by&quot; clause disconnect when serving second page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1294</td>
<td>-&gt; ACTION</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1293</td>
<td>-&gt; ACTION</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>JIRA</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>FE-1291</td>
<td>refactor hover popup (cru/jira) linker</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1289</td>
<td>turn global.js for fisheye into the jquery equivalent, and to split it up into proper modules.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1288</td>
<td>New UI treatment for extra change set page features</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1287</td>
<td>UI Rework</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1286</td>
<td>NPE viewing user-committer manager mappings in admin</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1285</td>
<td>Integrate new ui for file history page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1279</td>
<td>fisheye code pointers should highlight the line they are pointing to</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1276</td>
<td>Add permission checks to /fe/ ajax actions</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1272</td>
<td>consolidate scripts into a single place, into head tag.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1271</td>
<td>remove inline event handlers, replace with event binds</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1265</td>
<td>make tree open at current path</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1264</td>
<td>Integrate resizable column layout with dirlist.jsp</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1262</td>
<td>Layout in Repository plugin page is broken</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1259</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1258</td>
<td>plugin finangling</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1257</td>
<td>View review blockers report</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1256</td>
<td>plugin finangling</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1255</td>
<td>most active developers / directories</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1254</td>
<td>punch card chart</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1253</td>
<td>expose data via API</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1252</td>
<td>View code metrics report</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1251</td>
<td>create reports page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1250</td>
<td>create webitem + decorator</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1249</td>
<td>Reports tab</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1247</td>
<td>jsp wrangling</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1246</td>
<td>action</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1245</td>
<td>Add a project page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1244</td>
<td>update admin page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1243</td>
<td>db schema change to store proj info</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1242</td>
<td>Add a collection of &quot;content roots&quot; across one or more repositories.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1241</td>
<td>refactor javascript - remove prototype</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1240</td>
<td>Javascript refactor</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1239</td>
<td>jsp</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1238</td>
<td>new action to handle fetching / sorting</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1237</td>
<td>lucene searcher</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1236</td>
<td>List of users at /users and committers at /committers/REPO</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1235</td>
<td>UI integration</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1234</td>
<td>render chart table thingy</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1233</td>
<td>searchy extractor thingy</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1232</td>
<td>Activity calendar</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1231</td>
<td>UI integration</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1230</td>
<td>render chart</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1229</td>
<td>implement searchy extractor thingy</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1228</td>
<td>Activity histograms commits vs hour of day and day of week</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1227</td>
<td>UI integration</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1226</td>
<td>render the chart</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1225</td>
<td>implement searchy extractor thingy</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1224</td>
<td>Recent activity sparkline &amp; chart</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1223</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1222</td>
<td>Show personal review summary on user home page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1221</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1220</td>
<td>Show open review count on user pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1218</td>
<td>jiralinkspan span is created inside anchor tags</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1217</td>
<td>simple impl</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1216</td>
<td>Ubiquitous (cross repo) quicksearch/nav</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1212</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1211</td>
<td>repo dropdown in breadcrumb bar</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1209</td>
<td>build new JSP</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1207</td>
<td>Reload the “files pane” with ajax from a click in the “tree pane” and update the breadcrumbs</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1206</td>
<td>ajax remaining data</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1204</td>
<td>Use a full tree to navigate the new browse page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1196</td>
<td>make action</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1195</td>
<td>make a tag / javascript function / standardise</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1194</td>
<td>Built in smart Crucible linker</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>FE-1183</td>
<td>make repository and crucible defaults for jira issues that are not mapped redux</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1140</td>
<td>Syntax Highlighting</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1138</td>
<td>Review design of CommitterUserMapping.hmb.xml</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1136</td>
<td>Add the ability to return unique results in EyeQL</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1038</td>
<td>svn-connection getting wrong values out of the repository configuration</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-962</td>
<td>allow execution contexts to be added to quartz jobs / triggers</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-935</td>
<td>Allow users to remove the &quot;Email Review&quot; button via the GUI or add tool tip it to make its use clearer.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-928</td>
<td>Repository with long names does not wrap correctly in folder view listing</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-914</td>
<td>fix diff text caching</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-906</td>
<td>REST auth-tokens should be instance-wide</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-904</td>
<td>Display commits for a branched file before the trunk commits that were performed prior to the branch operation</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-854</td>
<td>add smart filename search to quicksearch (including CamelCase initials)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-851</td>
<td>Show related JIRA issues in annotated file view</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-838</td>
<td>FE: No hyperlink for copies expander</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-835</td>
<td>minify &amp; combine javascript</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-831</td>
<td>p4 Issue to do with line endings causes logs to fill up with unexpected line errors</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-800</td>
<td>Date Constraint query combined with checking empty directories takes forever to return</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-796</td>
<td>'reviews' EyeQL return clause should return all reviews that the revision is included in</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-792</td>
<td>fix images for quicksearch dropdown</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-788</td>
<td>Improve annotation Colours</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-787</td>
<td>refactor quicksearch jsp</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-758</td>
<td>Upgrade to Atlassian-Plugins 2.1</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-743</td>
<td>Introduce validation for Updater in fisheye or make configuration foolproof</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-682</td>
<td>the &quot;commit history&quot; histogram should optionally show changeset counts, not just revision counts</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-654</td>
<td>Setting show-changelog-calendar to true will cause the changelog to take forever to load.</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>

**From 1.6.6 to 2.0 Beta**

Full list of issues in this release:

### JIRA Issues (200 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-1865</td>
<td>removing a favourite and then clicking on it again causes an error</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1854</td>
<td>Activity stream on project page doesn't filter by mapped JIRA projects</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1847</td>
<td>'star' instead of 'favourite' in cog menu</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1818</td>
<td>Links from files in a changeset to the file history page don't include the revision</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1815</td>
<td>a con: XML namespace is being added to saved config.xml</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1809</td>
<td>Enhance README.html to include links to Evaluator Guide, Installation Guide, and Upgrade Guide</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1808</td>
<td>FishEye RC1 tarball contains Readme.HTML that links to Crucible docs</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1804</td>
<td>we shouldn't show an empty star before a search has been performed - they throw an exception when clicked</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1802</td>
<td>with the blue background on the comment form you can't tell which button is active when tabbing through them</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1799</td>
<td>Collapse all files in review doesn't work if there is a comment anchor in the URL</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1784</td>
<td>Soft wrapping always selected in prefs menu</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1778</td>
<td>admin project page doesn't render default and allowed reviewers</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1771</td>
<td>Commit by hour chart time axis numbering incorrect</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1770</td>
<td>Inconsistent and unsatisfactory highlighting colours on review page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1768</td>
<td>defect label on comments has no css, but draft does</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1760</td>
<td>Infinite loop in CalculatedBucketGraphXY</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1754</td>
<td>Cross-repo QS repository membership and ordering changes between pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1703</td>
<td>Typo error</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1698</td>
<td>Improve LDAP Authentication so it does not Abandon initial bind or do an CMP</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1519</td>
<td>Change include/exclude parameters for Restore</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1515</td>
<td>p4 calls labels command without supplying path when it needs to get the info on a particular label</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1513</td>
<td>Upgrade to latest AGSL-1</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1506</td>
<td>Truncate number of lines in scrolling log</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1505</td>
<td>Restore should not fail if run from a different directory</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1504</td>
<td>Update FE System Requirements - remove IE 6 from support browsers</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1500</td>
<td>Let BackupManager store the job data in config.xml</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1492</td>
<td>Anna's misc M7 non-ui tasks</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1489</td>
<td>Add user timezone to send request by support so that all support requests have a valid timezone.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1485</td>
<td>chart tweaks and fixes</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1484</td>
<td>Add directory tree to other pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1483</td>
<td>Enhance directory tree to display file</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1479</td>
<td>Make scale/chart nicer on code-metrics-report</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1477</td>
<td>Fix code-metrics-report</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1473</td>
<td>Make report plugin urls pretty</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1472</td>
<td>Fix 'revisions' tabulation on code-metrics report</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1467</td>
<td>Problem accessing Admin Remote API if the global anonymous access is off</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1465</td>
<td>Refactor report plugin packages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1464</td>
<td>Report Mode Navigation (similar to Search Mode Navigation)</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1463</td>
<td>Root (No Repository Context) Search Page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1455</td>
<td>Date Pickers should be jQuery pickers (remove old script file when done)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1453</td>
<td>Fix Structure (make search query controls in 'fixed/hard' section up top)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1452</td>
<td>Remove blanks from Author box</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1451</td>
<td>Change &quot;Search All Directories&quot; to &quot;Searching... &lt;BREADCRUMB TRAIL&gt;&quot;</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1449</td>
<td>Remove .do from global quicksearch url</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1448</td>
<td>QuickSearch Improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1447</td>
<td>Search Mode Navigation</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1441</td>
<td>Make revision comments appear on just one line in revision table</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1435</td>
<td>No progress indicator on dir tree twiddle</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1427</td>
<td>Improve group membership management</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1418</td>
<td>Combine summarize and close emails</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1417</td>
<td>CR-FE-1480 Rework</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1415</td>
<td>fix up pagination in users/ page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1414</td>
<td>Fix it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1412</td>
<td>Fix it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1411</td>
<td>Fix QuickNav</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1410</td>
<td>View Changeset Stream Context Fixes</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1408</td>
<td>activity-streams plugin wrangling</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1401</td>
<td>add a comment as one of the return columns in /search/FE/</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1392</td>
<td>Change mset messages displayed in browse history table should not be truncated (except via css)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1391</td>
<td>quicksearch pages continuously displays progress indicator</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1390</td>
<td>Fix error handling when loading directory subtrees via ajax</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1389</td>
<td>Poor Javascript performance on file history page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1388</td>
<td>can sort by checkboxes in browse view</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1387</td>
<td>bad title name in the config servlet for light svn plugin.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1386</td>
<td>Make physical/logical link labels indicate change to different state, not the current state</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1385</td>
<td>Make tooltip content more descriptive</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1383</td>
<td>RSS improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1365</td>
<td>Page listing only source files to browse</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1361</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1360</td>
<td>Chart jsp refactorings</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1357</td>
<td>write tag</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1356</td>
<td>Consistent time and age formats</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1355</td>
<td>update stream jsp to render a changeset</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1354</td>
<td>update all stream actions to take a csid</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1353</td>
<td>Maintain stream context when viewing a changeset page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1352</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>Task ID</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>FE-1351</td>
<td>preference and toggle to exclude own activity from home page stream</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1349</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1348</td>
<td>Switch all hover popups to use ajs.hover</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1347</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1346</td>
<td>Open settings pages in ajs.dialog</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1345</td>
<td>gravatar servlet</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1344</td>
<td>config screen + jsps</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1343</td>
<td>Avatar config screens</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1342</td>
<td>make the committer page group by user if there are multiple committer with same user</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1341</td>
<td>User &amp; committer list rationalisation</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1340</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1339</td>
<td>One jsp for global quicksearch and repo qsearch</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1338</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1337</td>
<td>Switch all dropdowns to ajs.dropdown</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1336</td>
<td>i.e. the old style changelog or what you get after you hit expand all</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1335</td>
<td>preference to show files rather than summary in activity streams</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1334</td>
<td>legend unfucking</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1333</td>
<td>sparkline</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1332</td>
<td>line history sparklines and user line history chart improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1331</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1330</td>
<td>Integrate new html for diff/annotation pages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PrismID</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>FE-1329</td>
<td>just do it</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1322</td>
<td>this should be done in a single query for all committers you're interested in, not once for every user or every committer</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1321</td>
<td>you could use Math.max/min in a few places here</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1320</td>
<td>you shouldn't need to set the committers as a parameter, just set the BreakdownOption in ParameterSetQuery to AUTHOR</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1313</td>
<td>there is no real point paging if we are going to do these types of calculations for every user in the system. It's just not going to work imho. Is sorting by num-reviews a must have? If so do one query that counts reviews for all users, grouped by u</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1312</td>
<td>change the comparator BY_RECENT_ACTIVITY to sort nulls last -- remove have NoActivityItem</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1311</td>
<td>use a better UI than ^ and V, see craig/ Pete</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1310</td>
<td>Rework from CR-FE-1441: FE-1236: sortable and paged list of users at /users and committers</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1307</td>
<td>Star in Changeset page breadcrumbs doesn't appear in Safari 3.2.1</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1306</td>
<td>Changeset page should show longer extract of Crucible review title -- there's plenty of room</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1305</td>
<td>Changeset page doesn't show spinner while loading diffs</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1296</td>
<td>EyeQL results which include &quot;group by&quot; clause disconnect when serving second page</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1294</td>
<td>-&gt; ACTION</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1293</td>
<td>-&gt; ACTION</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1291</td>
<td>refactor hover popup (cru/jira) linker</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1289</td>
<td>turn global.js for fisheye into the jquery equivalent, and to split it up into proper modules.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1288</td>
<td>New UI treatment for extra change set page features</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1287</td>
<td>UI Rework</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1286</td>
<td>NPE viewing user-committer manager mappings in admin</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1285</td>
<td>Integrate new ui for file history page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>Task ID</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1279</td>
<td>Fisheye code pointers should highlight the line they are pointing to</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1276</td>
<td>Add permission checks to /fe/ ajax actions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1272</td>
<td>Consolidate scripts into a single place, into head tag.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1271</td>
<td>Remove inline event handlers, replace with event binds</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1265</td>
<td>Make tree open at current path</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1264</td>
<td>Integrate resizable column layout with dirlist.jsp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1262</td>
<td>Layout in Repository plugin page is broken</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1259</td>
<td>Just do it</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1258</td>
<td>Plugin finangling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1257</td>
<td>View review blockers report</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1256</td>
<td>Plugin finangling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1255</td>
<td>Most active developers / directories</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1254</td>
<td>Punch card chart</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1253</td>
<td>Expose data via API</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1252</td>
<td>View code metrics report</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1251</td>
<td>Create reports page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1250</td>
<td>Create webitem + decorator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1249</td>
<td>Reports tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1247</td>
<td>JSP wrangling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1246</td>
<td>Action</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1245</td>
<td>Add a project page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE-1244</td>
<td>update admin page</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1243</td>
<td>db schema change to store proj info</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1242</td>
<td>Add a collection of &quot;content roots&quot; across one or more repositories.</td>
<td>🔄</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1241</td>
<td>refactor javascript - remove prototype</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1240</td>
<td>Javascript refactor</td>
<td>🔄</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1239</td>
<td>jsp</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1238</td>
<td>new action to handle fetching / sorting</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1237</td>
<td>lucene searcher</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1236</td>
<td>List of users at /users and committers at /committers/REPO</td>
<td>🔄</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1235</td>
<td>UI integration</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1234</td>
<td>render chart table thingy</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1233</td>
<td>searchy extractor thingy</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1232</td>
<td>Activity calendar</td>
<td>🔄</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1231</td>
<td>UI integration</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1230</td>
<td>render chart</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1229</td>
<td>implement searchy extractor thingy</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1228</td>
<td>Activity histograms commits vs hour of day and day of week</td>
<td>🔄</td>
<td>Resolved</td>
</tr>
<tr>
<td>FE-1227</td>
<td>UI integration</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1226</td>
<td>render the chart</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1225</td>
<td>implement searchy extractor thingy</td>
<td>🔄</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-1224</td>
<td>Recent activity sparkline &amp; chart</td>
<td>🔄</td>
<td>Resolved</td>
</tr>
<tr>
<td>Issue Key</td>
<td>Issue Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>FE-1223</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1222</td>
<td>Show personal review summary on user home page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1221</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1220</td>
<td>Show open review count on user pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1218</td>
<td>jiralinkspan span is created inside anchor tags</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1217</td>
<td>simple impl</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1216</td>
<td>Ubiquitous (cross repo) quicksearch/nav</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1212</td>
<td>just do it</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1211</td>
<td>repo dropdown in breadcrumb bar</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1209</td>
<td>build new JSP</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1207</td>
<td>Reload the &quot;files pane&quot; with ajax from a click in the &quot;tree pane&quot; and update the breadcrumbs</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1206</td>
<td>ajax remaining data</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1204</td>
<td>Use a full tree to navigate the new browse page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1196</td>
<td>make action</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1195</td>
<td>make a tag / javascript function / standardise</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1194</td>
<td>Built in smart Crucible linker</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1183</td>
<td>make repository and crucible defaults for jira issues that are not mapped redux</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>FE-1140</td>
<td>Syntax Highlighting</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1138</td>
<td>Review design of CommitterUserMapping.hmb.xml</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1136</td>
<td>Add the ability to return unique results in EyeQL</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-1038</td>
<td>svn-connection getting wrong values out of the repository configuration</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>Issue Key</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>FE-962</td>
<td>allow execution contexts to be added to quartz jobs / triggers</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-935</td>
<td>Allow users to remove the &quot;Email Review&quot; button via the GUI or add tool tip it to make its use clearer.</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-928</td>
<td>Repository with long names does not wrap correctly in folder view listing</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-914</td>
<td>fix diff text caching</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-906</td>
<td>REST auth-tokens should be instance-wide</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-904</td>
<td>Display commits for a branched file before the trunk commits that were performed prior to the branch operation</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-854</td>
<td>add smart filename search to quicksearch (including CamelCase initials)</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-851</td>
<td>Show related JIRA issues in annotated file view</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-838</td>
<td>FE: No hyperlink for copies expander</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-835</td>
<td>minify &amp; combine javascript</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-831</td>
<td>p4 Issue to do with line endings causes logs to fill up with unexpected line errors</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-800</td>
<td>Date Constraint query combined with checking empty directories takes forever to return</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-796</td>
<td>'reviews' EyeQL return clause should return all reviews that the revision is included in</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-792</td>
<td>fix images for quicksearch dropdown</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-788</td>
<td>Improve annotation Colours</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-787</td>
<td>refactor quicksearch jsp</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-758</td>
<td>Upgrade to Atlassian-Plugins 2.1</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-743</td>
<td>Introduce validation for Updater in fisheye or make configuration foolproof</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-682</td>
<td>the &quot;commit history&quot; histogram should optionally show changeset counts, not just revision counts</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>FE-654</td>
<td>Setting show-changelog-calendar to true will cause the changelog to take forever to load.</td>
<td>Closed</td>
<td></td>
</tr>
</tbody>
</table>
FishEye 2.0 Upgrade Notes

