



User Guide for FishEye 2.10

Contents

Getting started	3
Supported platforms	3
End of Support Announcements for FishEye	7
End of Support Announcement for IBM ClearCase	10
End of Support Announcement for Internally Managed Repositories	10
Installing FishEye on Windows	12
Running FishEye as a Windows service	15
Installing FishEye on Linux and Mac	18
Starting to use FishEye	22
Configuring JIRA Integration in the Setup Wizard	27
Using FishEye	35
Using the FishEye Screens	36
Browsing through a repository	38
Searching FishEye	41
Viewing a File	46
Viewing File Content	47
Using Side by Side Diff View	48
Viewing a File History	50
Viewing the changelog	51
FishEye Charts	54
Using favourites in FishEye	58
Changeset Discussions	59
Viewing the commit graph for a repository	59
Viewing People's Statistics	64
Using smart commits	65
Changing your User Profile	70
Re-setting your password	74
Pattern matching guide	75
Date Expressions Reference Guide	76
EyeQL Reference Guide	77

Getting started

FishEye lets you view the contents of your Source Code Management (SCM) repositories in your web browser.

You can:

- view changesets, revisions, branches, tags, diffs and annotations.
- search everything – file names, commit messages, authors, text as well as the source code.
- visualise how source changes were introduced, what changed, when it changed, where it was changed, and who changed it.
- track activity in your source code repository.
- link specific source with related [JIRA issues](#), [Crucible code reviews](#), and [Bamboo builds](#).
- get real-time notifications on code activity via email, RSS, or OpenSocial dashboards.
- construct your own sophisticated queries with [EyeQL](#) and integrate the results with other tools using the FishEye API.

Watch the [video](#) overview of FishEye's features.



To get started with FishEye:

1. Install and start FishEye on either [Windows](#), or [Linux and Mac](#).
2. Work through [Starting to use FishEye](#).
3. Tell FishEye about your [repositories](#).
4. Set up [users and groups](#).








Supported platforms

This page shows the supported platforms for **FishEye 2.10.x** and its minor releases.

Key:  = Supported;  = Not Supported

<i>Java Version</i>		
---------------------	--	--

JRE / JDK	 1.6  1.7	<p>FishEye requires Java Runtime (JDK or JRE), version as noted. Pre-release/Early access versions of the Java Runtime are <i>not supported</i>.</p> <p>You can download a Java Runtime for Windows/Linux/Solaris. On Mac OS X, the JDK is bundled with the operating system.</p> <p>We highly recommend that you use the Oracle JVM (or use the default Mac OS X JVM), as other implementations have not been tested.</p> <p>Please note:</p> <ul style="list-style-type: none"> • Once you have installed the JDK, you must set the JAVA_HOME environment variable. See Installing FishEye on Windows or Installing FishEye on Linux and Mac. • If you are using a 64-bit JVM, please ensure that you've set your max heap size (<code>-Xmx</code>) to a reasonable value, considering the RAM requirements of your system. • If you intend to run FishEye as a Windows Service, using the Java Service Wrapper, we recommend that you use the Java JDK rather than the JRE so as to take advantage of the <code>-server</code> parameter. • You'll need the JDK for the JSP source download. <p>Note also that a bug in Java 1.6.0_29 and above will prevent a connection to an external SQL Server 2008 database without an additional workaround.</p>
Operating Systems		
Microsoft Windows		FishEye is a pure Java application and should run on any platform provided the requirements for the JRE or JDK are satisfied.
Linux		
Apple Mac OS X		

Databases		
MySQL	<div><div>✔</div> MySQL Enterprise Server 5.x</div> <div><div>✔</div> MySQL Community Server 5.x</div> <div>For 5.0, version must be 5.0.21 or later</div> <div>For 5.1, version must be 5.1.10 or later</div>	<div>The FishEye built-in database, running HSQLDB, is somewhat susceptible to data loss during system crashes.</div> <div>External databases (such as MySQL) are generally more resistant to data loss during a system crash.</div> <div>See the FishEye Database documentation for further details.</div>
PostgreSQL	<div><div>✔</div> 8.2, 8.3, 8.4</div>	
Oracle	<div><div>✔</div> 11g</div>	
SQL Server	<div><div>✔</div> 2005, 2008, 2008 R2</div>	
HSQLDB	<div><div>✔</div> Bundled; for evaluation use only</div>	
Web Browsers		
Microsoft Internet Explorer	<div><div>✔</div> 8.0, 9.0</div> <div><div>✖</div> 6.0 and 7.0 are not supported</div>	
Mozilla Firefox	<div><div>✔</div> Latest stable version supported</div> <div><div>✔</div> 3.6, 4.0</div>	
Safari	<div><div>✔</div> Latest stable version supported</div> <div><div>✔</div> 4, 5</div>	
Chrome	<div><div>✔</div> Latest stable version supported</div>	
Version Control Systems		
Subversion	<div><div>✔</div> Server: 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7</div> <div><div>✔</div> Client: SVNKit (default) or native client with JavaHL 1.6</div>	
CVS (and CVSNT)	<div><div>✔</div> All versions</div>	
Perforce	<div><div>✔</div> Client version 2007.3 or later</div>	

Git	✓ 1.7.1.1 or later	Support for internal hosting of Git repositories by FishEye will end on August 14th 2013, after which Git internal hosting will be removed from newer versions of FishEye. Please see End of Support Announcement for Internally Managed Repositories .
Mercurial	✓ 1.5.1 or later	Mercurial 2.1 has a bug that makes it incompatible with FishEye. Please use Mercurial 2.1.1 or later. You should restart FishEye after upgrading Mercurial.

Hardware requirements

FishEye should ideally run on a 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. Note that NFS and SAN are not supported.

Component	Specifications
CPU	1.8GHz or higher, a single core is sufficient. More cores or higher GHz will result in better load-handling ability.
RAM	1GB minimum, 2GB will provide performance "headroom". Your Java heap should be sized at 512MB with the FISHEYE_OPTS environment variable, adjustable up to 1024MB depending on performance.
I/O	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).


i 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.

Repository Technology	Commits	Codebase Size (HEAD of trunk)	FishEye Index Size
-----------------------	---------	-------------------------------	--------------------

Subversion	14386	466 MB in 12151 files	647 MB
CVS	8210	115 MB in 11433 files	220 MB

 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.

Deployment notes for version control systems

Subversion (server)	FishEye can communicate with any repository running Subversion 1.1 or later.
Subversion (client)	FishEye now bundles the SVNkit 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.
Perforce (client)	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.
CVS	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.

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.

WAR deployment

FishEye/Crucible is currently a Java program that runs on its own. It cannot be deployed to web application servers such as WebSphere, Weblogic or Tomcat.

Single sign on with Atlassian Crowd

From version 2.8.x, FishEye is bundled with the Crowd 2.4.1 client library, and supports the Crowd 2.4.x server.

Font size tips

(Especially for Linux users.) For best results you may want to tweak your default monospace font and font-size. The default browser font is usually Courier New which can be hard to read in some browsers. We recommend choosing the same font you use in your IDE and selecting a font size approximately 2 points larger than your variable width font. Firefox 3, Internet Explorer 7 and Safari all have excellent font rendering. It is worth taking some time to tweak your fonts for the best experience.

End of Support Announcements for FishEye

This page contains announcements of the end of support for various platforms and browsers when used with FishEye. This is summarised in the table below. Please see the sections following for the full announcements.

End of Support Matrix for FishEye

Platform	FishEye End of Support
MySQL 5.0	January 2012
PostgreSQL 8.0 and 8.1	January 2012
IBM ClearCase (all versions)	4 April 2012 (announcement)

The table above summarises information regarding the end of support announcements for upcoming FishEye releases. If a platform (version) has already reached its end of support date, it is not listed in the table.

Why is Atlassian ending support for these platforms?

Atlassian is committed to delivering improvements and bug fixes as fast as possible. We are also committed to providing world class support for all the platforms our customers run our software on. However, as the complexity of our applications grows, the cost of supporting multiple platforms increases exponentially. Each new feature has to be tested on several combinations of application servers, databases, web browsers, etc, with setup and ongoing maintenance of automated tests. Moving forward, we want to reduce the time spent there to increase FishEye development speed significantly.

On this page (most recent announcements first):

- [Deprecated Database Support for FishEye \(4 October 2011\)](#)
- [Deprecated Web Browsers for FishEye \(21 March 2011\)](#)
- [Deprecated Java Platforms for FishEye \(21 March 2011\)](#)
- [Deprecated SCM Repository Support for FishEye \(4 April 2011\)](#)

Deprecated Database Support for FishEye (4 October 2011)

This section announces the end of Atlassian support for certain databases for FishEye.