This page contains information about upgrading to FishEye 2.0.

On this page:

- Browsers
- Known Issues
  - MySQL Database Issues
  - Problems with FishEye Freezing Unexpectedly

Browsers

FishEye 2.0 now supports the following browsers:

- Safari 3 (or later)
- FireFox 3 (or later)
- Internet Explorer 7 (or later)

Internet Explorer 6 is no longer supported.

Known Issues

MySQL Database Issues

When migrating your database to MySQL, you may encounter problems with very long comments in MySQL.

Problems with FishEye Freezing Unexpectedly

A known issue may cause FishEye 2.0 to freeze unexpectedly.

FishEye 2.0 Beta Release Notes

FishEye 2.0 Beta is a public development release leading up to FishEye 2.0. For all production use of FishEye, please use the latest complete 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 FishEye 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.
Atlassian presents FishEye 2.0 Beta

FishEye 2.0 adds enhanced JIRA integration and a brand new user interface.

Highlights of this release:

- Enhanced JIRA Integration
- New User Interface
- People View
- Plus numerous improvements and bug fixes

Thank you for your interest in FishEye 2.0 Beta.

See the documentation on Upgrading to this version.

Installing FishEye 2.0 Beta

You can now download the FishEye 2.0 Beta from here. See the documentation on Upgrading to this version.

Highlights of FishEye 2.0 Beta

Enhanced JIRA Integration

FishEye now has better JIRA integration, allowing you to see regular JIRA updates on your FishEye dashboard, as well as click on issue names to visit the JIRA instance they belong to. See instructions for JIRA configuration.

Screenshot: Enhanced JIRA Integration
New User Interface

Taking on board wide-ranging feedback from customers, the FishEye 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, allowing for different work styles.

Screenshot: New User Interface
People View

You can now view detailed charts and activity statistics people who use your FishEye instance. You can compare number of commits charted over time and other activity in detail.

Screenshot: People View
Plus numerous improvements and bug fixes

Alpha support for Git is activated in the beta but not complete. For more information, see the Git alpha documentation.

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.

Upgrading to the FishEye 2.0 Beta

**FishEye 2.0 Beta** is a public development release leading up to **FishEye 2.0**. For all production use and testing of FishEye, 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 FishEye 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 FishEye instance to the FishEye 2.0 Beta.

**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 FishEye.** Simply make a copy of your FishEye_install_dir/var/data/ directory.
- Download the FishEye zip file.

**Upgrade Procedure**

Your upgrade procedure depends on whether you are using a separate FISHEYE_INST directory. Read more about FISHEYE_INST in the Installation Guide.

**Method 1: Using a Separate FISHEYE_INST Directory**

1. Shutdown your existing fisheye server.
2. Make a backup of your FISHEYE_INST directory.
3. Extract the new FishEye version to a directory, leaving your FISHEYE_INST environment variable set to its existing location.
4. Start FishEye from the new installation.
5. Follow any version-specific instructions found in the Release Notes.

**Method 2: No Separate FISHEYE_INST Directory**

You will need to copy some files from your old FishEye installation to your new one.

1. Extract the new FishEye instance into a directory such as /NEW_FISHEYE/.
2. Delete the /NEW_FISHEYE/var directory.
3. Shut down the old FishEye 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 FishEye license, copy your fisheye.license to /NEW_FISHEYE/.
   (Atlassian-issued licenses are included within config.xml.)
7. Start FishEye from the new installation.

**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. Extract the new Fisheye archive into a directory such as /NEW_FISHEYE/.
6. Start Fisheye from the new installation by running NEW_FISHEYE/bin/run.sh. (Use run.bat on Windows).
7. If you 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.

**FishEye 2.0 Beta Reviewer’s Guide**
**FishEye 2.0 Beta** is a public development release leading up to **FishEye 2.0**. For all production use and testing of FishEye, 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 FishEye 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 FishEye 2.0 Beta. This page contains some direction on what is ready for testing in the beta, what the known issues are and how you can submit feedback.

**Known Issues**

This is a list of known issues with the beta; please do not raise requests related to these as solutions for them are already under way.

- Timestamp information will be destroyed; if you install the beta release as an upgrade on your existing FishEye instance data, then every timestamp in the database will be reset to midnight and that information is irreversibly lost. The date information remains intact.
- Clicking on the calendar date picker from the Changelog filter makes the filter disappear, without applying the constraints.

**Features Ready For Testing**

The following features in the FishEye 2.0 Beta are relatively hardened and using these thoroughly will help contribute to the final product.

- External Database Support; You can now store Crucible's internal data (reviews and associated data) in a MySQL or PostgreSQL database, as an alternative to the built-in HSQLDB. Note: Fisheye's repository cache's and the Fisheye user data is still stored on disk using infinity db.
- 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.

**Submitting feedback**

To submit feedback on the FishEye 2.0 Beta, please use the FishEye Forums.

**JIRA Integration in FishEye 2.0 Beta**

**FishEye 2.0 Beta** is a public development release leading up to **FishEye 2.0**. For all production use and testing of FishEye, 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 FishEye 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 FishEye.

1. **JIRA** is Atlassian's issue tracking product, which can be used to manage projects and associated work.

2. 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
- 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 FishEye. This requires you to enter details on the required JIRA server(s) via the FishEye 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.
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>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'.

**Editing Default JIRA Server Mappings**

This setting enables the FishEye feature that shows JIRA information in a dynamic window when you hover the mouse over a JIRA issue key in FishEye. It will also turn every issue key into a hyperlink to that issue in FishEye.

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 FishEye. 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 FishEye 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**

*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.
**FishEye 2.0 Beta** is a public development release leading up to **FishEye 2.0**. For all production use and testing of FishEye, 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 FishEye 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 information about the alpha pre-release support for Git 1.6, which is activated (but not complete) in the FishEye 2.0 Beta.

**Introduction and Disclaimer**

Git Alpha support is included in this beta release for Git 1.6. Atlassian stresses that this is alpha level support which means the following:

1. There is no admin support yet (no UI controls for the feature).
2. There are missing features - notably author line count support.
3. Subsequent betas are likely to require re-indexing (although Git indexing is reasonably fast).
4. Problems are to be expected. Do not use this on production instances.

However, feedback (see below) is appreciated and your comments will contribute to the quality of the final product.

**Configuration**

To configure a Git 1.6 repository in FishEye, you will need to stop FishEye and edit your `config.xml` file directly. An example repository configuration follows.

```
<repository name="test" enabled="true">
  <git location="git://git.example.com/repo.git"/>
  <security>
    <required-groups/>
  </security>
</repository>
```

The `<repository>` tag must directly follow the `<repository-defaults>` ending tag, or another ending tag for a repository (`</repository>`). If you do not place the repository tag correctly your application will not start up and is likely to throw an error like the following:
FishEye 1.6 Release Notes
23 September 2008

Atlassian presents FishEye 1.6

FishEye release 1.6 is a major release that adds functional and performance improvements. FishEye 1.6 has a faster, more powerful Quick Search which includes change indexing. It also now supports assigning administration privileges to user accounts or groups. This new FishEye is faster, containing both tune-ups of the core code as well as new features enhancing use in teams. Finally, there's also been additions made to the technology powering FishEye extensions, for third-party developers.

Highlights of this release:

- FishEye Search Enhancements
- Multiple Admin Users
- Remote API Improvements
- Changes to Charts
- Perforce Performance Tweaks
• Numerous improvements and bug-fixes

**DOWNLOAD**

**Upgrading to FishEye 1.6**

You can now download FishEye from here. If upgrading from a previous version, please follow the Upgrade Guide.

**Highlights of FishEye 1.6**

1

**FishEye Search Enhancements**

FishEye now indexes the content of every commit. This enables searching on all content in your Subversion, Perforce or CVS repository. Furthermore, you can search for added or deleted content. Read more.

**Quick Search**

FishEye's Quick Search has been completely rewritten for better accuracy and performance; Quick Search results are now returned instantaneously regardless of the repository size. Result types are better weighted to increase their relevance. Results have content preview with hit highlighting and are properly weighted, taking the date into account. Changeset results are returned based on content modifications, additions and deletions. FishEye now indexes the full content of every commit and will return changesets for content hits. Path and filename search is now an order of magnitude faster on large repositories. Finally, support for keywords in search enables you to quickly get the result you are looking for. For example, entering `cs:1902` will take you straight to changeset 1902.

**Screenshot: Improved FishEye Quick Search**

2

**General Search**

You can now search specifically for added and deleted content. This enables you to quickly find when code was deleted or modified. For instance, you can search for a method name to find out what it was previously called.

**Advanced Search and EyeQL**

All of the improvements made to Quick Search and General Search are available when using Advanced Search and EyeQL.
Multiple Admin Users

FishEye now allows the Administrator to grant other FishEye users Admin status. These Admin Users can also carry out any of the tasks that may have required the Administrator password. Admin privileges can be conferred using built-in or external directory group membership. Read more.

Screenshot: FishEye Admin Users

Remote API Improvements

Enhancing the Remote API, the FishEye team have incorporated a maxreturns option, which lets you control the quantity of returned results. Also the range of functions is extended with the new ancestor return clause and history search items. Finally, Perforce data is now exposed in the remote API, allowing Perforce jobs to be accessed via remote API calls. Read more.

Changes to Charts

FishEye 1.6 has a new change chart type. The change chart shows relative net line activity for a period. It is intended to give a quick “zoomed in” view of activity by extension, author, or subdirectory for a short period. Change charts start from a linecount of 0 at the start date for comparative purposes. Read more.

Screenshot: FishEye’s Change Chart
### Perforce Performance Tweaks

Perforce users can now specify a changelist to start scanning from. For some users, this will dramatically improve indexing time and runtime performance by ignoring irrelevant historical data. This is achieved with one simple configuration option called 'skip labels'.

### Numerous improvements and bug-fixes

<table>
<thead>
<tr>
<th>JIRA Issues (68 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-638 webwork 2.2.6 is not setting svnsymbolic in editrepository</td>
<td>🟥</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-483 Ensure all user preferences are in user profile</td>
<td>🟥</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-494 Add a link to the changeset on the annotation page</td>
<td>🟥</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-532 upgrade to webwork 2.2.7 (fixes security problem)</td>
<td>🟥</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-500 Expose Changeset &quot;Fixes Perforce Jobs&quot; data in EyeQL (and REST api)</td>
<td>🟥</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-617 for new svn repositories, default for t/b/t should be None</td>
<td>🟥</td>
<td>Closed</td>
</tr>
<tr>
<td>Task Number</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>FE-658</td>
<td>Chart constraint dropped on second level subdir</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-521</td>
<td>CLONE -StackOverflow</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-566</td>
<td>diff-to-previous on annotate page 404s</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-550</td>
<td>improve catch-all svn symbolic regex</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-552</td>
<td>Have an error page rather than 403 page when SVN permission denied</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-393</td>
<td>Use a single regularexpression to catch all tag/branch/trunk patterns</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-296</td>
<td>Get id button does not work in Trusted Application screen under IE</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-554</td>
<td>Add LIMIT clause to EyeQL documentation</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-559</td>
<td>Add 'ancestor' return clause to EyeQL</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-541</td>
<td>Allow limiting of number of results returned by remote API</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-578</td>
<td>&quot;Search just <a href="">repo:parh</a>&quot; breadcrumb links are borked - escapes parameter separators</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-549</td>
<td>Search tokenizes on underscores</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-536</td>
<td>&quot;List Repositories&quot; method in the remote API documentation</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-562</td>
<td>resolve springsource/log4j versioning problem</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-648</td>
<td>Disabling the check box, next to the config.xml file still sends the config.xml file via Admin &gt; SysInfo &gt; Raise a support request</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-378</td>
<td>review multithreading of RevCacheReader</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-389</td>
<td>only ask group to do group-membership tests for crowd users</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-434</td>
<td>show match-in-context in quicksearch (hit highlighting)</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-435</td>
<td>Quick-search redo UI requirements</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-438</td>
<td>Do content searches in quicksearch</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-338</td>
<td>Please add ability to specify initial revision from which to begin initial scan</td>
<td>Closed</td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>FE-584</td>
<td>Create the ability for customers to create support cases via Fisheye</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-436</td>
<td>Improved Quick Search</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-301</td>
<td>IndexOutOfBoundsException when opening annotated view</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-503</td>
<td>com.cenqua.fisheye.svn.SvnCache is throwing NPEs</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-504</td>
<td>command line reindex doesn't work when loopback is not 127.0.0.1</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-674</td>
<td>RSS Feed Title is missing space</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-288</td>
<td>Repositories still occasionally get stuck in Stopping state</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-600</td>
<td>TODO appearing in UI for comments</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-511</td>
<td>NPE when configuration file was not found</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-668</td>
<td>&quot;Edit repository details&quot; throws an NPE when a p4 repo has invalid info</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-524</td>
<td>upgrade to trusted apps 1.0, remove seraph dep</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-605</td>
<td>added files appear as empty diffs</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-675</td>
<td>RSS Feed Entries have almost no information in title</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-515</td>
<td>Allow P4 label scanning to be skipped</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-505</td>
<td>Retrieve Password</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-437</td>
<td>improve performance of filename searches in quicksearch</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-650</td>
<td>Documentation: New 'Advanced' mode hides attributes in Add Repository screen</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-467</td>
<td>Calculating the correct version for diffs doesn't work for perforce</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-630</td>
<td>Bundle SAL 1.1 in FishEye</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-540</td>
<td>&quot;Data Types and Structures&quot; information in the Remote API page</td>
<td>Closed</td>
</tr>
<tr>
<td>FE-479</td>
<td>next and previous links on diff and annotation pages</td>
<td>Closed</td>
</tr>
<tr>
<td>Ticket</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>FE-82</td>
<td>Re-index request: show message &quot;Could not stop repository within 20 seconds. Re-index aborted.&quot;</td>
<td></td>
</tr>
<tr>
<td>FE-555</td>
<td>Documentation: Add maxReturn parameter to remote API</td>
<td></td>
</tr>
<tr>
<td>FE-491</td>
<td>Hide optional fields in repo setup</td>
<td></td>
</tr>
<tr>
<td>FE-495</td>
<td>Self Signup layout borked</td>
<td></td>
</tr>
<tr>
<td>FE-636</td>
<td>Make update polling configurable in admin section</td>
<td></td>
</tr>
<tr>
<td>FE-643</td>
<td>&quot;Request Garbage Collection&quot; link on SysInfo/Support page redirects badly</td>
<td></td>
</tr>
<tr>
<td>FE-649</td>
<td>Cannot specify starting revision when creating perforce repo. The option only appears in the edit screen</td>
<td></td>
</tr>
<tr>
<td>FE-651</td>
<td>Weight quicksearch results by date</td>
<td></td>
</tr>
<tr>
<td>FE-428</td>
<td>Check for updates option</td>
<td></td>
</tr>
<tr>
<td>FE-482</td>
<td>SVN repositories default to UTF-8</td>
<td></td>
</tr>
<tr>
<td>FE-596</td>
<td>CSS syntax highlighting omissions</td>
<td></td>
</tr>
<tr>
<td>FE-513</td>
<td>Redirected to javax.servlet.ServletException 500 error page when accessing to a disabled/stopped repository</td>
<td></td>
</tr>
<tr>
<td>FE-319</td>
<td>Can't access /admin/ when logged in via Crowd</td>
<td></td>
</tr>
<tr>
<td>FE-607</td>
<td>Create documentation for new feature, create support issues via fisheye Admin &gt; Sysinfo screen</td>
<td></td>
</tr>
<tr>
<td>FE-685</td>
<td>Automatical user management from Crowd</td>
<td></td>
</tr>
<tr>
<td>FE-687</td>
<td>RSS Feed entry items truncate differently to 1.5.x</td>
<td></td>
</tr>
<tr>
<td>FE-464</td>
<td>upgrade to latest spring (at least 2.5.4)</td>
<td></td>
</tr>
<tr>
<td>FE-487</td>
<td>Remove the colon trailing repository names on the Fisheye home page</td>
<td></td>
</tr>
<tr>
<td>FE-529</td>
<td>Add an &quot;Add Repository&quot; link to the top of the repository list page in admin screen</td>
<td></td>
</tr>
<tr>
<td>FE-678</td>
<td>Link to file view in quicksearch results</td>
<td></td>
</tr>
</tbody>
</table>

**FishEye 1.6 Changelog**
From 1.6.5.a to 1.6.6

10th February 2009

This release is a bugfix release which address the following issues:

<table>
<thead>
<tr>
<th>JIRA Issues (19 issues)</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-1759</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-1481</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-1141</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-1086</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-1081</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-1078</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-1046</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-845</td>
<td>🟥</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-583</td>
<td>🍁</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-508</td>
<td>🍁</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-499</td>
<td>🍁</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-493</td>
<td>🍁</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-473</td>
<td>🍁</td>
<td>🍁Closed</td>
</tr>
<tr>
<td>FE-343</td>
<td>🍁</td>
<td>🍁Closed</td>
</tr>
</tbody>
</table>
### From 1.6.4 to 1.6.5.a

**22 December 2008**

This release contains a number of improvements and bug fixes.

- If you are using Perforce with a case-insensitive file system (such as in Windows), then you will need to re-index your repository after upgrading to this release.

Full list of issues fixed in this release:

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-1155</td>
<td>Different Chinese Encoded Characters in Annotated view &amp; Diff view</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-917</td>
<td>Author constraints with usernames that have &quot;&quot; fail</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-895</td>
<td>Using request.getPathInfo() may not be accurate when servlet is called from <a href="">jsp:include</a></td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-894</td>
<td>Generate Support Zip via the Administration console</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-888</td>
<td>If SVNKit cannot authenticate successfully, it goes into a tight loop retrying</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-878</td>
<td>Changing display name doesn't work</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-875</td>
<td>document system.properties</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-873</td>
<td>Documentation about Case Insensitive Perforce users</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-860</td>
<td>add the ability to set the max HTTP header size via a system property</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-857</td>
<td>Add doco for &quot;path != word&quot;</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-856</td>
<td>Add &quot;path = word&quot; to eyeql</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>
**FE-855** Improve the performance of watch manager

**FE-852** Diffs link shows up in HTML changeset emails for binary files

**FE-844** Improve handling of Perforce encoding - especially Shift-JIS

**FE-841** Files saved in another encoding beside UTF-8 (example shift-JIS) does not render properly in fisheye for perforce repositories.

**FE-840** Email notification takes longer times to be sent

**FE-833** Case of Sub Directories box doesn't match depot

**FE-830** Need to update svnkit that resolves an NPE to do with NTLM authentication.

**FE-827** Reading p4 annotate fails with "NumberFormatException" when customers annotate output does not being with <changeset>:

**FE-807** Update the jsvn scripts to point to the new 1.2.1r5011.atlassian.jar version of svnkit

**FE-754** Getting "Not Inserting Revision blah because a parent a parent directory is missing

**FE-624** Indexing changesets which contain a large number of changes (e.g. brach copy, etc) can cause out of memory errors

---

**From 1.6.3 to 1.6.4**

**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 (16 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>FE-819</td>
</tr>
<tr>
<td>FE-808</td>
</tr>
<tr>
<td>FE-802</td>
</tr>
<tr>
<td>FE-801</td>
</tr>
<tr>
<td>FE-799</td>
</tr>
</tbody>
</table>
This release rolls together several improvements and bug fixes.

- FishEye now indexes diff text for CVS repositories.
- Quicksearch improvements.
- Improve support for local SVN 1.5 repositories.

To take advantage of FishEye’s diff text search or historical file content quick search functions, you will need to re-index your repository.

Full list of issues fixed in this release:

<table>
<thead>
<tr>
<th>JIRA Issues (32 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>FE-786</td>
</tr>
<tr>
<td>FE-785</td>
</tr>
<tr>
<td>FE-777</td>
</tr>
<tr>
<td>FE-775</td>
</tr>
<tr>
<td>ID</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>FE-774</td>
</tr>
<tr>
<td>FE-768</td>
</tr>
<tr>
<td>FE-749</td>
</tr>
<tr>
<td>FE-747</td>
</tr>
<tr>
<td>FE-741</td>
</tr>
<tr>
<td>FE-730</td>
</tr>
<tr>
<td>FE-729</td>
</tr>
<tr>
<td>FE-726</td>
</tr>
<tr>
<td>FE-723</td>
</tr>
<tr>
<td>FE-721</td>
</tr>
<tr>
<td>FE-714</td>
</tr>
<tr>
<td>FE-711</td>
</tr>
<tr>
<td>FE-710</td>
</tr>
<tr>
<td>FE-709</td>
</tr>
<tr>
<td>FE-708</td>
</tr>
<tr>
<td>FE-707</td>
</tr>
<tr>
<td>FE-706</td>
</tr>
<tr>
<td>FE-703</td>
</tr>
<tr>
<td>FE-697</td>
</tr>
<tr>
<td>FE-694</td>
</tr>
<tr>
<td>Issue ID</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>FE-689</td>
</tr>
<tr>
<td>FE-672</td>
</tr>
<tr>
<td>FE-551</td>
</tr>
<tr>
<td>FE-457</td>
</tr>
<tr>
<td>FE-419</td>
</tr>
<tr>
<td>FE-305</td>
</tr>
<tr>
<td>FE-267</td>
</tr>
<tr>
<td>FE-240</td>
</tr>
</tbody>
</table>

**From 1.6.1 to 1.6.2**

This version number was skipped, in order to keep FishEye and Crucible version numbers in parallel. There is no FishEye release with a version of 1.6.2.

**From 1.6.0 to 1.6.1**

**24 September 2008**

This is a bug fix release.

- FE-700 An initial import of a subversion repository will generate a NPE in some circumstances.

**FishEye 1.5 Release Notes**

**15 April 2008**

**Atlassian presents FishEye 1.5**

FishEye release 1.5 is a major release that adds the ability to present historical, per-author line count information. This new suite of graphs show how much each user has contributed to the code base, over time.

**Highlights of this release:**

- Per-author lines of code statistics
- Charting improvements
- Customisable email templates
- Numerous improvements and bug-fixes
Upgrading to FishEye 1.5

You can now download FishEye from here. If upgrading from a previous version, please follow the Upgrade Guide.

Highlights of FishEye 1.5

1

Per-author lines of code statistics

Statistics for lines of code is now broken down per-author, providing an all-new level of detail. This allows you to see how many lines of code were contributed to your project by each author, over time.

⚠️ This requires changing a setting and re-scanning existing repositories. See the FishEye Upgrade Guide for more information.

Screenshot: FishEye Per-Author Line Count Chart

2

Charting improvements

The line graphs in FishEye have been improved, providing a better view of lines of code statistics from your project, as well as showing how this has grown.
Customisable email templates

You can now customise the content and appearance of email notifications that are sent to FishEye users. For example you can append a legal disclaimer, alter the subject line or provide custom header text for all messages.

Numerous improvements and bug-fixes

<table>
<thead>
<tr>
<th>JIRA Issues (54 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>FE-394</td>
</tr>
<tr>
<td>FE-382</td>
</tr>
<tr>
<td>Issue</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>FE-358</td>
</tr>
<tr>
<td>FE-344</td>
</tr>
<tr>
<td>FE-336</td>
</tr>
<tr>
<td>FE-335</td>
</tr>
<tr>
<td>FE-333</td>
</tr>
<tr>
<td>FE-331</td>
</tr>
<tr>
<td>FE-326</td>
</tr>
<tr>
<td>FE-325</td>
</tr>
<tr>
<td>FE-323</td>
</tr>
<tr>
<td>FE-321</td>
</tr>
<tr>
<td>FE-320</td>
</tr>
<tr>
<td>FE-317</td>
</tr>
<tr>
<td>FE-312</td>
</tr>
<tr>
<td>FE-307</td>
</tr>
<tr>
<td>FE-303</td>
</tr>
<tr>
<td>FE-302</td>
</tr>
<tr>
<td>FE-299</td>
</tr>
<tr>
<td>FE-298</td>
</tr>
<tr>
<td>FE-295</td>
</tr>
<tr>
<td>FE-290</td>
</tr>
<tr>
<td>FE-287</td>
</tr>
</tbody>
</table>
FE-282 P4 Files of type "unicode" appear as binary

FE-278 Cannot edit or delete Trusted Application

FE-273 Upgrade to Cenqua Licensing 1.6

FE-269 Editing repository details does not always end with a "you need to restart repository..." message

FE-265 Include appropriate licence/notice files

FE-264 When Crowd integration is enabled, Trusted Application requests should use the Crowd Db when determining if users exist

FE-263 XML-RPC calls generate responses non-conforming to XML-RPC spec

FE-258 Upgrade to seraph 0.36 when released

FE-242 autoadd login with crowd and max users creates spinning browser

FE-237 StackOverflow

FE-235 Don't "Index Content" on every server restart

FE-234 Add REST API docs to Confluence

FE-229 Duplicate LDAP users created with differing case

FE-224 Handle dependencies with Maven 2

FE-218 ensure this NPE doesn't crash the watch mechanism

FE-210 doc: tag names in FE, eyeql help

FE-209 404 page instead of diff view

FE-200 In Search Results, don't list every page

FE-197 Create a pom.xml for Fisheye/Crucible

FE-185 Add web resource plugin module support

FE-182 Lots of StackOverflow Errors in Log
FishEye 2.1 Documentation

FE-181  Multithread initial repository scan too  
Closed

FE-172  Email feed unsubscribe & default format  
Closed

FE-170  capitalisation on 'User Profile' page is a little inconsistent  
Closed

FE-164  Fisheye does not seem to recognize mac os line ending \r when displaying diffs  
Closed

FE-155  Documentation unclear on interaction of include/exclude & tag/branch configuration  
Closed

FE-139  Please add the ability to watch a single file  
Closed

FE-122  Allow Repositories to be indexed in parallel.  
Closed

FE-116  FishEye is returning a bad response for a particular annotated file  
Closed

FE-62  Admin screens for custom homepage and footer content  
Closed

FE-5  [P4] need to consider "utf8" etc file types  
Closed

FishEye 1.5 Changelog

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.