We will **stop supporting older versions of databases** as follows:

- For the next major version of FishEye, in January 2012, support for MySQL 5.0, PostgreSQL 8.0 and 8.1 will end.

Please refer to the [Supported platforms](#) for more details regarding platform support for FishEye. If you have questions or concerns regarding these announcements, please email `eol-announcement` at `atlassian dot com`.

Database	Support End Date
MySQL 5.0	January 2012
PostgreSQL 8.0 and 8.1	January 2012

End of Support Notes for MySQL 5.0 and PostgreSQL 8.0 and 8.1:

- Atlassian intends to end of life support for MySQL 5.0, PostgreSQL 8.0 and 8.1 in January 2012. The release of FishEye after January 2012 will not support MySQL 5.0, PostgreSQL 8.0 or 8.1.
- As mentioned above, the releases of FishEye before January 2012 will contain support for MySQL 5.0

and PostgreSQL 8.0 and 8.1.

Deprecated Web Browsers for FishEye (21 March 2011)

This section announces the end of Atlassian support for certain web browsers for FishEye.

We will **stop supporting older versions of web browsers** as follows:

- From FishEye 2.6, due in May 2011, support for Internet Explorer 7 will end.

The details are below. Please refer to the [Supported platforms](#) for more details regarding platform support for FishEye. If you have questions or concerns regarding this announcement, please email `eol-announcement@atlassian.com`.

End of Life Announcement for Web Browser Support

Web Browsers	Support End Date
Internet Explorer 7	When FishEye 2.6 releases (target May 2011)

Internet Explorer 7 Notes:

- FishEye 2.5 is the last version to officially support Internet Explorer 7.
- FishEye 2.6 is currently targeted to release in May 2011 and will not be tested with Internet Explorer 7. After the FishEye 2.6 release, Atlassian will not provide fixes in older versions of FishEye for bugs affecting Internet Explorer 7.

Deprecated Java Platforms for FishEye (21 March 2011)

This section announces the end of Atlassian support for certain Java Platforms for FishEye.

We will **stop supporting the following Java Platforms**:

- From FishEye 2.6, due in May 2011, support for Java Platform 5 (JDK/JRE 1.5) will end.

We are ending support for Java Platform 5, in line with [Sun's Java SE Support Road Map](#) (i.e. "End of Service Life" for Java Platform 5 dated October 30, 2009). We are committed to helping our customers understand this decision and assist them in updating to Java Platform 6, our supported Java Platform.

The details are below. Please refer to the [Supported platforms](#) for more details regarding platform support for FishEye. If you have questions or concerns regarding this announcement, please email `eol-announcement@atlassian.com`.

End of Life Announcement for Java Platform Support

Java Platform	Support End Date
Java Platform 5 (JDK/JRE 1.5)	When FishEye 2.6 releases (target May 2011)

Java Platform 5 End of Support Notes:

- FishEye 2.5 is the last version to officially support Java Platform 5 (JDK/JRE 1.5).
- FishEye 2.6 is currently targeted to release in May 2011 and will not be tested with Java Platform 5 (JDK/JRE 1.5). After the FishEye 2.6 release, Atlassian will not provide fixes in older versions of FishEye for bugs affecting Java Platform 5 (JDK/JRE 1.5).

Deprecated SCM Repository Support for FishEye (4 April 2011)

This section announces the end of Atlassian support for certain SCM repositories for FishEye. End of support means that Atlassian will remove all functionality related to certain SCM repositories past the specified date. Releases before that date will contain the functionality that supports the SCM, however, Atlassian will fix only critical bugs that affect functionality for that SCM, and will not add any new features for that SCM. After the specified date, Atlassian will not support the functionality in any version of FishEye.

Please refer to the [Supported platforms](#) for more details regarding platform support for FishEye. If you have questions or concerns regarding these announcements, please email `eol-announcement` at `atlassian dot com`.

SCM Repository	Support End Date
IBM ClearCase (all versions)	4 April 2012

IBM ClearCase End of Support Notes:

- Atlassian intends to end of life IBM ClearCase functionality on 4 April 2012. The release of FishEye after 4 April 2012 will not contain any IBM ClearCase functionality.
- As mentioned above, the releases of FishEye before 4 April 2012 will contain support for IBM ClearCase. However, we will only be fixing critical bugs related to IBM ClearCase and will not be adding any feature.
- After 4 April 2012, Atlassian will not support IBM ClearCase functionality in any version of FishEye.

End of Support Announcement for IBM ClearCase

Support in FishEye for [IBM ClearCase](#) ended on **April 4th 2012**. FishEye 2.8, and later versions, do not have support for ClearCase.

We have made these decisions to reduce the testing time required for each release and to help us to deliver market-driven features faster.

You can stay on older versions of FishEye to support your existing installations with ClearCase. However, Atlassian will not be providing any ClearCase-related support for any FishEye version after 4 April 2012, and **has removed all functionality** related to ClearCase from FishEye versions released after April 4th 2012. We are committed to helping our customers understand this decision and to assist you in migrating to a different SCM, if needed.

For more details about the announcement, please refer to this page: [End of Support Announcements for FishEye](#).

End of Support Announcement for Internally Managed Repositories

On **August 14th 2013**, we are ending support for [internally managed repositories](#).

You can stay on older versions of FishEye to support your existing installations with Git repository management. However, Atlassian will **remove all functionality** related to repository management, from FishEye versions released after August 14th 2013. We are committed to helping our customers understand this decision and to assist you in migrating your repositories to one of the two other solutions offered by Atlassian if needed:

- [Stash](#) if you need to host your repositories behind your firewall
- [Bitbucket](#) if you prefer a SaaS hosting solution

Why is repository management being removed?

FishEye was built to enable browsing, searching and visualising source code in various Version Control Systems. With many customers requesting repository management, we have decided to provide a solution on top of FishEye. However, the part of FishEye's architecture that allows it to index different types of repositories and access your Subversion and Git repositories in one place, turned out to not be adequate for a repository management solution.

We have decided to focus on the core strengths of FishEye - browsing, searching and visualizing multiple source code management systems - and strengthen the product around these features. This also has enabled us to deliver a much more focused approach to Git repository management and offer a new solution – [Atlassian Stash](#) – which was build from the ground-up with repository management as a focus.

Going forward FishEye will continue to deliver new features and enhancements to help users browse, search and visualize across different Version Control Systems including Git, Subversion, Mercurial, Perforce and CVS.

My team manages Git repositories in FishEye, how do we migrate?

Here are suggestions to migrating your repositories to **Stash** or **Bitbucket**. The following steps will guide you through the commands that you need to run to migrate your FishEye hosted repositories to a different location.

We will assume that you already have a new repository ready to be used and that you have the latest local copy on your computer. In this case we will use a **Stash** example.

1. Create a new repository on the service you chose (Stash, BitBucket...)

This repository will be used as the new remote for your development.

2. Open your terminal and go to the local copy of the directory that you want to push.

```
cd /path/to/myrepo
```

3. Update the address of your remote origin to point it a the new repository.

```
git remote set-url origin ssh://git@stash.mycompany.com/MYPROJECT/myrepo.git
```

4. Push all the branches to the remote repository.

```
git push --all origin
```

5. Push all the tags to the remote repository.

```
git push --tags origin
```

At this stage all your local branches and tags should be present in your new repository and you can have the same development process as the one you had before.

7. [Index your newly created repository](#) with FishEye to be able to search, track and view report on your source.

Past this point the migration is complete – your repository should be hosted on a different service and indexed by FishEye as an external Git repository.

8. Delete your old managed repository

You will push and pull against your new service and FishEye will index the changes just like for any external repository.

Installing FishEye on Windows

Hey! We're going to install FishEye on Windows. There are a few steps involved, but we think you'll find it easy to follow along. If you are upgrading an existing installation, please refer to the [FishEye upgrade guide](#) instead.

1. Check supported platforms

Better check the [Supported platforms](#) page first; it lists the application servers, databases, operating systems, web browsers and JDKs that we have tested FishEye with, and that we recommend.

Atlassian only officially supports FishEye running on x86 hardware and 64-bit derivatives of x86 hardware.

Related pages:

- [Running FishEye as a Windows service](#)
- [Installing FishEye on Linux and Mac](#)
- [Starting to use FishEye](#)
- [Supported platforms](#)
- [FishEye upgrade guide](#)

2. Check your version of Java


In a command prompt, run this:

```
java -version
```

The version of Java should be **1.6.0** or higher. If you intend to [run FishEye as a Windows Service](#), using the Java Service Wrapper, you should use 32-bit Java (even on a 64-bit machine), and the JDK rather than the JRE (so as to take advantage of the `-server` parameter).