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

<table>
<thead>
<tr>
<th>JIRA Issues (21 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>FE-662</td>
</tr>
<tr>
<td>FE-593</td>
</tr>
<tr>
<td>FE-579</td>
</tr>
<tr>
<td>FE-577</td>
</tr>
<tr>
<td>FE-571</td>
</tr>
<tr>
<td>FE-568</td>
</tr>
<tr>
<td>FE-565</td>
</tr>
<tr>
<td>FE-561</td>
</tr>
<tr>
<td>FE-560</td>
</tr>
<tr>
<td>FE-558</td>
</tr>
<tr>
<td>FE-557</td>
</tr>
<tr>
<td>FE-546</td>
</tr>
<tr>
<td>FE-542</td>
</tr>
<tr>
<td>FE-517</td>
</tr>
<tr>
<td>FE-475</td>
</tr>
<tr>
<td>FE-474</td>
</tr>
<tr>
<td>FE-472</td>
</tr>
<tr>
<td>FE-402</td>
</tr>
<tr>
<td>FE-306</td>
</tr>
<tr>
<td>FE-280</td>
</tr>
<tr>
<td>FE-215</td>
</tr>
</tbody>
</table>

**From 1.5.2 to 1.5.3**

**23 June 2008**
This release contains bug fixes.

## JIRA Issues (2 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-512</td>
<td>FishEye may stop sending emails after a backup</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-478</td>
<td>&quot;Bad format for response:jira&quot; error when attempting to get ID from JIRA 3.12.3</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
</tbody>
</table>

### From 1.5.1 to 1.5.2

**27 May 2008**

This release contains bug fixes.

⚠️ Note: This release of FishEye corrects to some Perforce line counts when storing diffs. If you have been having this problem, you will need to ensure that the store-diffs setting is set to "true" and do a full re-index of your repository.

## JIRA Issues (16 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-485</td>
<td>update doco re p4:jobid regex</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-481</td>
<td>Support regex p4:jobid eyql search clause</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-476</td>
<td>author blame error on svn file replace</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-465</td>
<td>[mvn] upload jarjar 1.0rc7 into 3rdparty</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-455</td>
<td>Charting Colors Broken</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-441</td>
<td>Watches are not being deleted</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-439</td>
<td>Author info/Store diff info Warning when disabled</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-431</td>
<td>Email watch notification does not properly parse the checkin comments for links (to jira/cru for example)</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-429</td>
<td>DownloadableClasspathResource passes null content type to GzipFilter</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-383</td>
<td>Linecount graph calculation performance improvements</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-374</td>
<td>number render bug in blame legends</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
<tr>
<td>FE-359</td>
<td>Date constraint should do more than clip</td>
<td>![Priority]</td>
<td>![Status] Closed</td>
</tr>
</tbody>
</table>
### JIRA Issues (21 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-421</td>
<td>update doco wrt to &quot;enable/disable SSO&quot; in crowd</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-415</td>
<td>upgrade svnkit to 1.1.7</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-413</td>
<td>[crowd] logged out after 2 minutes when using another &quot;incompatible&quot; crowd app with FishEye/Crucible</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-412</td>
<td>Add ability to turn off SSO in FishEye/Crucible's crowd support</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-409</td>
<td>&quot;Using the fisheye screens&quot; doc page out of date</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-408</td>
<td>in subdir breakdown charts the &quot;. (this dir)&quot; category is not self-explanatory</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-406</td>
<td>&quot;Files in Dir&quot; entry in subdir chart</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-404</td>
<td>Setting diff style to side-by-side in profile doesn't stick</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-403</td>
<td>incrementallIndexThreads and initialIndexThreads are incorrectly stored in config.xml</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-392</td>
<td>debug logging overly verbose in 1.5</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-390</td>
<td>User display preference setting in profile not saved</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-386</td>
<td>Investigate UI Preference Behaviour</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>
FishEye 2.1 Documentation

<table>
<thead>
<tr>
<th>Issue ID</th>
<th>Description</th>
<th>Status</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-376</td>
<td>Old charting code can probably be removed now</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-372</td>
<td>charting title is wrong</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-369</td>
<td>clicking (eg) Changelog from Chart tab loses directory constraint</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-360</td>
<td>Small Line History Chart Tweaks</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-357</td>
<td>Create an admin interface to edit the username force-lowercase configuration in security</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-350</td>
<td>admin screen for force-lowercase</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-346</td>
<td>static content is not being gz encoded</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-277</td>
<td>Using &quot;Test path&quot; button on &quot;Add repository&quot; page saves the data and closes the form</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-26</td>
<td>FishEye Quick Start Guide</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>

**FishEye 1.4 Release Notes**

*December 5, 2007.*

The Atlassian FishEye team is delighted to present FishEye 1.4. FishEye 1.4 is a major release which focuses on integration, user management and performance.

The updated FishEye Administration interface provides support for groups and improved user management screens. 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.

The latest releases of FishEye and Crucible work together even more tightly than before, allowing you to see at a glance which files/changesets have been reviewed. You can also search within FishEye for files that have not yet been reviewed.

**Highlights of this release:**

- Enhancements to user management
- Crowd/SSO support
- Crucible integration
- Enhancements to JIRA plugin
- Plus over 30 improvements and bug-fixes

**Responding to your feedback:**

🌟 19 new feature requests/improvements implemented

*Your votes and issues* [http://jira.atlassian.com/browse/FE] help us keep improving our products, and are much appreciated.
Upgrading to FishEye 1.4

You can now download FishEye from here. If upgrading from a previous version, please follow the Upgrade Guide.

Highlights of FishEye 1.4

1

Enhancements to user management

In FishEye 1.3.7, we introduced support for public signup (self-registration). Now in release 1.4:

- Improved user interface makes user administration easier.
- Groups are supported.
- Renaming of users is supported.
- Read the documentation.

![Edit User Groups: matt](image)

2

Crowd/SSO support

- Inbuilt integration with Atlassian Crowd for authentication and authorisation.
- Users and groups in your Crowd directories now supported in FishEye.
- Single sign on (SSO) support via Crowd e.g. you can now sign in just once to access Atlassian JIRA, FishEye, Crucible, Confluence and Bamboo, and any other applications which support SSO.
- Read the documentation.
Crucible integration

Closer integration between FishEye 1.4 and Crucible 1.2:

- Links to existing Crucible reviews on the FishEye screens. So you can see which files/changesets have been reviewed.
- Search for Crucible data via EyeQL. For example, you can search for files that have not yet been reviewed.
Enhancements to JIRA plugin

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
- the 'FishEye' tab now shows review status (if applicable)
- ability to connect your JIRA instance to multiple FishEye instances
- ability to configure the FishEye plugin via the AppLinks plugin
- the FishEye plugin is now fully internationalisable

Plus over 30 improvements and bug-fixes
| FE-163  | DbException: Problem getting diff information for rev1 | Closed |
| FE-156  | Bug/feature request link at bottom of screens is wrong | Closed |
| FE-149  | eyeql textbox too big in safari3.0.4 | Closed |
| FE-148  | Ability to delete user groups | Closed |
| FE-147  | Subversion revision indexing fails | Closed |
| FE-146  | Upgrade to new version of yahoo library | Closed |
| FE-141  | Hit NPE when trying to add new user (built-in) | Closed |
| FE-138  | Upgrade atlassian-extras dependency to 1.10 | Closed |
| FE-136  | suggestion: in the FishEye Admin menu, consider change 'Misc' to 'System Administration' | Closed |
| FE-133  | Cascading documentation links within FishEye | Closed |
| FE-132  | Report FishEye and/or API version via the API | Closed |
| FE-128  | Make online help link to CAC documentation | Closed |
| FE-126  | UI preferences (showing/hiding graph, directory sort order) don't work if user isn't logged in | Closed |
| FE-121  | Excessively long debug and error log entry when using AJP auth, automatic fisheye user creation, and exceeding license limit. | Closed |
| FE-120  | NPE when using ajp for authentication | Closed |
| FE-119  | constraint in email watches can become corrupted by url escaping | Closed |
| FE-118  | per-rep linkers don't necessarily trump default linkers when they match the same string | Closed |
| FE-114  | Regex syntax highlighting StackOverflowError | Closed |
| FE-111  | FishEye should ignore FISHEYE_HOME variable | Closed |
| FE-108  | Syntax Highlighting is wrong (keywords match within identifiers) | Closed |
| FE-103  | ability to rename users | Closed |
| FE-100  | allow usernames to contain the @ character | Closed |
### FishEye 1.4 Changelog

On this page:

- From 1.4.2 to 1.4.3
- From 1.4.1 to 1.4.2
FishEye 2.1 Documentation

- From 1.4 to 1.4.1

From 1.4.2 to 1.4.3

7 February 2008

This release contains bug fixes.

### JIRA Issues (35 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-309</td>
<td>Trusted Application not stored properly in configuration file</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-304</td>
<td>Improve documentation on recommended hardware and software (JVM) settings for FishEye</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-284</td>
<td>login error and logout pages return blank page</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-272</td>
<td>Linker regex application is case-sensitive</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-266</td>
<td>Support protocols such as pserver for remote CVS repositories</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-262</td>
<td>trusted app admin screen doesn't support https:// urls</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-259</td>
<td>certificateTimeout isn't saved to config.xml</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-253</td>
<td>Adding large repository causes all other repositories to stop indexing</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-249</td>
<td>Cancelling SVN Operation due to timeout: what operation?</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-247</td>
<td>&lt;properties&gt; in config.xml not passed to custom authenticators</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-225</td>
<td>List public FishEye instances in FishEye documentation</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-217</td>
<td>Linecount graphs give incorrect results on antlr perforce database</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-211</td>
<td>Linkers don't work in fisheye</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-208</td>
<td>Linkers setting in Repository Defaults ignored</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-205</td>
<td>CLONE -Custom Authenticator's init method is being passed an empty Properties object</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-195</td>
<td>tmp folder fills up disk space rapidly</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>FE-186</td>
<td>Connect remote CVS</td>
<td></td>
<td>Closed</td>
</tr>
</tbody>
</table>
### FishEye 2.1 Documentation

| FE-169 | Username is shown instead of Display Name under AJPv13 | 🟣 | Closed |
|--------|--------------------------------------------------------|——|--------|
| FE-162 | whenever i start fisheye my net stops working, i could not open other web pages except fisheye | 🟣 | Closed |
| FE-124 | Subscribe to changelog RSS using permissions does not work as expected | 🟣 | Closed |
| FE-83  | Update links etc to new FishEye docs on CAC | 🟣 | Closed |
| FE-75  | Review FishEye docs on CAC | 🟣 | Closed |
| FE-74  | Export the XML, PDF and HTML versions and upload to ALLDOCS space | 🟣 | Closed |
| FE-60  | Ability to customize Fisheye welcome message | 🟣 | Closed |
| FE-36  | New Diff UI (in 1.3.5) Is A Step Down | 🟣 | Closed |
| FE-35  | apply new left-nav to Bamboo and Clover spaces | 🟣 | Closed |
| FE-33  | Point online help links to new FishEye doc space | 🟣 | Closed |
| FE-30  | Left-hand nav panel needs adaptation to Confluence 2.6 styles | 🟣 | Closed |
| FE-28  | Move ‘ Improve FishEye Scan Performance’ guide to FishEye permanent documentation space | 🟣 | Closed |
| FE-27  | Switch to Atlassian Logos | 🟣 | Closed |
| FE-25  | Move Fisheye docs to Confluence | 🟣 | Closed |
| FE-22  | After Fisheye forums are moved, update all links in docs | 🟣 | Closed |
| FE-18  | Allow FishEye to run inside Tomcat | 🟣 | Closed |
| FE-14  | Woke up to stack overflow in fisheye | 🟣 | Closed |
| FE-2   | I installed latest Crucible and fisheye.access = md5:... appeared to public frontend | 🟣 | Closed |

### From 1.4.1 to 1.4.2

This release contains some minor improvements and bug fixes.

- **Trusted Application Support**
  
  FishEye 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 would have used a single ‘system’ account to determine permissions. Now, FishEye/Crucible can apply the correct permission settings for the logged-in user.

- FishEye now bundles the SVNkit Client as the default library for interfacing with Subversion. This streamlines FishEye configuration for Subversion users.
- FishEye now pre-calculates line-graph data. This should improve performance in the rendering of line graphs.
- Hyphens are now allowed in project key names.

### JIRA Issues (7 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-207</td>
<td>upgrade JFree chart</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
<tr>
<td>FE-203</td>
<td>NPE in api when calling getRevision() on a tag directory</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
<tr>
<td>FE-193</td>
<td>Confirm recent svnkit/javahl still works with subversion server 1.1.x</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
<tr>
<td>FE-187</td>
<td>clarify how groups are associated with repositories</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
<tr>
<td>FE-177</td>
<td>Add Application Trust Capability to Fisheye and Crucible</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
<tr>
<td>FE-171</td>
<td>please update help-paths.properties to accommodate some page-renumbering</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
<tr>
<td>FE-55</td>
<td>File has empty history in FishEye</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
</tbody>
</table>

### From 1.4 to 1.4.1

This is a small bug-fix release.

### JIRA Issues (1 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE-178</td>
<td>CursorLengthException when indexing P4</td>
<td></td>
<td><img src="https://fish-eye.red-bean.com/images/icons/closed.png" alt="Closed" /></td>
</tr>
</tbody>
</table>

## FishEye 1.3 Release Notes

⚠️ FishEye 2.1 has now been released. Read the Release Notes.

FishEye 1.3 contains many bug fixes and improvements, and adds support for Perforce.

### Upgrading FishEye

You can now download FishEye from here. Information on installing FishEye can be found here. If upgrading from a previous version, please follow the Upgrade Guide.

### Highlights of FishEye 1.3

- Support for the Perforce version control system.
- SVN properties are now shown.
- Quicksearch now searches for changeset ids.
- New ‘mixed’ chart on annotation pages, showing author-over-time breakdown.
- Side by side diffs (1.3.1)
• Refer to the changelog for more details.

FishEye 1.3 Changelog

On this page:

• From 1.3.7 to 1.3.8
• From 1.3.6 to 1.3.7
• From 1.3.5 to 1.3.6
• From 1.3.4 to 1.3.5
• From 1.3.3 to 1.3.4
• From 1.3.2 to 1.3.3
• From 1.3.1 to 1.3.2
• From 1.3 to 1.3.1
• From 1.3beta9 to 1.3
• From 1.3beta8 to 1.3beta9
• From 1.2.5 to 1.3beta8

From 1.3.7 to 1.3.8

This release provides performance and bug fix improvements for Subversion and Perforce indexing including:

• Perforce determination of line counts is much more efficient
• Handle cases where the content of a file has been removed from a Perforce depot
• Change Subversion indexing strategy for repositories whose initial check-in is a large commit or copy
• Fix error which broke line count graph caching
• correct issues dealing with changes to tagged files
• correct issue with property changes in German locale when using SvnKit

From 1.3.6 to 1.3.7

This is a small bug-fix release (list of issues).

From 1.3.5 to 1.3.6

This is a small bug-fix release. It addresses a stack-overflow problem for some configurations.

From 1.3.4 to 1.3.5

Note: Upgrading to this version will force a complete re-index of P4 repositories.

Improvements

• User-friendly UI-based license entry and maintenance.
• Syntax highlight files when displaying a diff.
• Add Ruby and Coldfusion syntax highlighting. Look for custom syntax files in FISHEYE_INST/syntax.
• EyeQL enhancement: a new clause to match on changeset ids csid = "1234".
• EyeQL enhancement: more return types such as isAdded and isBinary.

Fixes

• Many performance improvements when scanning Perforce repositories.
• Changelog missing some changesets in CVS when there is a path constraint.
• Handle svn diff output in languages other than English.
• Fix problem with long directory names wrapping in the directory tree.
From 1.3.3 to 1.3.4

- Fix compatibility between Cenqua and Atlassian licenses.
- [SVN] Convert slash / characters to a hyphen - in tag/branch names.
- [SVN] Better handling of precedence of tag/branch/trunk symbolic rules.
- [SVN] Add a manual test field in Admin screens to test symbolic rule setup.

From 1.3.2 to 1.3.3

This build allows FishEye to be used with Atlassian licenses.

From 1.3.1 to 1.3.2

- Fix potential XSS vulnerability in quick-search page.
- Fix problem sending watch emails where the commit message contains a tab character.
- [SVN] Add support for requesting a rescan between given revisions.
- [SVN] Improve scan performance, and better handle add operations from outside FishEye's view of the repository.
- [SVN] Improve scan performance by not fetchings diffs for binary files.
- [SVN] Timeout settings now configurable via Admin screens.
- [SVN] Display SVN properties at the directory level.
- Fix Javascript problem in IE when logging into the Admin screens.

From 1.3 to 1.3.1

- The truncate diff setting should now work in Internet Explorer.
- Fix issue with duplicate paths in tarball generation.
- Unknown repos now return a 404 status rather than 500.
- [SVN] Handle empty content files when using SvnKit.
- [CVS] Allow $ in author names.
- FishEye now uses the tabwidth setting in each user's profile.
- [SVN] Fix issue where FishEye incorrectly states that no username was supplied.
- Fix IE7 directory spacing problem.
- Implement side-by-side diffs.

From 1.3beta9 to 1.3

- Various improvements when scanning Perforce repositories.
- [SVN] Fix for problem with diff hyperlinks to re-added files.
- Fix problem where some paths were not correctly html-escaped.
- Fix 'NoSuchFieldError deferredExpression' problem on some platforms (due to a 3rd-party library being included twice).
- Ensure LDAP connections are closed in all situations.

From 1.3beta8 to 1.3beta9

⚠️ Upgrading to 1.3beta9 will force a complete re-index of CVS repositories.

- Upgrade JVM requirement to 1.5+.
- Upgrade embedded HTTP engine (Jetty). This fixes some bugs and improves performance under load.
- Fix a performance problem (esp. under load). "Recent Changes" pages should return much faster now.
- Fix a very slow memory leak when FishEye is under load (for example, when it is being crawled by a web spider).
- Fix a problem where daily-emails would break after a backup was performed.
- [CVS] Fix an error introduced when FishEye builds its repository cache. This requires a full re-scan of CVS repositories.
- [CVS] Fix a problem where FishEye could not parse in RCS files author names that were only numerical digits.
- [CVS] Fix bug when creating tar/zip files from a branch constraint.
- [SVN] FishEye will now timeout long running SVN connections that have blocked.
- [SVN] Fix problem where FishEye was not storing SVN properties correctly.
- [SVN] Fix a bug when entering a revision beyond the current last revision in quick search.

**From 1.2.5 to 1.3beta8**

- [SVN] When importing a repository from a given start revision, you can now nominate if it should import the state of the repository at that revision, or just import changes made after that revision.
- [CVS] Fix a bug where FishEye would send out watch emails for historical changesets after a re-index.
- Performance improvements to changeset page when one of the files in the changeset has a very large history.
- [SVN] Some changes that improve the speed of the initial-scan for some SVN repositories.
- Fix a bug when FishEye generates RSS feed urls constrained by author, when the author has an "@" in their name.
- [SVN] Fix a bug when a tag is deleted (as part of a move).

**FishEye Release Summary**

**FishEye 2.1 (12-Nov-09)**

- Wiki Markup in Commit Messages
- Streamlined JIRA Integration
- FishEye Admin API
- History Page Performance Increases
- ClearCase Support Now in Beta
- More in release notes.

**FishEye 2.0 (30-Jun-09)**

- Activity streams
- People statistics
- Favourites, bookmarks & saved search
- Enhanced JIRA integration
- New user interface
- Git beta
- More in release notes.

**FishEye 1.6 (23-Sep-08)**

- FishEye search enhancements
- Multiple admin users
- Remote API improvements
FishEye 2.1 Documentation

- Changes to charts
- Perforce performance tweaks
- More in release notes.

FishEye 1.5 (14-Apr-08)

- Per-author lines of code statistics
- Charting improvements
- Customisable email templates
- More in release notes.

FishEye 1.4 (5-Dec-07)

- Enhancements to user management
- Crowd/SSO support
- Crucible integration
- Enhancements to JIRA plugin
- More in release notes.

FishEye 1.3 (1-Aug-07)

- Support for the Perforce version control system.
- SVN properties are now shown.
- Quicksearch now searches for changeset ids.
- New "mixed" chart on annotation pages, showing author-over-time breakdown.
- Side by Side diffs (1.3.1)
- More in release notes.

FishEye FAQ

FishEye FAQ

Answers to frequently asked questions about configuring and using FishEye.

- Top Evaluator Questions
  - Does FishEye support GIT?
  - How do I fix problems with indexing my repository?
  - How do I setup JIRA integration?
  - How do I setup LDAP or external user management?
  - How do I speed up slow CVS updates?
  - How do I start FishEye as a Windows service?
  - How do I view changesets and diffs?
  - How is FishEye licensed?
  - What kind of search capabilities does FishEye have?
  - What programing languages are supported?

- CVS FAQ
  - How does FishEye calculate CVS changesets?
FishEye 2.1 Documentation

• Example EyeQL Queries
  • How do I find changes made to a branch after a given revision?
  • How do I filter results?
  • How do I find changes between two versions, showing separate histories?
  • How do I find changes made between two version numbers?
  • How do I find commits without comments?
  • How do I find files on a branch, excluding deleted files?
  • How do I find files removed from a given branch?
  • How do I find revisions made by one author between versions?
  • How do I select the most recent revisions in a given branch?

• FishEye Developer FAQ

• General FAQ
  • About the Lines of Code metric
  • How do I avoid long reindex times when I upgrade?
  • What SCM systems are supported by FishEye?

• Installation & Configuration FAQ
  • Can FishEye be run as a Windows service?
  • Can I deploy FishEye or Crucible as a WAR?
  • How Do I Investigate the Crucible Database Schema?
  • Improve FishEye Scan Performance
  • Migrating FishEye Between Servers
  • Setting Up a CVS Mirror with rsync
  • What are the FishEye System Requirements?

• Licensing FAQ
  • Are anonymous users counted towards FishEye's licence limits?
  • What are the Starter License restrictions?

• Subversion FAQ
  • Errors 'SEVERE assert' or 'Checksum mismatch'
  • FishEye fails to connect to the Subversion repository after a short time of successful operation.
  • How can FishEye help with merging of branches in Subversion?
  • SVN Authentication Issues
  • Why do I need to describe the branch and tag structure for Subversion repositories?
  • Why don't all my tags show up in FishEye?

• 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

• Tips of the Trade
Troubleshooting
- After I commit a change to my CVS repository, it takes a long time before it appears in FishEye.
- FishEye freezes unexpectedly
- Fix Out of Memory errors by increasing available memory
- Generating a Thread Dump Externally
- I have installed FishEye, and the initial scan is taking a long time. Is this normal?
- I have installed FishEye, but there is no data in the Changelog.
- Initial scan and page loads are slow on Subversion
- It seems that FishEye's HTTP Header is Too Small
- JIRA Integration Issues
- Message 'org.tigris.subversion.javahl.ClientException svn Java heap space'
- On my Red Hat Linux system, after running for several days FishEye freezes and does not accept any more connections.
- Problems with very long comments and MySQL migration
- URLs with encoded slashes don't work, especially in Author constraints

Do you have a question, or need help with FishEye? Please create a support request.

CVS FAQ

FishEye CVS FAQ

- How does FishEye calculate CVS changesets? — FishEye's goal is to allow changesets to be seen as a consistent stream of atomic commits. Revisions are collated into the same changeset provided that:
  - They have the same commit comment.
  - They are by the same author.
  - They are on the same branch.
  - The changeset does not span more than 10 minutes.
  - The same file does not appear in a changeset more than once.

Example EyeQL Queries
EyeQL

- How do find changes made to a branch after a given revision?
- How do I filter results?
- How do I find changes between two versions, showing separate histories?
- How do I find changes made between two version numbers?
- How do I find commits without comments?
- How do I find files on a branch, excluding deleted files?
- How do I find files removed from a given branch?
- How do I find revisions made by one author between versions?
- How do I select the most recent revisions in a given branch?

For more information on using EyeQL, see the Reference guide.

**How do find changes made to a branch after a given revision?**

Find changes made to Ant 1.5.x after 1.5 FINAL:

```
select revisions where on branch ANT_15_BRANCH and after tag ANT_MAIN_15FINAL group by changeset
```

**How do I filter results?**

This query, finds files removed on the Ant 1.5 branch, but just returns the person and time the files were deleted:

```
select revisions where modified on branch ANT_15_BRANCH and is dead return path, author, date
```

**How do I find changes between two versions, showing separate histories?**

As above, but show the history of each file separately:
How do I find changes made between two version numbers?
Find changes made between Ant 1.5 and 1.5.1:

```
select revisions where between tags
  (ANT_MAIN_15FINAL, ANT_151_FINAL] group by
  file
```

How do I find commits without comments?
Using the Advanced Search and EyeQL you can find commits that do not have comments using the following query:

```
select revisions from dir / where  comment = 
  ""  group by changeset
```

How do I find files on a branch, excluding deleted files?
Find files on branch and exclude delete files:

```
select revisions where modified on branch
  ANT_15_BRANCH and not is deleted group by
  changeset
```

How do I find files removed from a given branch?
Find files removed on the Ant 1.5 branch:
How do I find revisions made by one author between versions?

Find changes made by conor to Ant 1.5.x since 1.5.0:

```
select revisions where modified on branch ANT_15_BRANCH and is dead group by changeset
```

How do I select the most recent revisions in a given branch?

Find Java files that are tagged ANT_151_FINAL and are head on the ANT_15_BRANCH: (i.e. files that haven't changed in 1.5.x since 1.5.1)

```
select revisions from dir /src/main where is head and tagged ANT_151_FINAL and on branch ANT_15_BRANCH and path like *.java group by changeset
```

FishEye Developer FAQ

This page contains answers to frequently asked questions posed by FishEye developers.

Feel free to comment, make submissions, or pose your own question on FishEye Development here.

- **Q:** I'm getting the error "API access is disabled" as a response from http://fisheye/api/rest/repositories on my installation. How do I enable the API as a Fisheye administrator?
  - **A:** There is a toggle to enable the API under "Server Settings" in the web admin interface. See Configuring the FishEye Web Server for more details.
Q: Is there any way to return unique results from an EyeQL query?
A: It is not currently possible to return unique results. An improvement request exists: FE-1136. Your vote and comments on that issue are appreciated.

General FAQ

GENERAL FAQ

About the Lines of Code metric

This page contains information about the Lines of Code metric and how it is processed and represented by FishEye.

On this page:
- Definition
- Disadvantages
- LOC in FishEye
- User-Specific LOC

Definition

Lines of Code or LOC (also known as Source Lines of Code - SLOC) is a quantitative measurement in computer programming for files that contains code from a computer programming language, in text form. The number of lines indicates the size of a given file and gives some indication of the work involved. LOC is literally the count of the number of lines of text in a file or directory.

LOC for a file/directory is the total number of lines in the relevant files, while LOC for an author is the number of lines blamed on that author. Neither of these should ever be less than zero. However, the change in LOC over a period of time can be negative if there was a net reduction in the LOC over the period.

Disadvantages

While it can be useful, LOC has some well documented disadvantages. Keep these disadvantages and limitations in mind when you use LOC in your work environment.

In addition, the nature of branching in SCM applications means that calculating a LOC value for a whole project is not possible. A naive summation of the LOC of all the branches will give a meaningless number that jumps every time a branch is copied to create a new branch. Thus, in FishEye we usually look at the LOC of the trunk, unless we can infer from the context that another branch is more appropriate.

LOC in FishEye

FishEye calculates the LOC for the trunk and each branch as defined in the repository (see here for more information about defining branches in Subversion). Also, FishEye calculates the LOC for each user, unless that facility is turned off in the repository (see Store Diff Info). The LOC count will include all files except those identified...
by the SCM as binary.

Fisheye presents LOC data as charts of the change in LOC over time, and as informational statistics in various places.

- **Chart pages**
  The best way to explore the evolution of LOC in your project is the LOC chart report where you can easily filter the LOC by branch, author, file extension and date range. Here you can investigate what caused a particular spike in the LOC charts, or find the user whom has the most lines of code blamed on them and how this has changed over time.

- **Repository-specific activity pages**
  These show trunk LOC statistics for the repository, limited to the directory being viewed and its subdirectories. The LOC charts show the LOC for the directory, using trunk LOC unless the directory can be identified as a branch.

- **User pages**
  Here, the statistics pane in the sidebar shows the trunk LOC blamed on the user for the all repositories that have user-specific LOC enabled. The chart shows the trunk LOC from all the repositories that the user has contributed to.

- **The global User List page**
  This shows the trunk LOC for all users from the repositories that have user-specific LOC enabled. Repository-specific user lists (in repositories that have user-specific LOC enabled) show the trunk LOC for the users and committers, limited to the directory being viewed and its subdirectories.

- **Project pages** show a chart of the LOC for all associated repository paths, and statistics include the trunk LOC for those paths.

**User-Specific LOC**

The evolution of user-specific LOC over subsequent commits can appear at first glance to be counter-intuitive. It is important to keep in mind that the LOC for a given user is the number of lines in the repository that were last changed by them (as calculated by Fisheye).

A couple of simple examples:

- Alice adds a files with 30 lines to the SCM. Her LOC for this file is now 30. She then edits the file, deletes 10 lines and adds 20 (+20 -10). Her LOC is now 40, as is the LOC of the file.

- Alice adds a files with 30 lines to the SCM. Her LOC for this file is now 30. Now Bob edits the file, deletes 10 lines and adds 20 (+20 -10). Alice now has LOC of 20, because Bob deleted 10 lines that were blamed on her, and Bob has LOC of 20, from the 20 lines he added. The total LOC is still 40.

A user can have LOC on a branch that they have never committed on, if something that has been blamed on them is copied. For example, a developer may have never committed to a particular branch, but FishEye may still report a lot of LOC for them in that area.

One current limitation of FishEye's user-specific LOC calculation is the handling of merging. For example, if a file has been changed on both trunk and branch, and the changes made on the branch are merged to trunk, the changes made on branch will generally be blamed on the person who did the merge; not the person who made the change.

**How do I avoid long reindex times when I upgrade?**
Mitigating Lengthy Re-Index Times

If re-indexing your repository takes longer than you can allow, you can use a temporary copy of your repository and FishEye instance to reduce downtime during the re-indexing process.