▼ If you don't see Java 1.6.0 or higher, then get Java...

Download and install the Java Platform JDK from [Oracle's website](#).

 *The Java install path should not contain spaces, so don't install into C:\Program Files\Java\. Instead, use a path like C:\Java.*

Now try running 'java -version' again to check the installation. The version of Java should be **1.6.0** or higher.

3. Check that Windows can find Java

Windows uses the JAVA_HOME environment variable to find Java. To check that, in a new command prompt, run:

```
echo %JAVA_HOME%
```

You should see a path to the Java install location that does *not* contain spaces. We recommend that JAVA_HOME should point to the Java executable in your PATH.

▼ If you don't see a path without spaces...

- If you see a path with spaces, like `C:\Program Files\Java\`, then sorry, but go back to 2. and reinstall Java to a location that doesn't have spaces.
- If you don't see a path at all, or if you just see `%JAVA_HOME%`, then set `JAVA_HOME` as follows:

For Windows 7:

1. Go to **Start**, search for "sys env" and choose **Edit the system environment variables**.
2. Click **Environment Variables**, and then **New** under 'System variables'.
3. Enter "JAVA_HOME" as the **Variable name**, and the absolute path to where you installed Java as the **Variable value**. Don't use a trailing backslash. We recommend that `JAVA_HOME` should point to the Java executable in your PATH.
4. Now, in a new command prompt, try running `%JAVA_HOME%\bin\java -version`. You should see the same version of Java as you saw above.

4. Create a dedicated FishEye user (recommended)

For production installations, we recommend that you create a new dedicated Windows user that will run FishEye on your system. This user:

- Should *not* have admin privileges.
- Should be a non-privileged user with read, write and execute access on the FishEye home (install) directory and instance (data) directory. These directories are described below.
- Should only have read access to your repositories.

If you created a dedicated FishEye user, ensure you are logged in as this user to complete the remaining instructions.

5. Now it's time to get FishEye


[Download FishEye](#) from the Atlassian download site.

Extract the downloaded file to an install location:

- Folder names in the path to your FishEye executable should not have spaces in them. The path to the extracted directory is referred to as the `<FishEye home directory>` in these instructions.
- If you expect to have a large number of users for this FishEye installation, and FishEye will be [connected to an external database](#), consider installing FishEye on a different server from the one running the external database, for improved performance.

6. Tell FishEye where to store your data

The FishEye instance directory is where your FishEye data is stored.

 You *should not* locate your FishEye instance directory inside the `<FishEye home directory>` — they should be entirely separate locations. If you do put the instance directory in the `<FishEye home directory>` it will be overwritten, and lost, when FishEye gets upgraded. And by the way, you'll need separate FishEye instance directories if you want to run multiple copies of FishEye.

Create your FishEye instance directory, and then tell FishEye where you created it by setting a `FISHEYE_INST` environment variable, as follows:

For Windows 7:

1. Go to **Start**, search for "sys env" and choose **Edit the system environment variables**.
2. Click **Environment Variables**, and then **New** under 'System variables'.

3. Enter "FISHEYE_INST" as the **Variable name**, and the absolute path to your new FishEye instance directory as the **Variable value**. Don't use a trailing backslash.
4. Now copy the newly extracted <FishEye home directory> /config.xml file to the root of your new FishEye instance directory.

i Note that if FishEye is run as a Windows service using the Java Service Wrapper, FishEye-specific environment variables such as FISHEYE_INST are ignored – these must be set in the wrapper.conf file. See [Running FishEye as a Windows service](#).

If you have a large number of repositories, we recommend you increase the default number of files that FishEye is allowed to open. See the following knowledge base article for more info: [Subversion Indexer Paused with "Too many open files" Error](#).

7. Start FishEye!

In a command prompt, change directory to <FishEye home directory> and run this:

```
bin\start.bat
```

After a few moments, in a web browser on the same machine, go to <http://localhost:8060/> (or, from another machine, type <http://hostname:8060/>, where hostname is the name of the machine where you installed FishEye).

Enter your license, then an admin password, to finish the setup. Note that this password is for the 'built-in' FishEye admin user. You can log in as this user, if necessary, by clicking the **Administration** link in the page footer.

You can postpone setting up JIRA integration until later if you wish; see [Configuring JIRA integration in the Setup Wizard](#).

8. Add repositories

Now you can tell FishEye about any existing repositories you have. Please read [Starting to use FishEye](#) for the details.

FishEye will perform an initial index of your repositories, during which it accesses, indexes and organizes a view of your repositories (including all historical items) back to the earliest commits. If you are evaluating FishEye, we suggest that you index a single project, so you can use FishEye as soon as possible. If you choose to index your entire repository, 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). The basic process is slightly different for each SCM type.

9. Add users and groups

You will want to set up your users and groups in FishEye. You can [add users directly](#) to FishEye, or connect to an [external user directory](#). Please read [Starting to use FishEye](#) for an introduction.

10. Set up your mail server

Configure the FishEye email server so that users can get notifications from FishEye. See [Configuring SMTP](#).

11. Connect to an external database (recommended)

If you intend to use this FishEye installation in a production environment, it is highly recommended that you use one of the [supported](#) external databases. See [Migrating to an external database](#).

If you are evaluating FishEye, or don't wish to do this now, FishEye will happily use its embedded HSQL database, and you can easily migrate later.

12. Stop FishEye (optional)

In a command prompt, change directory to <FishEye home directory> and run this:

```
bin\stop.bat
```

13. Tuning FishEye performance


To get the best performance from your new FishEye installation, please consult [Tuning FishEye performance](#).

Running FishEye as a Windows service

FishEye can be run as a service under Microsoft Windows using a [Java Service Wrapper](#).

The service wrapper provides the following benefits:

- Allows FishEye, which is a Java application, to be run as a Windows Service.
- No need for a user to be logged on to the system at all times, or for a command prompt to be open and running on the desktop to be able to run FishEye.
- The ability to run FishEye in the background as a service, for improved convenience, system performance and security.
- FishEye is launched automatically on system startup and does not require that a user be logged in.
- Users are not able to stop, start, or otherwise tamper with FishEye unless they are an administrator.
- Provides advanced failover, error recovery, and analysis features to make sure that FishEye has the maximum possible uptime.

 Please note that:

- This page should be read in conjunction with [Installing FishEye on Windows](#).
- You should use 32-bit Java to run the service wrapper provided via the link in the install instructions below, even on a 64-bit machine.
- You should use the Java JDK, rather than the JRE, to take advantage of the `-server` parameter, provided in the Wrapper configuration of [wrapper.zip](#), which enables the [Java HotSpot\(TM\) Server VM](#). See the [note](#) below for details.

On this page:

- [Installing the Java Service Wrapper](#)
- [Setting FishEye environment variables for Windows Services](#)
- [Troubleshooting](#)
 - [Extracting files from wrapper.zip](#)
 - [Warning when using 64-bit Java JDK](#)
 - [Wrapper configuration and "-server" parameter](#)

Related pages:

- [Installing FishEye on Windows](#)

Installing the Java Service Wrapper

To install the Java Service Wrapper on Windows:

1. Download wrapper.zip from [here](#).
2. Unzip the wrapper zip file into your <FishEye home directory> (that is, the directory into which FishEye was originally installed). Note, the resulting folder structure should be <FishEye home directory>\wrapper, <FishEye home directory>\wrapper\bin, etc and NOT <FishEye home directory>\wrapper\wrapper, <FishEye home directory>\wrapper\wrapper\bin. The location of the wrapper directory is important.
3. Tell the wrapper where to find the Java JDK by editing the <FishEye home directory>\wrapper\conf\wrapper.conf file, replacing this:

```
# Java Application
wrapper.java.command=java
```

with the following, and comment out the option you don't wish to use:

```
# Java Application

# Option 1: If you have JAVA_HOME defined in your Windows system
environment variables, then you can use:
wrapper.java.command=%JAVA_HOME%/bin/java

# Option 2: If you have multiple JDKs installed, and you don't want to use
a Windows environment variable to specify which one to use, provide the
absolute path to where the JDK is installed (e.g.
C:/Java/jdk1.7.0_05/bin/java):
wrapper.java.command=C:/<path to Java location>/bin/java
```

To get confirmation in the wrapper log that the wrapper is using the correct Java JDK, add the following lines to the wrapper.conf file:

```
# Tell the Wrapper to log the full generated Java command line.
wrapper.java.command.loglevel=INF
```

You can find the logs at <FishEye home directory>\var\log\wrapper.log.

4. Set the FISHEYE_INST environment variable (and other FishEye-specific environment variables) in the <FishEye home directory>\wrapper\conf\wrapper.conf file, following the [instructions](#) below.
5. Install FishEye as a service as follows:
 - a. Open an Administrator command prompt by searching for 'Command prompt' in the Windows Start menu, right-clicking on **Command Prompt** and then choosing **Run as administrator**.
 - b. Change directory to <FishEye home directory>\wrapper\bin and run Fisheye-Install -NTService.bat. If you run into any problems starting the wrapper, you'll find its logs in <FishEye home directory>\var\log\wrapper.log.
6. Start the Fisheye service under the Windows Control Panel; you can search in the Start menu for 'services', and in the list of services, right-click on the 'FishEye' item and choose **Start**. You can also stop the FishEye service in this way.

 Please note that:

- If you make changes to the wrapper.conf file, having already started the service, you need to stop and then restart the service for it to make use of the changed configuration.
- If in future you move the FishEye home directory, you will need to uninstall (using Fisheye-Uninstall -NTService.bat) and then reinstall the FishEye service.

Setting FishEye environment variables for Windows Services

Please note, that if you run FishEye as a Windows service, any FishEye-specific [environment variables](#) must be set in your <FishEye home directory>\wrapper\conf\wrapper.conf file.

If you run into any problems starting the wrapper, you'll find its logs in <FishEye home directory>\var\log\wrapper.log.

If there are other Java parameters you wish to add, then you will need to add them under the additional parameters section, e.g.

```
# JDK Additional Parameters for jmx
wrapper.java.additional.4=-Dcom.sun.management.jmxremote
wrapper.java.additional.5=-Dcom.sun.management.jmxremote.port=4242
wrapper.java.additional.6=-Dcom.sun.management.jmxremote.authenticate=false
wrapper.java.additional.7=-Dcom.sun.management.jmxremote.ssl=false
wrapper.java.additional.8=-Dcom.sun.management.jmxremote.authenticate=false
wrapper.java.additional.9=-Dcom.sun.management.jmxremote.password.file=.
/wrapper/jmxremote.password
wrapper.java.additional.10=-Dwrapper.mbean.name="wrapper:type=Java
Service Wrapper Control"
```

To add the FISHEYE_INST environment variable, the Java MaxPermSize parameter, or the -Xrs options, use the following:

```
wrapper.java.additional.11=-Dfisheye.inst="c:/path/to/FISHEYE_INST"
wrapper.java.additional.12=-XX:MaxPermSize=128m
wrapper.java.additional.13=-Xrs
```

Note that the the -Xrs options should be used when running FishEye as a service under Windows to prevent the JVM closing when an interactive user logs out.

Your memory settings can also be found in this file:

```
# Initial Java Heap Size (in MB)
wrapper.java.initmemory=256

# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=1024
```

Increase these values if you have a large repository or expect to use more memory (init of 256, and a max of 1024 are the default values).

In Fisheye/Crucible 1.6.4 and higher, you can check the JVM input arguments by clicking **System info**, under 'System Settings' in the admin area.

Troubleshooting

Extracting files from wrapper.zip

Some customers have reported trouble running the wrapper. These can be avoided by:

- Uncompressing wrapper.zip with Winzip or WinRar rather than using the **Extract All** command in the Windows right-click contextual menu.
- If the wrapper.zip filename appears green instead of black in Windows Explorer, decrypt it, prior to unzipping its contents, by right-clicking on the file, choose **Properties**, click the **Advanced** button, then clear the **Encrypt contents to secure data** checkbox.

Warning when using 64-bit Java JDK

When using a 64-bit Java JDK with the wrapper obtained via the link in the install instructions above, you may see the following in the wrapper.log file:

```
WARNING - Unable to load the Wrapper's native library 'wrapper.dll'. The file is located on the path
at the following location but could not be loaded:
```

```
C:\installs\service\fisheye28\wrapper\lib\wrapper.dll.
```

```
Please verify that the file is readable by the current user and that the file has not been corrupted in
any way. System signals will not be handled correctly.
```

This is caused by using a 64-bit JDK (even on a 64-bit machine). Changing to a 32-bit version of the JDK will prevent this warning. Community Edition versions of the 64-bit Windows Java Service Wrapper are not currently available.

Wrapper configuration and "-server" parameter

Please note that the wrapper configuration provided above uses the `-server` parameter to enable the [Java HotSpot\(TM\) Server VM](#). This feature is only available if you use the JDK. If you use the JRE you will likely get the following error in your logs:

```
INFO | jvm 1 | 2010/12/20 18:19:28 | Error: missing 'server' JVM at 'C:\Program
Files\Java\jre6\bin\server\jvm.dll'.
```

A common issue is that customers remove the `-server` parameter from the `wrapper.conf` file. Please note that if you do this, the wrapper script will ignore any of the following JVM parameters in the file unless you change the sequence to be in order, starting from `wrapper.java.additional.1`. This is an issue with the wrapper application.

In this situation it's best to install and run FishEye/Crucible with the JDK to get all the advantages of the `-server` functionality. You also need to force the wrapper to use the JDK by specifying the path to the Java JDK in the `wrapper.conf` file, as described in the installation instructions above.

Installing FishEye on Linux and Mac

Hey! We're going to install FishEye on a Linux box, or a Mac. There are a few steps involved, but we think you'll find it easy to follow along. If you are upgrading an existing installation, please refer to the [FishEye upgrade guide](#) instead.

1. Check supported platforms

Better check the [Supported platforms](#) page first; it lists the application servers, databases, operating systems, web browsers and JDKs that we have tested FishEye with, and that we recommend.

Atlassian only officially supports FishEye running on x86 hardware and 64-bit derivatives of x86 hardware.

Related pages:

- [Installing FishEye on Windows](#)
- [Starting to use FishEye](#)
- [Supported platforms](#)
- [FishEye upgrade guide](#)

2. Check your version of Java

In a terminal, run this:

```
java -version
```

The version of Java should be **1.6.0** or higher.

▼ **If you don't see Java 1.6.0 or higher, then get Java...**

Download and install the Java Platform JDK from [Oracle's website](#).

Now try running 'java -version' again to check the installation. The version of Java should be **1.6.0** or higher.

3. Check that the system can find Java

In a terminal, run this:

```
echo $JAVA_HOME
```

You should see a path like `/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/`.

▼ **If you don't see a path to the Java location, then set JAVA_HOME...**

Linux	Mac
<p>Do either of the following:</p> <ul style="list-style-type: none"> If <code>JAVA_HOME</code> is not set, log in with 'root' level permissions and run: <pre>echo JAVA_HOME="path/to/JAVA_HOME" >> /etc/environment</pre> <p>where <code>path/to/JAVA_HOME</code> may be like: <code>/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/</code></p> <ul style="list-style-type: none"> If <code>JAVA_HOME</code> needs to be changed, open the <code>/etc/environment</code> file in a text editor and modify the value for <code>JAVA_HOME</code> to: <pre>JAVA_HOME="path/to/JAVA_HOME"</pre> <p>It should look like:</p> <pre>JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/</pre>	<p>Insert the following in your <code>~/.profile</code> file:</p> <pre>JAVA_HOME="path/to/JAVA_HOME" export JAVA_HOME</pre> <p>where <code>path/to/JAVA_HOME</code> may be like: <code>/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/</code></p> <p>Refresh your <code>~/.profile</code> in the terminal and confirm that <code>JAVA_HOME</code> is set:</p> <pre>source ~/.profile \$JAVA_HOME/bin/java -version</pre> <p>You should see a version of Java that is 1.6.0 or higher, like this:</p> <pre>java version "1.6.0_24"</pre>

4. Create a dedicated FishEye user (recommended)

For production installations, we recommend that you create a new dedicated user that will run FishEye on your system. This user:

- Should *not* have admin privileges.
- Should be a non-privileged user with read, write and execute access on the FishEye home (install) directory and instance (data) directory. These directories are described below.
- Should only have read access to your repositories.

If you created a dedicated FishEye user, ensure you are logged in as this user to complete the remaining instructions.

5. Now it's time to get FishEye

[Download FishEye](#) from the Atlassian download site.

Extract the downloaded file to an install location:

- Folder names in the path to your FishEye executable should not have spaces in them. The path to the extracted directory is referred to as the `<FishEye home directory>` in these instructions.
- If you expect to have a large number of users for this FishEye installation, and FishEye will be [connected to an external database](#), consider installing FishEye on a different server from the one running the external database, for improved performance.