Re-indexing With a Temporary Copy of Your FishEye Instance

To re-index a temporary copy of your FishEye instance:

1. Make a copy of your FishEye instance to another server. See 'How to Make a Temporary Copy of Your FishEye Instance' below for instructions.
2. Upgrade the temporary FishEye, then start it up, connected to your repository. It will automatically begin the scanning process.
   - If you are concerned about the repository being overloaded by the scanning process, you can make a copy of that as well. See 'How To Make a Temporary Copy of Your Repository' below for instructions.
      - If you do that, you must edit the config.xml of your temporary FishEye instance to point to your temporary repository.
3. The copied instance will run its course without affecting your production instance.
4. a. Shutdown both your servers completely.
   b. Make a backup of your FISHEYE_INST directory.
   c. Replace the FISHEYE_INST/var/cache directory on live FishEye with the FISHEYE_INST/var/cache from your test server.
   d. Download the latest Crucible from Atlassian downloads.
   e. Follow the instructions in the Upgrade Guide to upgrade to the new version.
5. The scan of the temporary FishEye instance (and repository, if you copied that also) is complete.
   - You're now free to delete the temporary copy(s).

How To Make a Temporary Copy of Your FishEye Instance

To make a copy of your FishEye instance, follow the instructions for Migrating FishEye Between Servers.

How To Make a Temporary Copy of Your Repository

To make a copy of your repository, do the following. Use rsync (for CVS repositories in the Linux environment) or svnsync documentation (for Subversion only).

How to Re-Index a Single Repository on a Test Server

If you need to re-index your repository on your production system but don't want to burden your production server, carry out the following steps:

1. Install another instance of FishEye on a test server (the same FishEye version as the one you are using).
2. Add a repository to Crucible with the exact same name and details as that referenced by the production server.
3. Let that finish indexing. Go to 'Administration' > 'View Repository List' > 'Stop' (shown next to the name of your repository) and disable on both production and test.
4. Copy over the FISHEYE_INST/var/cache/REPO directory on the production FishEye with the FISHEYE_INST/var/cache/REPO directory from the test server.
   - For this procedure, neither server needs to be shut down.

What SCM systems are supported by FishEye?

To see the list of SCM systems that is supported by FishEye, see System Requirements.

Installation & Configuration FAQ
Can FishEye be run as a Windows service?

FishEye can be run as a service under Microsoft Windows.

To run FishEye as a service you can either use SRVANY and INSTSRV to run java.exe or create a Java Service Wrapper. A mechanism to run FishEye as a service will be incorporated at a later stage. In the meantime, example wrapper files written by FishEye 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.

```
# JDK 1.5 Additional Parameters for jmx
wrapper.java.additional.4=-Dcom.sun.management.jmxremote
wrapper.java.additional.5=-Dcom.sun.management.jmxremote.password.file=./wrapper/jmxremote.password
wrapper.java.additional.10=-Dwrapper.mbean.name="wrapper:type=Java Service Wrapper Control"
```
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:

```java
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:

```ini
# Initial Java Heap Size (in MB)
wrapper.java.initmemory=32

# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=256
```

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 FishEye or Crucible as a WAR?**

Unfortunately FishEye and Crucible can not 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.

**How Do I Investigate the Crucible Database Schema?**

The purpose of this guide is to walk you through connecting to Crucible's embedded Hypersonic SQL Database using the Database Administration tool DBVisualizer. This will allow you to investigate the FishEye database schema.

*On this page:*

- Prerequisites
- Connection Procedure
  - Create the Database Connection
  - Select a JDBC Driver
  - Select the Database Path
  - Enter the Connection Details
  - Connect to the Embedded Database
  - Viewing the Tables and their References

Read on for step by step instructions on how to Configure DbVisualizer and connect it to FishEye's HSQLDB database.

**Prerequisites**

1. Download and install the latest copy of DBVisualizer.
2. Create a backup of your database, which you will connect to.
3. Ensure that Crucible is not running or use DBVisualizer to connect to the backup of your database and not the actual production version.

Connection Procedure

Please ensure that you read and follow the instructions below carefully.

![Ensure you create a backup (Administration > Backup) of your data before attempting any modifications.](image)

Create the Database Connection

1. Open Connection Wizard.

   **Screenshot: Opening the Connection Wizard**

2. Enter an identifiable name for the connection. For example: `cruc1.6.6-std`

   **Screenshot: Entering a Name for the Connection**
3. Click 'Next'.

Select a JDBC Driver

1. From the drop down list select **HSQLDB Embedded**

*Screenshot: Selecting a Database*
2. Click on **Load Driver Files**

3. Browse to FISHEYE_HOME/lib directory where the hsqldb*.jar file is located, eg
   /Users/pkamal/crucible-1.6.6/lib/hsqldb-1.8.0.10.jar

**Select the Database Path**

1. **Browse** to your `<Crucible-Home>` directory if you are going to connect to your production database (crucible is shutdown), or **Browse** to your expanded `<Backup>` directory that was created when you unzipped the backup file you made (if you just want to run some queries against your data).

   2. Open the `/var/data/crudb` folder.
   3. Select the `crucible.properties` file

**Enter the Connection Details**

1. Remove the `.properties` from the end of `crucible`

   *Screenshot: Entering Connection Details*
2. Type in sa for the username.
3. Leave the password field blank.
4. Click on Test Connection to verify that the details are correct.
5. Click on ‘Finish’ to complete the setup.
   Refer to the example screenshot above if you are unsure

Connect to the Embedded Database

1. Select the connection from the list on the left hand side.

   Screenshot: Connecting to the Embedded Database
2. You can now click on ‘Connect’ to connect to the embedded database.

Viewing the Tables and their References

1. Click on tables on the left and references in the middle to see an image that displays all tables and shows references.

2. The following image applies to Crucible 1.6.6 tables. Please note that the tables will be different in different versions of the product.

Improve FishEye Scan Performance

This page contains information about improving the performance of FishEye repository scans.

**Background Information**

When you add a repository, FishEye needs to perform a once-off scan through the repository to build up its initial index and cache. This scan can take some time. Until this scan is complete, you may find that some data is not
displayed. As a guide, FishEye should be able to process about 100KB-200KB per second on an averaged-size PC. If FishEye is accessing the repository over the network (e.g. over a NFS mount), then you should expect the initial scan to take longer.

**General Improvements**

You can increase the speed of your scans using the following options:

- If your repository is non-local, set up a local repository mirror on the FishEye server. This will provide a major speed boost for anyone scanning a repository across a network.
- Exclude unused file types, unused directories and specific large files from FishEye.

**Improve Update Performance during Initial Scan**

One option is break large repositories into multiple smaller repositories. While this technique will not improve the overall initial scan time, it allows for all fully scanned repositories to be updated while the initial scan is still being performed on those remaining.

In FishEye 1.3.4 and later, the initial and incremental scans happen in separate, single threads. So, splitting the repositories will allow incremental scans to run concurrently alongside the initial scans. You may also wish to split projects into separate repositories, since permissions in FishEye are applied on a per-repository basis.

**Improving Initial Scan performance for an SVN Repository**

The http/s protocol has the slowest performance during the initial scan. The svn protocol (svn://) is faster and the file protocol (file:///) is the fastest. Therefore if you find your initial scan takes an extended amount of time (in some cases weeks), you should consider switching over from the http/s protocol to the svn or file protocol to define the location of your SVN repository. (Use svnsync to mirror the repository onto the fisheye server, so that you can access it with the file protocol.)

```
E.g. Switch from
https://example.com/svn/project/

to
svn://example.com/svn/project/

or
file:///home/user/some/location/svn/project
```

In order for SVN protocol to work you need to have set up an svnservice based server.

**Performance Support**

If you have implemented at least one of the suggestions above but are still experiencing slow performance, ask us for help:

1. First read the Tuning Fisheye document.
2. Turn on debug logging using the command line debug flag.
3. Allow FishEye to continue its initial scan overnight.
4. Create a new support request in the FishEye project, including your server environment and log files with the problem description.
Migrating FishEye Between Servers

This page describes the process for migrating FishEye between servers.

If you have defined the FISHEYE_INST Environment Variable, then upgrades and migrations of your FishEye instance will be relatively simpler.

If you have defined FISHEYE_INST

1) Shut down your current FishEye server completely.
2) Copy the FISHEYE_INST directory to your destination server.
3) Copy and set up all of your Environment Variables from your source server to your destination server (remembering to set up your FISHEYE_INST directory to point to the location where you copied the data to in Step 2).
4) Install FishEye on your destination server.
5) Start FishEye. It should pick up your environment variables, and from that access your FISHEYE_INST directory, which contains your configuration.

If you have not defined FISHEYE_INST

1) Shut down your current FishEye server completely.
2) Copy the FISHEYE_HOME/config.xml file and FISHEYE_HOME/var folder into to one folder on your destination server, called fisheye_inst.
3) Copy and set up all of your Environment Variables from your source server to your destination server. In addition to this, set up the FISHEYE_INST env variable such as follows

```bash
export FISHEYE_INST=/path/to/fisheye_inst
```

replacing the /path/to/fisheye_inst with the fully qualified path to the fisheye_inst folder you set up in Step 2.
4) Install FishEye on your destination server.
5) Start FishEye. It should pick up your environment variables, and from that access your FISHEYE_INST directory, which contains your configuration.

Setting Up a CVS Mirror with rsync

In situations where running FishEye on the same server as your CVS repository is not practical or possible, you can use the Linux utility rsync to mirror the CVS repository contents onto the FishEye server. This is possible because CVS data is stored in a reasonably simple form in the file system.

We recommend this to achieve best performance when FishEye and CVS cannot be hosted on the same machine.

⚠️ This workaround requires the ability to SSH into both machines. Linux and Mac OS X operating systems have rsync built in. For Windows, you will need to install rsync.

Diagram: A Scenario Where rsync is Required
To set up a CVS mirror with rsync,
1. You will need to set up a local directory on the FishEye server for the mirrored CVS content, ensuring that this server has ample disk space to store the current CVS database and any future space requirements.

2. We will refer to your CVS instance as ‘CVS_HOME’ and your new ‘mirror directory’ on the FishEye server as ‘MIRROR_HOME’.

3. Type the following rsync command into the console of the FishEye server:

   ```
   rsync --backup CVS_HOME MIRROR_HOME
   ```

   A real-world example would look something like this:

   ```
   rsync --backup \julius\webapps\cvs\data \\
datastore\FishEye\cvs-mirror\n   ```

4. Schedule the rsync command to run regularly with a cron job. Running hourly is a good default interval. Under Windows, use a native task scheduler.

5. With the cron job active, you will have established rsync to run an hourly comparison of the two directories and copy any changes across to the mirror directory as they occur. Note that running the rsync process will impact the FishEye server's performance (and also the CVS server's) to a certain degree.

6. In the FishEye admin interface, add the local 'mirror directory' as a new CVS repository and run the initial scan. As this is local data on the same file system, FishEye's scanning of this data will be optimal.

7. Adjust the FishEye Updater Full Scan period to one hour (the default is 15 minutes).

8. The rsync configuration is now complete. Monitor the disk space on both servers to ensure there is adequate headroom for the mirroring process.

For more information on the syntax for rsync, visit the rsync home page.

**What are the FishEye System Requirements?**

Visit the FishEye System Requirements.

**Licensing FAQ**

- Are anonymous users counted towards FishEye's licence limits? Users accessing FishEye anonymously are, for all intents and purposes, unlimited users.
- What are the Starter License restrictions? FishEye Starter Licenses are restricted to ten users, ten committers per repository, as well as a total of five repositories.

**Are anonymous users counted towards FishEye's licence limits?**
The short answer is no. If you are using FishEye in your organisation but most users require only anonymous access (that is, you have not set access restrictions on the content in your repositories), then an unlimited number of anonymous users can be accommodated regardless of the FishEye licence you are using.

Users accessing FishEye anonymously are, for all intents and purposes, *unlimited* users.

However, if your users require permissions and controlled access to specific content in your repositories, then they will need to log in to FishEye. Hence, these users will need to create accounts and will be factored into the licence limit.

**What are the Starter License restrictions?**

This page explains the limitations of the FishEye Starter license and provides general information about using this license in production.

*On this page:*

- What is a Starter License?
- What are the Starter License restrictions?
- What happens if I exceed the Starter License limits?
  - Evaluate
  - Upgrade
  - Reconfigure your repository
- Frequently Asked Questions

**What is a Starter License?**

Starter licenses are low-cost licenses that allow small teams to make use of Atlassian products (see more information). FishEye Starter licenses were introduced with the release of FishEye 2.0.5 (October 2009).

**What are the Starter License restrictions?**

FishEye Starter Licenses are restricted to ten users, ten committers per repository, as well as a total of five repositories.

**What happens if I exceed the Starter License limits?**

If you have more than five repositories, FishEye will prevent more than five repositories from being enabled at any given time. Administrators can control which five repositories are enabled.

If you exceed more than ten committers in a repository, a warning will appear at the top of pages for the entire system, stating the following:

```
NOTE: This repository, (repository-name) has more than ten committers which exceeds the limits for your Starter license. Indexing has stopped. To fix this, you can ‘Evaluate’, ‘Upgrade’ or ‘Reconfigure your repository’.
```

*Screenshot: Warning for Starter License Limits*
The links in this warning will lead you to the following solutions:

Evaluate

30-day evaluation licenses are available to allow you to try out FishEye and other Atlassian products. You can select a license that allows more users than your current license.

Upgrade

You can upgrade your license at any time (via the Atlassian online store), which will remove the committer and repository limits which apply to the Starter License.

Reconfigure your repository

This option lets you configure your repository to remain within the limits of the Starter License. You can take the following actions to reduce the scope of FishEye’s indexing:

1. **Change the repository definition to look at a subset of your repository**
   Typically this involves setting the path within your repository that you wish FishEye to index. Read more.

2. **Exclude parts of the repository**
   You can exclude portions of your repository that you are not interested in. Committers that are active in only these areas will not appear in FishEye and not be included in the committer count. Read more.

3. **Set a starting point**
   Some of the FishEye SCM integrations allow you to configure a starting revision from which to start indexing. All commits prior to this starting point are not included in FishEye and do not contribute to the committer count. Read more.

Once you have reconfigured your repository, you will need to re-index the repository, allowing you to remain under the limits of the Starter license.