6. Tell FishEye where to store your data

The FishEye instance directory is where your FishEye data is stored.

⚠ You *should not* locate your FishEye instance directory inside the <FishEye home directory> — they should be entirely separate locations. If you do put the instance directory in the <FishEye home directory> it will be overwritten, and lost, when FishEye gets upgraded. And by the way, you'll need separate FishEye instance directories if you want to run multiple copies of FishEye.

Create your FishEye instance directory, and then tell FishEye where you created it by adding a FISHEYE_INST environment variable as follows:

Linux	Mac
Open the <code>/etc/environment</code> file in a text editor and insert: <pre>FISHEYE_INST="path/to/<FishEye instance directory>"</pre>	Open the <code>~/.profile</code> file for the current user in a text editor and insert: <pre>FISHEYE_INST="path/to/<FishEye instance directory>" export FISHEYE_INST</pre>

Now, copy the <FishEye home directory> /`config.xml` to the root of the FISHEYE_INST directory, so that FishEye can start properly.

Also, if you have a large number of repositories, we recommend you increase the default number of files that FishEye is allowed to open. See the following knowledge base article for more info: [Subversion Indexer Paused with "Too many open files" Error](#).

7. Start FishEye

In a terminal, change directory to <FishEye home directory> and run this:

```
bin/start.sh
```

After a few moments, in a web browser on the same machine, go to <http://localhost:8060/> (or, from another machine, type <http://hostname:8060/>, where `hostname` is the name of the machine where you installed FishEye).

Enter your license, then an admin password, to finish the setup. Note that this password is for the 'built-in' FishEye admin user. You can log in as this user, if necessary, by clicking the **Administration** link in the page footer.

You can postpone setting up JIRA integration until later if you wish; see [Configuring JIRA integration in the Setup Wizard](#).

8. Add repositories

Now you can tell FishEye about any existing repositories you have. Please read [Starting to use FishEye](#) for the details.

FishEye will perform an initial index of your repositories, during which it accesses, indexes and organizes a view of your repositories (including all historical items) back to the earliest commits. If you are evaluating FishEye, we suggest that you index a single project, so you can use FishEye as soon as possible. If you choose to index your entire repository, 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). The basic process is slightly different for each SCM type.

9. Add users and groups

You will want to set up your users and groups in FishEye. You can [add users directly](#) to FishEye, or connect to an [external user directory](#). Please read [Starting to use FishEye](#) for an introduction.

10. Set up your mail server

Configure the FishEye email server so that users can get notifications from FishEye. See [Configuring SMTP](#).

11. Connect to an external database (recommended)

If you intend to use this FishEye installation in a production environment, it is highly recommended that you use one of the [supported](#) external databases. See [Migrating to an external database](#).

If you are evaluating FishEye, or don't wish to do this now, FishEye will happily use its embedded HSQL database, and you can easily migrate later.

12. Stop FishEye (optional)

In a terminal, change directory to <FishEye home directory> and run this:

```
bin/stop.sh
```

13. Tuning FishEye performance

To get the best performance from your new FishEye installation, please consult [Tuning FishEye performance](#).

Starting to use FishEye

This page will guide you through the basics of using FishEye. By the end of it you should be able to:

- Create accounts for your collaborators, organize them into groups.
- Add repositories that need to be indexed and setup permissions.
- Use the Commit Graph to trace the history of your code

Assumptions

This page assumes that:

- You have installed and started the latest version of FishEye.
- You are using a [supported browser](#).

On this page:

- [Assumptions](#)
- [Install FishEye](#)
- [Create users in FishEye](#)
- [Add a repository](#)

Related pages:

- [Installing FishEye on Windows](#)
- [Installing FishEye on Linux and Mac](#)
- [Supported platforms](#)
- [Managing your repositories](#)
- [Setting up your Users and Security](#)

Install FishEye

Get FishEye running on your computer. See the details here:

- [Installing FishEye on Linux and Mac](#)
- [Installing FishEye on Windows](#)

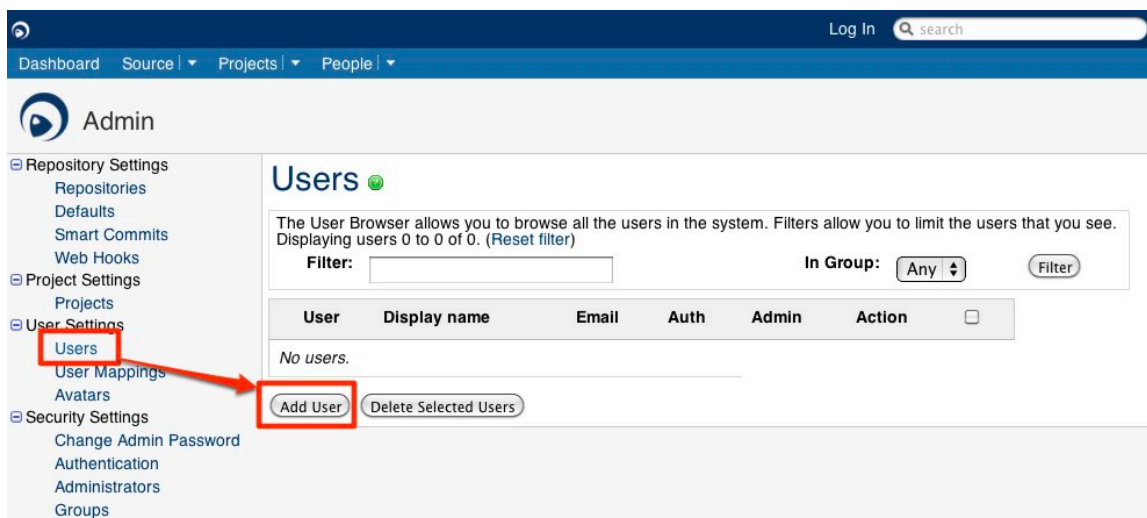
Create users in FishEye

FishEye doesn't have any users created when you install it for the first time. You need to go to the Administration interface to add the first users of the system.

Click on the **Administration** link in the footer.



In the **Users** listing page click **Add User** to go to the user creation form



Fill in the form and create your user

Add New User

Information about the new user

Username

Display name

Email

Auth Type

Password

Confirm Password

From the **User** page you can click on **Create another user** to repeat this operation

User

The user John Doe has been created.
[Create another user.](#)

User Details

Username **jdoe**

Display name **John Doe**

Email **john@doe.com**

Type **built-in**

Admin **false**

[Edit](#) | [Change Password](#) | [Rename](#) | [Delete](#)

Group Details

Groups **No groups.**

[Edit Groups](#)

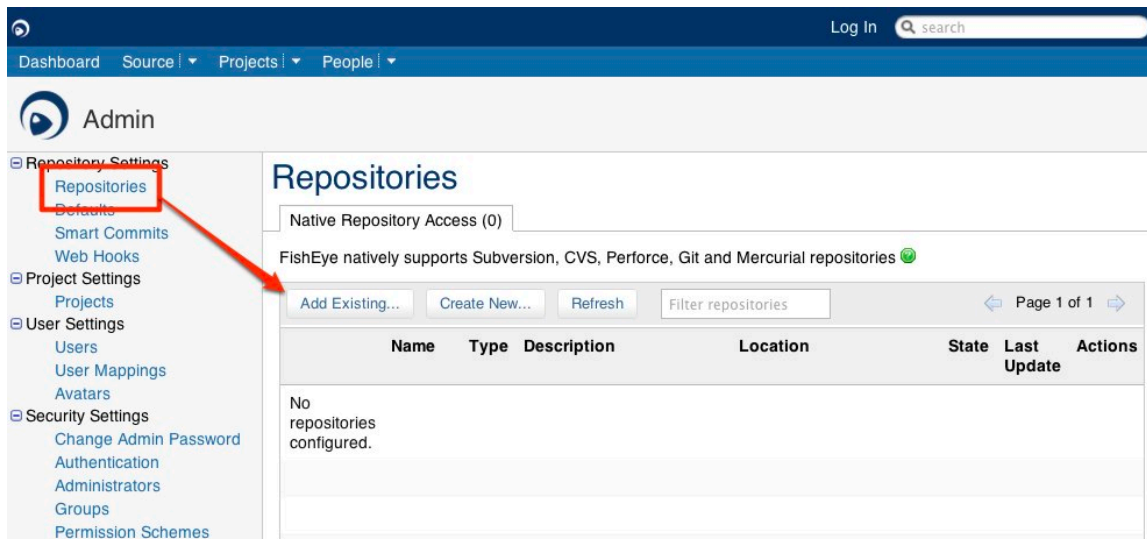
List of users

[Return to the list of users](#)

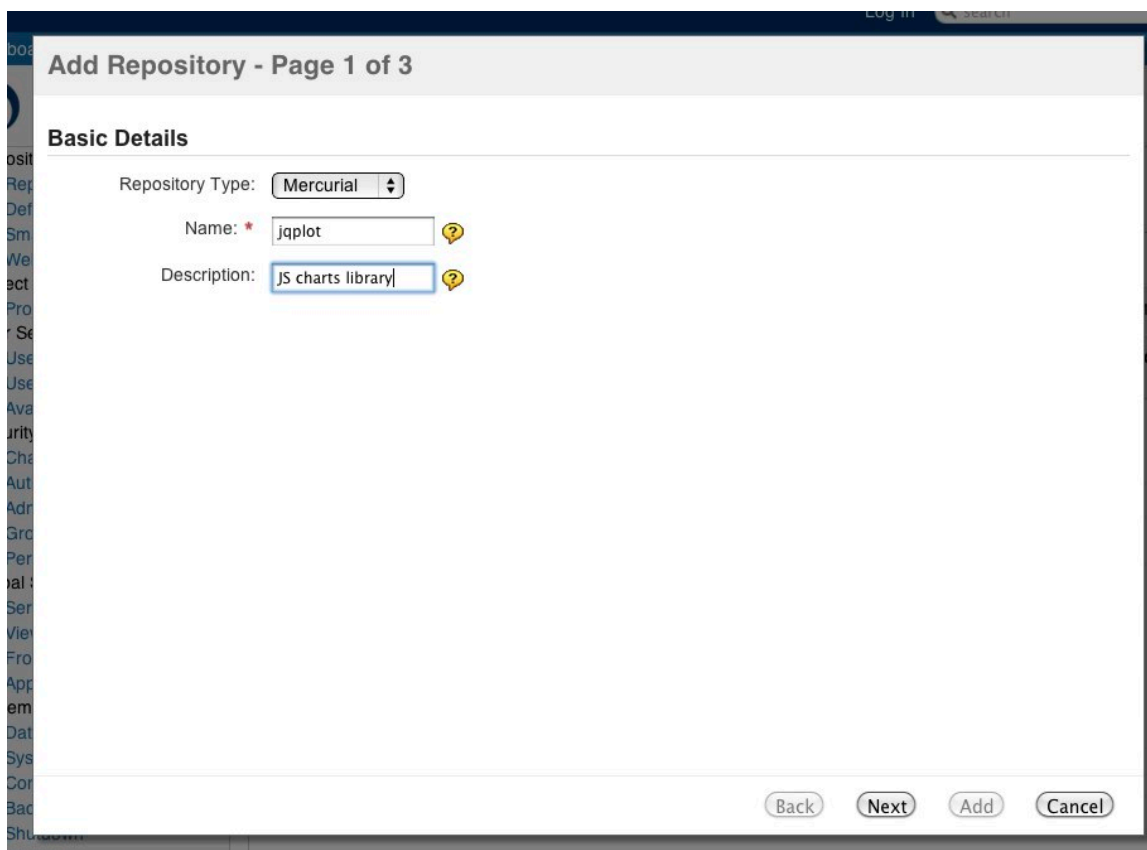
Add a repository

In this section we're going to add a repository to FishEye.

Click on **Add Existing...** in the **Repositories** listing of the Administration.



Choose the repository type and fill in the name and description.



In the repository configuration put the location of your repository. Fill in the authentication details if necessary.

Add Repository - Page 2 of 3

Mercurial Connection Details

Repository Location: * ?

Mercurial Authentication

Authentication Style: ?

FishEye will not add any authentication to requests. Authentication will be handled transparently by the SCM.

☐ Show advanced settings

Finally indicate whether or not you would like diff indexing should be turned on and if the repository should not be indexed right away.

Click **Add** to finish the process.

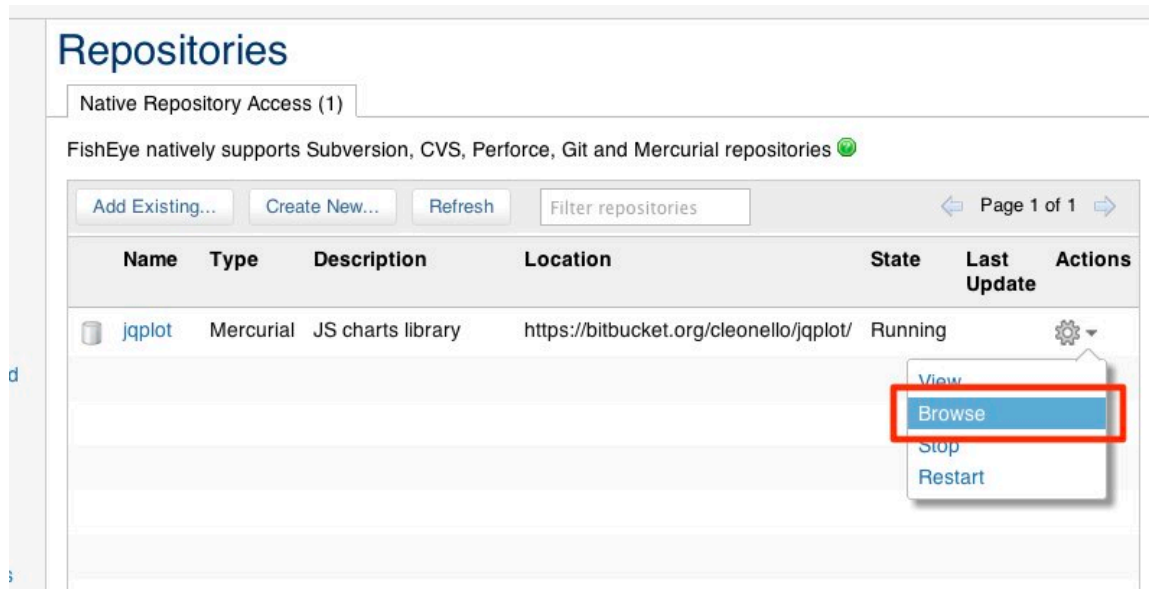
Add Repository - Page 3 of 3

Final Settings

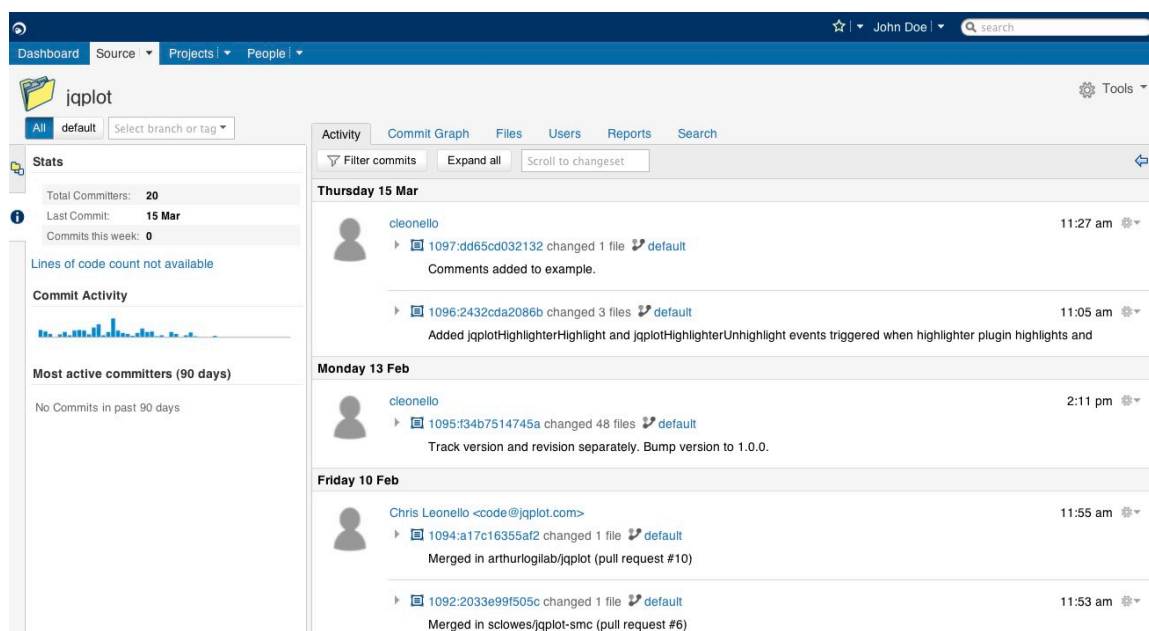
Store Diff Info: ☐ ?