**Frequently Asked Questions**

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>What happens when the 11th unique committer is encountered during indexing?</td>
<td>For all SCMs other than CVS FishEye will index all revisions up to but not including the revision that introduced the 11th committer. Since CVS is indexing is file-by-file based, FishEye will index files until it reaches the committer limit. Remaining files in the repository are not indexed. This means only files which have been indexed will be displayed in changesets and changesets may be incomplete.</td>
</tr>
<tr>
<td>What happens when a FishEye instance with a Starter license is started, using existing indexes with more than five repositories?</td>
<td>FishEye will only start indexing on the first five repositories. An administrator can use admin UI to adjust which repositories are enabled and hence control the five repositories that are started. FishEye should then be restarted.</td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>What happens when a FishEye instance with a Starter license is started, using existing indexes with one or more repositories with more than ten committers?</td>
<td>FishEye will display all information currently indexed but for each repository that has reached the ten committer limit, no further revisions will be indexed.</td>
</tr>
<tr>
<td>What happens on upgrade from a Starter license, if indexing has been paused due to the committer limit being reached?</td>
<td>On restart of FishEye, indexing will resume for all repositories. Each repository can restarted individually to avoid restarting FishEye. Due to the nature of CVS indexing, we recommend reindexing any CVS repositories which have reached the committer limit prior to the license upgrade.</td>
</tr>
<tr>
<td>What happens when upgrading from a Starter license, when repositories have not started due to the repository limit being reached?</td>
<td>On restart of FishEye, all enabled repositories will start. Each repository can restarted individually to avoid restarting FishEye.</td>
</tr>
<tr>
<td>What happens if my evaluation license has expired and I upgrade to a Starter license, however I have exceeded the Starter license limitations?</td>
<td>As described above, a maximum of five repositories will start and for any repository with more than 10 committers, no further indexing will occur. All existing indexed content is retained and can be viewed.</td>
</tr>
<tr>
<td>What happens when downgrading to a Starter license, where the repository limit has been exceeded?</td>
<td>A maximum of five of your configured repositories will start running. The remainder will not start but will continue to be available.</td>
</tr>
<tr>
<td>What happens when downgrading to a Starter license, where the committer limit has been exceeded for one or more repositories?</td>
<td>No further indexing will occur for the repositories where the committer limit has been exceeded.</td>
</tr>
</tbody>
</table>

**Subversion FAQ**
### FishEye Subversion FAQ

- **Errors 'SEVERE assert' or 'Checksum mismatch'** — SVNKit may have problems with older version Subversion servers - versions 1.1.x and prior.

- **FishEye fails to connect to the Subversion repository after a short time of successful operation.** — On Unix systems, the `svn://` protocol is usually handled by `inetd` or `xinetd`. These daemons apply, by default, a connection per second limit to incoming connections. Any connections above this rate are rejected by the server.

- **How can FishEye help with merging of branches in Subversion?** — In merge management, the main advantages of FishEye come from its search functionality. If you record the revisions merged when you check in a merge result, you can find this information in FishEye easily for the next merge operation.

- **SVN Authentication Issues** — If you have defined multiple repositories in FishEye for the same Subversion server and those repositories use different credentials, FishEye may not use the correct credentials.

- **Why do I need to describe the branch and tag structure for Subversion repositories?** — In Subversion, branches and tags are defined by convention, based on their path within a repository, and not directly defined by the repository. A few different layout alternatives are commonly used. It is also possible that you are using your own custom layout. As a result you need to describe to FishEye which paths in your repository are used as branches and tags.

- **Why don't all my tags show up in FishEye?**

---

### Errors 'SEVERE assert' or 'Checksum mismatch'

When using SVNKit, you may see errors in the FishEye log such as 'SEVERE: assert #B' or 'Checksum mismatch'. SVNKit may have problems with older version Subversion servers - versions 1.1.x and prior. If this is the case you should either use the native JavaHL layer or upgrade your Subversion server to a more recent version.

### FishEye fails to connect to the Subversion repository after a short time of successful operation.

If you use the `svn://` protocol to access a Subversion repository, you may notice that FishEye fails to connect to the repository after a short time of successful operation.

On Unix systems, the `svn://` protocol is usually handled by `inetd` or `xinetd`. These daemons apply, by default, a connection per second limit to incoming connections. Any connections above this rate are rejected by the server.

Two options for fixing this problem:

- Ask your system administrator increase the connection rate allowed for the `svn` connection by updating the `xinetd` configuration, or
- Specify a connection per second limit in your FishEye repository definition, to prevent FishEye from exceeding the `xinetd` limits.

### How can FishEye help with merging of branches in Subversion?

FishEye gives you a logical view of your branched files so you can see activity on a single file across multiple branches/trunk.

In merge management, the main advantages of FishEye come from its search functionality. If you record the
revisions merged when you check in a merge result, you can find this information in FishEye easily for the next merge operation.

As an example, let's say you have a branch `dev` created at revision 1300 from `trunk`. Development has proceeded on both `trunk` and `dev`. At some point you wish to add the latest `trunk` changes into the `dev` branch. Let's say that is at revision 1400. When you check in the results of this merge, you would use some standard format checkin comment such as:

```
merge from trunk to dev 1300:1400
```

When you come to do the next merge, say at revision 1500, you can use FishEye search to find this checkin comment and know what the starting point for the merge should be. You can then check this in as:

```
merge from trunk to dev 1400:1500
```

Merges back to `trunk` from the `dev` branch are managed in the same way.

**SVN Authentication Issues**

If you have defined multiple repositories in FishEye for the same Subversion server and those repositories use different credentials, FishEye may not use the correct credentials.

FishEye does not directly control when authentication information is used to access Subversion repositories. It delegates this operation to the JavaHL layer in use. JavaHL will ask FishEye to supply credentials when required, using a callback. The default JavaHL layer shipped with FishEye, svnkit, can cache credentials at the server level rather than at the repository level.

If you experience this problem, you can configure FishEye to use the native JavaHL implementation, which will correctly apply the appropriate credentials.

**Why do I need to describe the branch and tag structure for Subversion repositories?**

In Subversion, branches and tags are defined by convention, based on their path within a repository, and not directly defined by the repository. A few different layout alternatives are commonly used. It is also possible that you are using your own custom layout. As a result you need to describe to FishEye which paths in your repository are used as branches and tags.

It is very important that you correctly define in FishEye the layout you are using. If you do not, FishEye will not know which paths represent tags and branches. This will prevent FishEye from relating different versions of the same logical file across separate paths within your repository. It will also mean that FishEye's cache will be much larger as each tagged path will be indexed separately. This will result in an increase in the initial scan time and may reduce runtime performance.

If you are having trouble using Subversion tags, see How Tags Work in Subversion.

**Why don't all my tags show up in FishEye?**

This page gives a detailed technical explanation of why certain issues affect Subversion users.

*On this page:*

- Introduction
- How Subversion Processes Tags and Branches
- An Example from a Live Subversion Repository
- Avoid Modifications in the Tag Area
Introduction

When accessing Subversion via FishEye, you may see references to tags in the branches drop-down menu. In the example below, we can see tag1 and tag3 in the drop-down menu but not tag2:

Screenshot: The Branches Drop-Down Menu in FishEye

In actual fact, the branches drop-down menu shows only branch names. It does not show tags, but in some instances FishEye will synthesise a branch name to record certain operations. To understand how this occurs, you will need some background knowledge on Subversion tagging (introduced in the following segments of this page).

How Subversion Processes Tags and Branches

In Subversion, tags are only a convention and are typically the result of a copy operation from the trunk to a tag area in the tags directory. When FishEye processes this copy operation, it recognises that the destination is a tag directory and tags the source file on trunk with the name of the tag.

i.e. FishEye is interpreting the Subversion copy to a tag directory as a tagging operation on the trunk files.

For regular changes in your Subversion repository, FishEye records each change against a branch where the change took place. If, however, after tagging, you make a change to a file in the tagged area, you are making a change outside trunk or a recognized branch. FishEye records such changes by creating an artificial branch name and associating that branch name with the change. The branch name is derived from the tag name by prepending "tag:" (in other words, the characters "tag:" appear as the first part of the name). The same thing will occur if you create a new file in the tagged area which does not come from an existing branch or trunk.

This is the reason you see some of your tags in the branch drop down. It means that for those tags, you have made a modification after the tagging operation.

An Example from a Live Subversion Repository

For example, consider tag4 in this screenshot:

Screenshot: Subversion Tag Changes in FishEye
There are two changes here. The first creates the tag and the second adds a new file in the tagged area. This will result in the creation of an artificial branch, called "tag:tag4" within FishEye.

**Avoid Modifications in the Tag Area**

In general, it's not good practice to make changes in the tag areas of a Subversion repository. Such changes can easily get lost if they are not applied to trunk or a current branch. It is preferable to make the change in trunk or a branch and then create a new tag to capture the update. Nevertheless, since Subversion tagging is merely a convention, this is sometimes convenient. FishEye handles this situation as described above.

**Conclusion**

In general a lot of systems have a large number of tags which would make the drop-down unworkable. This is the reason the tag field is a text-entry box below the branch drop-down menu in FishEye.

Since tags and branches are based on location convention in Subversion, the constraint is less effective than on other SCMs. You can always see the tag or branch you are interested in, based on its location in the repository. For example, the subdirectory list here shows all tags:

*Screenshot: Subdirectory Listing in FishEye*
Support Policies

This page is an index for the Atlassian Support Policies that apply to this product:

- 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

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 (i.e. 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](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 encourage and display customer comments and votes openly in our issue tracking systems,
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**

**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 (e.g. JIRA 4.0) 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 (ie it doesn't require a major architectural change)
  OR
- The issue is a security issue, and falls under our Security 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 issue in an Atlassian product is discovered and resolved, Atlassian will inform customers through the following mechanisms:

- A security advisory will be posted in the documentation.
- A copy of the advisory will be sent to the product mailing-lists. These lists are mirrored on our forums.
- If the person who reported the issue wants to publish an advisory through some other agency (for example, CERT), Atlassian will assist in the production of that advisory, and link to it from our own.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Patch Policy

Our Security Patch Policy

When a security issue is discovered, Atlassian will endeavour to:

- Issue a new, fixed version as soon as possible,
- Issue a patch to the current release (e.g. JIRA 4.0) and the latest maintenance release for the last major version of a product (e.g. JIRA 3.13.5),
- Issue patches for older versions if feasible.

Patches will generally be attached to the relevant JIRA issue.

Visit our general Atlassian Patch Policy as well.

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, rating the vulnerability as one of the following:

- Critical
- High
- Moderate
- Low

Below is a summary of the factors which we use to decide on the severity level, and the implications for your installation.

**Severity Level: Critical**

We classify a vulnerability as critical if most or all of the following are true:

- 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.

**Severity Level: High**

We give a high severity level to those vulnerabilities which have the potential to become critical, but have one or more mitigating factors that make exploitation less attractive to attackers.

For example, given a vulnerability which has many characteristics of the critical severity level, we would give it a level of high if any of the following are true:

- The vulnerability is difficult to exploit.
- Exploitation does not result in elevated privileges.
- The pool of potential victims is very small.

Note: If the mitigating factor arises from a lack of technical details, the severity level would be elevated to critical if those details later became available. If your installation is mission-critical, you may want to treat this as a critical vulnerability.

**Severity Level: Moderate**

We give a moderate severity level to those vulnerabilities where the scales are slightly tipped in favour of the potential victim.

The following vulnerabilities are typically rated moderate:

- Denial of service vulnerabilities, since they do not result in compromise of a target.
- 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**

We give a low severity level to those vulnerabilities which by themselves have typically very little impact on an organisation's infrastructure.

Exploitation of such vulnerabilities usually requires local or physical system access. Exploitation may result in client-side privacy or denial of service issues and leakage of information about organisational structure, system configuration and versions, or network topology.
Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Tips of the Trade

Below are some links to external blog posts and articles containing technical tips and instructions on setting up and using FishEye. This page presents an opportunity for customers and community authors to share information and experiences.

The references here are specific to FishEye and are technical 'how to' guides written by bloggers who use FishEye. For more general information on source repository insight, 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 FishEye. 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 FishEye. 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 it on your production site.

On this page:

- Using a reverse proxy to Fisheye from Tomcat 5+ - Blog
- Diary of a FishEye Hacker - Part 1 and Part 2

Technical Tips

Using a reverse proxy to Fisheye from Tomcat 5+ – Blog

- By: Dieter Wimberger, on his Coalevo blog.
- About: How to use a reverse proxy to Fisheye from Tomcat 5.
- Date and FishEye version: 21 Jan 2008; FishEye 1.4.2.
- Related documentation: Integrating with Other Web Servers.
Plugin Development

Diary of a FishEye Hacker – Part 1 and Part 2

- By: Dan Hardiker, on the Adaptavist blog
- About: How to write a FishEye reporting plugin that provides a framework for alternative views on the repository data
- Date and FishEye version: 6 May 2008; FishEye 1.5
- Related documentation: FishEye Development Hub

Have you written a technical tip for FishEye?
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
FishEye documentation
Atlassian website
Atlassian forums
Atlassian Blog
FishEye plugins

Troubleshooting
FishEye Troubleshooting

- After I commit a change to my CVS repository, it takes a long time before it appears in FishEye. — If you do not have a `CVSROOT/history` file, then a commit will not appear in FishEye until after FishEye has done a periodic full scan of your repository. You can configure the period of this scan in the Admin pages.
- FishEye freezes unexpectedly — If your FishEye 2.0 or 2.0.1 instance freezes unexpectedly, this could be caused by a known issue with FishEye and MySQL database technology.
- Fix Out of Memory errors by increasing available memory — Since the default memory setting usually is around 64MB or 128MB, you might have to adjust the settings to run a bigger FishEye instance with sufficient memory.
- Generating a Thread Dump Externally — If Fisheye stops responding or is showing poor performance, providing thread dumps to support can help diagnose the problem.
- I have installed FishEye, and the initial scan is taking a long time. Is this normal? — As a guide, FishEye should be able to process about 100KB-200KB per second on an averaged-size PC. If FishEye is accessing the repository over the network (e.g. over a NFS mount), then you should expect the initial scan to take longer.
- I have installed FishEye, but there is no data in the Changelog. — When you add a repository, FishEye needs to scan through the repository to build up its index and cache. This scan can take some time. Until this scan is complete, you may find that some data is not displayed.
- Initial scan and page loads are slow on Subversion — It's possible that you've mis-configured your tag and branch structure and caused FishEye to process some or all files as trunk files. You should recheck your tag configuration.
- It seems that FishEye's HTTP Header is Too Small
- JIRA Integration Issues
- Message `org.tigris.subversion.javahl.ClientException svn Java heap space` — The Java heap space needs to be increased to an acceptable size. See the FishEye Tuning documentation for more information.
- On my Red Hat Linux system, after running for several days FishEye freezes and does not accept any more connections. — On some Linux systems (particularly RH9), there are socket problems between the JVM and the kernel. To fix this, you need to set the `LD_ASSUME_KERNEL` environment variable before starting FishEye.
- Problems with very long comments and MySQL migration — There is a known issue with FishEye 2.0.x and very long comments when migrating your database to MySQL.
- URLs with encoded slashes don't work, especially in Author constraints — If the author names in your repository contain slashes or back-slashes, and you are using Apache, you may run into a problem where these characters are incorrectly escaped.

After I commit a change to my CVS repository, it takes a long time before it appears in FishEye.

If possible, FishEye will monitor and parse the `CVSROOT/history` file in your repository to quickly work out what has changed. You may want to check with your CVS administrator to ensure this feature of CVS is turned on.

If you do not have a `CVSROOT/history` file, then a commit will not appear in FishEye until after FishEye has done a periodic full scan of your repository. You can configure the period of this scan in the Admin pages.

FishEye freezes unexpectedly
**Issue Symptoms**

If your FishEye 2.0 or 2.0.1 instance freezes unexpectedly, this could be caused by a known issue with FishEye and MySQL database technology.