Enable Repository After Adding: ☒ ?

Once created you can click on **Browse** from the repository options menu to access your repository.



You can now browse your files in FishEye, search through your code or track modifications via the commit graph.



Configuring JIRA Integration in the Setup Wizard

This page describes the **Connect to JIRA** tab of the FishEye setup wizard.

You can connect your application to a JIRA server, to manage your users via JIRA and share information with JIRA. When you are installing the application, the setup wizard gives you the opportunity to configure the JIRA connection automatically. This is a quick way of setting up your JIRA integration with the most common options.

You can also configure the JIRA connections via the application administration screens. In that case, you will need to set up connections individually. There are two parts to the integration process:

- A peer-to-peer link between JIRA and the application for sharing information and facilitating integration features. This link is set up via Application Links.
- A client-server link between the application and JIRA for delegating user and group management to your JIRA server.

Requirements: You need JIRA 4.3 or later.

On this page:

- [Connecting to JIRA in the Setup Wizard](#)
- [Troubleshooting](#)
- [Notes](#)

Related pages:

- [Starting to use FishEye](#)
- [JIRA Integration in FishEye](#)
- [Connecting to JIRA for user management](#)
- [User Management Limitations and Recommendations](#)

Connecting to JIRA in the Setup Wizard

To configure JIRA integration while running the FishEye setup wizard:

1. Configure the following setting in JIRA: [Allow remote API access](#).
2. Enter the following information on the 'Connect to JIRA' step of the setup wizard:

JIRA base URL	The web address of your JIRA server. Examples are: http://www.example.com:8080/jira/ http://jira.example.com
JIRA admin username	The credentials for a user with the 'JIRA System Administrators' global permission in JIRA.
JIRA password	
FishEye base URL	JIRA will use this URL to access your FishEye server. The URL you give here will override the base URL specified in your FishEye administration console, for the purposes of the JIRA connection.
Groups to synchronize	Click Advanced Options to see this field. Select at least one JIRA group to synchronize. The default group is <code>jira-users</code> . JIRA will synchronize all changes in the user information on a regular basis. The default synchronization interval is 1 hour.
Admin Groups	Click Advanced Options to see this field. Specify a JIRA group whose members should have administrative access to FishEye/Crucible. The default group is <code>jira-administrators</code> .

3. Click **Connect to JIRA**.
4. Finish the setup process.

1. License

2. Inclusions

3. Connect to JIRA

4. Admin Password

5. Done

Connect to JIRA

If you have JIRA 4.3 or later, Crucible can use JIRA for user management. This is not recommended for more than 500 users. [Learn more here.](#)

JIRA Base URL:

For example: <http://jira.mycompany.com> or <http://mycompany.com/jira>

Admin Username:

This username must have system administrator rights on your JIRA server.

Admin Password:

[Advanced Options...](#)

Troubleshooting

▼ Click to see troubleshooting information...

This section describes the possible problems that may occur when integrating your application with JIRA via the setup wizard, and the solutions for each problem.

Symptom	Cause	Solution
<p>The setup wizard displays one of the following error messages:</p> <ul style="list-style-type: none"> Failed to create application link from JIRA server at <URL> to this <application> server at <URL>. Failed to create application link from this <application> server at <URL> to JIRA server at <URL>. Failed to authenticate application link from JIRA server at <URL> to this <application> server at <URL>. Failed to authenticate application link from <application> server at <URL> to this JIRA server at <URL>. 	<p>The setup wizard failed to complete registration of the peer-to-peer application link with JIRA. JIRA integration is only partially configured.</p>	<p>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>

<p>The setup wizard displays one of the following error messages:</p> <ul style="list-style-type: none"> Failed to register <application> configuration in JIRA for shared user management. Received invalid response from JIRA: <response> Failed to register <application> configuration in JIRA for shared user management. Received: <response> 	<p>The setup wizard failed to complete registration of the client-server link with JIRA for user management. The peer-to-peer link was successfully created, but integration is only partially configured.</p>	<p>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>
<p>The setup wizard displays the following error message:</p> <ul style="list-style-type: none"> Error setting Crowd authentication 	<p>The setup wizard successfully established the peer-to-peer link with JIRA, but could not persist the client-server link for user management in your <code>config.xml</code> file. This may be caused by a problem in your environment, such as a full disk.</p>	<p>Please investigate and fix the problem that prevented the application from saving the configuration file to disk. Then remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>
<p>The setup wizard displays the following error message:</p> <ul style="list-style-type: none"> Error reloading Crowd authentication 	<p>The setup wizard has completed the integration of your application with JIRA, but is unable to start synchronizing the JIRA users with your application.</p>	<p>Restart your application. You should then be able to continue with the setup wizard. If this solution does not work, please contact Atlassian Support.</p>
<p>The setup wizard displays the following error message:</p> <ul style="list-style-type: none"> An error occurred: <code>java.lang.IllegalStateException: Could not create the application in JIRA/Crowd (code: 500)</code>. Please refer to the logs for details. 	<p>The setup wizard has not completed the integration of your application with JIRA. The links are only partially configured. The problem occurred because there is already a user management configuration in JIRA for this <application> URL.</p>	<p>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>
<p>No users can log in after you have set up the application with JIRA integration.</p>	<p>Possible causes:</p> <ul style="list-style-type: none"> There are no users in the group that you specified on the 'Connect to JIRA' screen. For FishEye: There are no groups specified in the 'groups to synchronize' section of your administration console. For Stash: You may not have granted any JIRA groups or users permissions to log in to Stash. 	<p>Go to JIRA and add some usernames to the group.</p> <ul style="list-style-type: none"> For FishEye: Go to the FishEye administration screens and specify at least one group to synchronize. The default is 'jira-users'. For Stash: Grant the Stash User permission to the relevant JIRA groups on the Stash Global permissions page. <p>If this solution does not work, please contact Atlassian Support.</p>

Solution 1: Removing a Partial Configuration – The Easiest Way

If the application's setup wizard fails part-way through setting up the JIRA integration, you may need to remove the partial configuration from JIRA before continuing with your application setup. Please follow the steps below.

Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup wizard:

1. Log in to JIRA as a user with the '**JIRA System Administrators**' global permission.
2. Click the '**Administration**' link on the JIRA top navigation bar.
3. Remove the application link from JIRA, if it exists:
 - a. Click '**Application Links**' in the JIRA administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
 - b. Look for a link to your application. It will have a base URL of the application linked to JIRA. For example:
 - If you want to remove a link between JIRA and FishEye, look for the one where the '**Application URL**' matches the base URL of your FishEye server.
 - If you want to remove a link between JIRA and Confluence, look for the one where the '**Application URL**' matches the base URL of your Confluence server.
 - If you want to remove a link between JIRA and Stash, look for the one where the '**Application URL**' matches the base URL of your Stash server.
 - c. Click the '**Delete**' link next to the application link that you want to delete.
 - d. A confirmation screen will appear. Click the '**Confirm**' button to delete the application link.
4. Remove the user management configuration from JIRA, if it exists:
 - a. Go to the JIRA administration screen for configuring the applications that have been set up to use JIRA for user management:
 - In JIRA 4.3: Click '**Other Applications**' in the '**Users, Groups & Roles**' section of the JIRA administration screen.
 - In JIRA 4.4: Select '**Administration**' > '**Users**' > '**JIRA User Server**'.
 - b. Look for a link to your application. It will have a name matching this format:

```
<Type> - <HostName> - <Application ID>
```

For example:

```
FishEye / Crucible - localhost -  
92004b08-5657-3048-b5dc-f886e662ba15
```

Or:

```
Confluence - localhost -  
92004b08-5657-3048-b5dc-f886e662ba15
```

If you have multiple servers of the same type running on the same host, you will need to match the application ID of your application with the one shown in JIRA. To find the application ID:

- Go to the following URL in your browser:

```
<baseUrl>/rest/applinks/1.0/manifest
```

Replace <baseUrl> with the base URL of your application.

For example:


```
http://localhost:8060/rest/applinks/1.0/manifest
```

- The application links manifest will appear. Check the application ID in the <id> element.
- c. In JIRA, click **Delete** next to the application that you want to remove.
- 5. Go back to the setup wizard and try the 'Connect to JIRA' step again.