This issue manifests itself in some FishEye pages returning a server timeout error. To identify the issue, check the FishEye 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
2009-07-15 15:34:45,556 ERROR [btpool0-519] fisheye.app HibernateUtil-commitTransaction - Commit fail msg-1:Lock wait timeout exceeded; try restarting transaction
...
Caused by: java.sql.BatchUpdateException:
Lock wait timeout exceeded; try restarting transaction
 at com.mysql.jdbc.ServerPreparedStatement.executeBatch(S... 163 more
```

The FishEye 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 FishEye instance. This will return FishEye to normal operation.

The FishEye development team is actively working on a solution and this be part of an upcoming point release of FishEye.

**Requesting Support**

If you require assistance in resolving the problem, please raise a support request under the FishEye project.

**Fix Out of Memory errors by increasing available memory**

I am getting Out of Memory errors, how can I allocate more memory to FishEye?

Since the default memory setting usually is around 64MB or 128MB, you might have to adjust the settings to run a bigger FishEye instance with sufficient memory.

On this page:

- Out Of Memory Errors
  - OutOfMemoryError: Java Heap Space
  - OutOfMemoryError: PermGen space, or Permanent Generation Size
  - OutOfMemoryError: unable to create new native thread
  - OutOfMemoryError: GC overhead limit exceeded
  - java.lang.OutOfMemoryError: requested 32756 bytes for ChunkPool::allocate. Out of swap space?

**Out Of Memory Errors**

There are a number of different memory errors that the JVM will throw. The most common are listed as follows. In the following, you will be required to set your memory settings via your FISHEYE_OPTS Environment Variables. You will need to restart your server after setting your FISHEYE_OPTS.

After having set the FISHEYE_OPTS and restarting your server, go to Administration > Sys Info/Support > System Info, and check your JVM Input Arguments to ensure that your server is picking up your FISHEYE_OPTS as expected.

**OutOfMemoryError: Java Heap Space**

If you are running Fisheye/Crucible as a windows service, increasing memory needs to be done in the wrapper.conf file. Refer to the Can Fisheye be run as a Windows Service for instructions.

To solve this error, you will need to add the argument `-Xmx1024m` to FISHEYE_OPTS, in addition to any argument you use to set the heap size. Often you need to increase the amount of memory allocated to fisheye during the initial scan and period and once this is completed you can reduce back down.

```
FISHEYE_OPTS="-Xms128m -Xmx1024m -XX:MaxPermSize=128m"
```

After having set the FISHEYE_OPTS and restarting your server, go to Administration > Sys Info/Support > System Info, and check your JVM Input Arguments to ensure that your server is picking up your FISHEYE_OPTS as
expected.

OutOfMemoryError: PermGen space, or Permanent Generation Size

If you get the error message: java.lang.OutOfMemoryError: PermGen space this means that you have exceeded Java's fixed 64MB block for loading class files. You will need to add the argument -XX:MaxPermSize=128m to FISHEYE_OPTS, in addition to any argument you use to set the heap size.

```
FISHEYE_OPTS="-Xms128m -Xmx512m -XX:MaxPermSize=128m"
```

After having set the FISHEYE_OPTS and restarting your server, go to Administration > Sys Info/Support > System Info, and check your JVM Input Arguments to ensure that your server is picking up your FISHEYE_OPTS as expected.

OutOfMemoryError: unable to create new native thread

This error occurs when the operating system is unable to create new threads. This is due to the JVM Heap taking up the available RAM.

```
Big heaps take away from the space that can be allocated for the stack of a new thread
```

For Linux the maximum heap size of the JVM cannot be greater than 2GB. If you only have 2GB RAM in your server, it is not recommended to set the Max size of the JVM that high.

The size of the stack per thread can also contribute to this problem. The stack size can reduce the number of threads that can be created.

To fix this problem, you should reduce the size of your JVM Heap and also the size of the stack per thread.

```
FISHEYE_OPTS="-Xms128m -Xmx1024m -XX:MaxPermSize=128m -Xss512k"
```

Please refer to this guide as a reference for JVM tuning.

After having set the FISHEYE_OPTS and restarting your server, go to Administration > Sys Info/Support > System Info, and check your JVM Input Arguments to ensure that your server is picking up your FISHEYE_OPTS as expected.

OutOfMemoryError: GC overhead limit exceeded

This error indicates that the JVM took too long to free up memory during its GC process. This error can be thrown from the Parallel or Concurrent collectors.

```
The parallel collector will throw an OutOfMemoryError if too much time is being spent in garbage collection: if more than 98% of the total time is spent in garbage collection and less than 2% of the heap is recovered, an OutOfMemoryError will be thrown. This feature is designed to prevent applications from running for an extended period of time while making little or no progress because the heap is too small. If necessary, this feature can be disabled by adding the option -XX:-UseGCOverheadLimit to the command line.
```

This kind of OutOfMemoryError can be caused if your java process is starting to use swapped memory for its heap. This will cause the JVM to take a lot longer than normal to perform normal GC operations. This can eventually cause a timeout to occur and cause this error.

To overcome this issue, you need to make sure that all processes can't allocate more memory than there is system memory. In practice this is impossible to do for all processes. At a minimum you should make sure that all your jvm's do not have a total maximum memory allocation than your normally available system memory.

Please refer to this guide for more information.

java.lang.OutOfMemoryError: requested 32756 bytes for ChunkPool::allocate. Out of swap space?

Essentially the native objects does not have enough memory to use. This is usually because you have allocated too much memory to your heap reducing the amount available for native objects. See this article.

The solution is to reduce the amount of heap memory you have allocated. For example if you have set -Xmx4096, you should consider reducing this to -Xmx2048m.

Remember if you are using a 32bit JVM you cannot allocate more than -Xmx2048m for linux (and even less for windows). Using a 64 bit JVM can resolve this problem, but is not recommended for fisheye/crucible instances (refer to System Requirements).

Read the Tuning FishEye page for more detail on adjusting resource limits and performance settings in FishEye.

Generating a Thread Dump Externally

If Fisheye stops responding or is showing poor performance, providing thread dumps to support can help diagnose the problem.

Generating a Thread Dump for Windows

To take a thread dump:

2. Click Run for any security warnings
3. Select Process -> Thread Dump
4. Under Process Id, select the '...' button.
5. From the drop-down list, select the Confluence process. Users running Fisheye, select the 'Java ...' option.
6. Ensure that the "Thread dump" and "Keep Remote Thread Running" is selected.
7. Click OK to capture the thread dump.
8. Save the output to a file, eg 'threaddump.log'

If you were asked by Atlassian technical support to create the thread dump, please take 4 thread dumps with a time interval in between (eg. 30 secs) so we can see some patterns. Attach the log file to the support ticket.
Generating a Thread Dump on Linux, including Solaris and other Unixes

Find the process ID of the JVM and use the `ps` command to get list of all processes:

```
kill -3 <pid>
```

**Note:** This will not kill your server (so long as you included the "-3" option, no space in between).
The thread dump will be printed to Fisheye's standard output (fisheye.out).
If you have trouble generating the thread dumps with this method, then use the method "Generating a Thread dump for Windows" as they can also apply for linux, etc.

**Output**

Standard logging for Fisheye is sent to the `FISHEYE_INST/var/log/fisheye-debug.log.*` files, in the `FISHEYE_INST` directory. Thread dumps are an exception since they dump the threads of the entire application server - they'll appear in the `FISHEYE_INST/var/log/fisheye.out` file. You can search for the term "thread dump" in the log file for the beginning of the dump.

**Thread Dump Tools**

- Samurai
- Thread Dump Analyzer TDA TDA 1.0 Final can be obtained from the java.net

**I have installed FishEye, and the initial scan is taking a long time. Is this normal?**

When you add a repository, FishEye needs to scan through the repository to build up its index and cache. This scan can take some time. Until this scan is complete, you may find that some data is not displayed.

As a guide, FishEye should be able to process about 100KB-200KB per second on an averaged-size PC. If FishEye is accessing the repository over the network (e.g. over a NFS mount), then you should expect the initial scan to take longer.

For more details, see Improve FishEye Scan Performance.

**I have installed FishEye, but there is no data in the Changelog.**

When you add a repository, FishEye needs to scan through the repository to build up its index and cache. This scan can take some time. Until this scan is complete, you may find that some data is not displayed.

As a guide, FishEye should be able to process about 100KB-200KB per second on an averaged-size PC. If FishEye is accessing the repository over the network (e.g. over a NFS mount), then you should expect the initial scan to take longer.

**Initial scan and page loads are slow on Subversion**

**Background Information**

When you add a repository, FishEye needs to perform a once-off scan through the repository to build up its initial
index and cache. This scan can take some time. Until this scan is complete, you may find that some data is not displayed. As a guide, FishEye should be able to process about 100KB-200KB per second on an averaged-size PC. If FishEye is accessing the repository over the network (e.g. over a NFS mount), then you should expect the initial scan to take longer. Read on if your scan appears to be considerably slower than expected.

**Solutions**

It's possible that you've mis-configured your tag and branch structure and caused FishEye to process some or all files as trunk files. You should recheck your tag configuration.

If that fails, then the Atlassian support team will be happy to help you with this issue. Please sign up for a support account if you don't already have one, then login and create a FishEye support request.

Users with very large or non-local repositories may be able to improve their FishEye scan performance.

**It seems that FishEye's HTTP Header is Too Small**

If you are receiving errors about FishEye’s HTTP header being too small, it is adjustable. See the page on Setting JVM System Properties for instructions.

**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 FishEye, an error will occur. Depending on the error returned from JIRA, FishEye 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 FishEye debug log, named fisheye-debug.log.YYYY-MM-DD under the dist.inst/var/log folder of your FishEeye installation and 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.

**Message 'org.tigris.subversion.javahl.ClientException svn Java heap space'**

When adding a new repository and on the initial scan, you may receive messages similar to this in the logs: org.tigris.subversion.javahl.ClientException: svn: Java heap space

The Java heap space needs to be increased to an acceptable size. See the FishEye Tuning documentation for more information.

**On my Red Hat Linux system, after running for several days FishEye freezes and does not accept any more connections.**

On some Linux systems (particularly RH9), there are socket problems between the JVM and the kernel. To fix this, you need to set the LD_ASSUME_KERNEL environment variable before starting FishEye.

Add the following code to the script that starts FishEye:

```bash
export LD_ASSUME_KERNEL=2.4.1
```
Problems with very long comments and MySQL migration

Issue Symptoms

There is a known issue with FishEye 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:

```
- 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 this issue is resolved fully. Alternatively, use PostgreSQL or the default (built-in) HSQLDB database.

The FishEye developers are actively working on a solution to this problem and it will be addressed in an upcoming FishEye point release.

Requesting Support

If you require assistance in resolving the problem, please raise a support request under the FishEye project.

URLs with encoded slashes don't work, especially in Author constraints

If the author names in your repository contain slashes or back-slashes, and you are using Apache, you may run into a problem where these characters are incorrectly escaped. By default Apache explicitly forbids encoded slashes or back-slashes in URLs. You can change this behavior with the following httpd.conf directive:
Glossary

Code repository or SCM (Source Code Management) software terminology can be confusing. This page provides definitions for some of the most commonly used terms.

On this page:

- Branch
- Changeset
- Commit
- Committer
- Head
- Repository
- SCM
- Slurp
- Tag
- Trunk

FishEye and its documentation uses the following terms:

**Branch**

A branch is an independent stream of work in a repository. For example, you might copy a set of files in the repository into a new branch, where you can carry on with a separate stream of work without cluttering up the main production area on trunk.

Different SCMs handle branching and merging in different ways. The common aspects allow users to create a branch in which to make changes which do not affect the files in other branches and the trunk development stream. These changes can then be merged into the trunk in a controlled fashion when a development unit of work is complete. Branches can also be used for experimental changes so that these do not affect the main development.

**Changeset**

A changeset is a collection of changes to files in a repository which are committed to the repository in a single operation with a single commit message. Not all SCMs support atomic commit operations. For these SCMs, FishEye will determine the file revisions which make up the changeset using a reliable heuristic (set of rules).

Different SCMs use different terms for the concept of a changeset, such as "changelist", which is generally interchangeable with changeset.

**Commit**

A commit is a single entry of content (usually source code) into a repository. It can be a single file or comprised of multiple file versions.
**Committer**
A committer is a user of an SCM repository who is adding content to the repository (where it will be permanently archived). Typically, a committer is a programmer who is committing source code but SCMs can also store other files such as documents, images and schematics.

**Head**
The "head" revision is the latest change to be made to a file in either a trunk or a branch part of a repository.

**Repository**
A repository is an area managed by an SCM where you store a set of related files, typically from a project or set of projects. The SCM is responsible for version controlling the files in the repository and maintaining their change history. A repository will contain the trunk and all branches for the files of the various projects. A single SCM instance can typically house multiple repositories.

**SCM**
SCM (Source Code Management) software is a category of computer software that archives complex sets of files, for example, all the source code comprised in a large multi-part software project. SCM systems keep copies of every version of every file, allowing you to completely restore and build any version of the software from any point in time.

Committers typically add new versions of code to the SCM once it is tested and approved by peer code review or quality assurance.

Each instance of an SCM can host multiple repositories.

**Slurp**
"Slurping" is a term that is synonymous with "repository scanning". FishEye must do intensive scans of the contents of repositories (SCM systems) in order to provide its lightning-fast web-based browsing of their contents. This can be referred to as a slurp, or slurping.

**Tag**
In SCM terminology, a "tag" is a label that is added to a number of files, to capture their collective state at a particular moment in time. A typical tag would be a specific software version number, that could be referenced to see all the files that belong to a specific version build of a software project.

**Trunk**
In SCM terminology, the "trunk" is the central part of the version control "tree". For example, you might copy a set of files in the repository into a new branch, where you can do new experimental work without cluttering up the main production area on trunk.

**About FishEye**
Your source code repository contains so much useful information, but it is not always easy to extract, interpret or keep up to date.

**How FishEye can Help**
FishEye opens up your repository, helping you to understand your changing source code:
• Track changes to your own, your team's, or everyone's source code.
• Choose to be notified by email and/or RSS feeds.
• View the configurable changelog.
• Use the powerful search functionality
• Construct your own sophisticated queries with EyeQL and integrate the results with other tools via the FishEye API.
• Link to any artifact in your repository: commits, diffs, directories, file histories, revisions, source lines, and search results.
• Analyse your repository via:
  • Line graphs at every node from root to revision.
  • History charts showing branches and tags.
  • File annotations for age and ownership.

Starting Points
For an overview of FishEye's features, take the Feature Tour.
If you are installing FishEye for the first time, read the Quick Start Guide.
For FishEye troubleshooting information, see the FAQ.

What's New in FishEye?
See the FishEye Release Notes.

Known Limitations
• Currently, FishEye does not handle the $Log RCS expansion keyword correctly. Some diff results (and line numbers in diffs) may appear incorrect in files where $Log is used.
• When indexing the content of files, FishEye has an internal limit on the number of tokens/words in the file it can index. Any text past the one-millionth token/word in a file is ignored.

System Requirements
See the FishEye Installation Guide.

Why is it called FishEye?
A fish eye has a wide viewing angle, allowing it to see many things at once. This is a metaphor for how FishEye allows you to easily view the complexity of your source control repositories.