Solution 2: Removing a Partial Configuration – The Longer Way

If solution 1 above does not work, you may need to remove the partial configuration and then add the full integration manually. Please follow these steps:

1. Skip the 'Connect to JIRA' step and continue with the setup wizard, to complete the initial configuration of the application.
2. Log in to JIRA as a user with the **'JIRA System Administrators'** global permission.
3. Click the **'Administration'** link on the JIRA top navigation bar.
4. Remove the application link from JIRA, if it exists:
 - a. Click **'Application Links'** in the JIRA administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
 - b. Look for a link to your application. It will have a base URL of the application linked to JIRA. For example:
 - If you want to remove a link between JIRA and FishEye, look for the one where the **'Application URL'** matches the base URL of your FishEye server.
 - If you want to remove a link between JIRA and Confluence, look for the one where the **'Application URL'** matches the base URL of your Confluence server.
 - If you want to remove a link between JIRA and Stash, look for the one where the **'Application URL'** matches the base URL of your Stash server.
 - c. Click the **Delete** link next to the application link that you want to delete.
 - d. A confirmation screen will appear. Click the **Confirm** button to delete the application link.
5. Remove the user management configuration from JIRA, if it exists:
 - a. Go to the JIRA administration screen for configuring the applications that have been set up to use JIRA for user management:
 - In JIRA 4.3: Click **'Other Applications'** in the **'Users, Groups & Roles'** section of the JIRA administration screen.
 - In JIRA 4.4: Select **'Administration' > 'Users' > 'JIRA User Server'**.
 - b. Look for a link to your application. It will have a name matching this format:

```
<Type> - <HostName> - <Application ID>
```

For example:

```
FishEye / Crucible - localhost -  
92004b08-5657-3048-b5dc-f886e662ba15
```

Or:

```
Confluence - localhost -  
92004b08-5657-3048-b5dc-f886e662ba15
```


If you have multiple servers of the same type running on the same host, you will need to match the application ID of your application with the one shown in JIRA. To find the application ID:

- Go to the following URL in your browser:

```
<baseUrl>/rest/applinks/1.0/manifest
```

Replace `<baseUrl>` with the base URL of your application.

For example:

```
http://localhost:8060/rest/applinks/1.0/manifest
```

- The application links manifest will appear. Check the application ID in the `<id>` element.
 - c. In JIRA, click **Delete** next to the application that you want to remove.
6. Add the application link in JIRA again, so that you now have a two-way trusted link between JIRA and your application:
- Click **Add Application Link**. Step 1 of the link wizard will appear.
 - Enter the **server URL** of the application that you want to link to (the 'remote application').
 - Click the **Next** button.
 - Enter the following information:
 - **Create a link back to this server** – Tick this check box to add a two-way link between the two applications.
 - **Username** and **Password** – Enter the credentials for a username that has administrator access to the remote application.
Note: These credentials are only used to authenticate you to the remote application, so that Application Links can make the changes required for the new link. The credentials are not saved.
 - **Reciprocal Link URL** – The URL you give here will override the base URL specified in your remote application's administration console, for the purposes of the application links connection. Application Links will use this URL to access the remote application.
 - Click the **Next** button.
 - Enter the information required to configure authentication for your application link:
 - **The servers have the same set of users** – Tick this check box, because the users are the same in both applications.
 - **These servers fully trust each other** – Tick this check box, because you trust the code in both applications and are sure both applications will maintain the security of their private keys.
For more information about configuring authentication, see [Configuring Authentication for an Application Link](#).
 - Click the **Create** button to create the application link.
7. Configure a new connection for user management in JIRA:
- Go to the JIRA administration screen for configuring the applications that have been set up to use JIRA for user management:
 - In JIRA 4.3: Click **Other Applications** in the **Users, Groups & Roles** section of the JIRA administration screen.
 - In JIRA 4.4: Select **Administration** > **Users** > **JIRA User Server**.
 - Add** an application.
 - Enter the **application name** and **password** that your application will use when accessing JIRA.
 - Enter the **IP address** or addresses of your application. Valid values are:
 - A full IP address, e.g. 192.168.10.12.
 - A wildcard IP range, using CIDR notation, e.g. 192.168.10.1/16. For more information, see the introduction to [CIDR notation on Wikipedia](#) and [RFC 4632](#).
 - **Save** the new application.
8. Set up the JIRA user directory in the application.

- For Confluence:
 - a. Go to the **Confluence Administration Console**.
 - b. Click '**User Directories**' in the left-hand panel.
 - c. **Add** a directory and select type '**Atlassian JIRA**'.
 - d. Enter the following information:
 - **Name** – Enter the name of your JIRA server.
 - **Server URL** – Enter web address of your JIRA server. Examples:


```
http://www.example.com:8080/jira/
http://jira.example.com
```
 - **Application name** and **Application password** – Enter the values that you defined for Confluence in the settings on JIRA.
 - e. Save the directory settings.
 - f. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the '**User Directories**' screen.
For details see [Connecting to Crowd or JIRA for User Management](#).
- For FishEye/Crucible:
 - a. Click **Authentication** (under 'Security Settings').
 - b. Click **Setup JIRA/Crowd authentication**. Note, if LDAP authentication has already been set up, you will need to remove that before connecting to JIRA for user management.
 - c. Make the following settings:

Authenticate against	Select a JIRA instance
Application name and password	Enter the values that you defined for your application in the settings on JIRA.
JIRA URL	The web address of your JIRA server. Examples: <div data-bbox="1021 1348 1361 1480" data-label="Text" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <pre>http://www.example.c om:8080/jira/ http://jira.example. com</pre> </div>
Auto-add	Select Create a FishEye user on successful login so that your JIRA users will be automatically added as a FishEye user when they first log in.
Periodically synchronise users with JIRA	Select Yes to ensure that JIRA will synchronize all changes in the user information on a regular basis. Change the value for Synchronise Period if required.
When Synchronisation Happens	Select an option depending on whether you want to allow changes to user attributes from within FishEye.

Single Sign On	Select Disabled . SSO is not available when using JIRA for user management and if enabled will make the integration fail.
-----------------------	--

- d. Click **Next** and select at least one user group to be synchronised from JIRA. If necessary, you could create a new group in JIRA, such as 'fisheye-users', and select this group here.
 - e. Click **Save**.
 - For Stash:
 - a. Go to the **Stash Administration Console**.
 - b. Click '**User Directories**' in the left-hand panel.
 - c. **Add** a directory and select type '**Atlassian JIRA**'.
 - d. Enter the following information:
 - **Name** – Enter the name of your JIRA server.
 - **Server URL**– Enter web address of your JIRA server. Examples:


```
http://www.example.com:8080/jira/
http://jira.example.com
```
 - **Application name** and **Application password** – Enter the values that you defined for Stash in the settings on JIRA.
 - e. Save the directory settings.
 - f. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the '**User Directories**' screen.
- For details see [Connecting to JIRA for user management](#).

Notes

When you connect to JIRA in the setup wizard, the setup procedure will configure **Trusted Applications authentication** for your application. Please be aware of the following security implications:

- Trusted applications are a **potential security risk**. When you configure Trusted Applications authentication, you are allowing one application to access another as any user. This allows all of the built-in security measures to be bypassed. Do not configure a trusted application unless you know that all code in the application you are trusting will behave itself at all times, and you are sure that the application will maintain the security of its private key.

Using FishEye

- [Using the FishEye Screens](#)
 - [Browsing through a repository](#)
 - [Searching FishEye](#)
 - [Viewing a File](#)
 - [Viewing File Content](#)
 - [Using Side by Side Diff View](#)
 - [Viewing a File History](#)
 - [Viewing the changelog](#)
 - [FishEye Charts](#)
 - [Using favourites in FishEye](#)
 - [Changeset Discussions](#)
 - [Viewing the commit graph for a repository](#)
 - [Viewing People's Statistics](#)

- [Using smart commits](#)
- [Changing your User Profile](#)
 - [Re-setting your password](#)
- [Pattern matching guide](#)
- [Date Expressions Reference Guide](#)
- [EyeQL Reference Guide](#)

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 do the following,

- 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](#)).
- Click the arrow next to the **'Source'** tab to open the dropdown menu. *Logged-in users* will be able to see a list of links to repositories they have recently visited.
- *If you are using Crucible:* Click the **'Projects'** tab to see a list of all projects (see the [Crucible documentation](#)). Click the arrow to open the dropdown menu. *Logged-in users* will be able to see a list of links to projects they have recently visited.
- Click the **'People'** tab to view statistics about committers to your FishEye repositories (see [Viewing People's Statistics](#)). Click the arrow to open the dropdown menu. *Logged in users* will be able to see a list of links to user pages they have recently visited.
- *If you are using Crucible:* Click the **'Reviews'** tab to go to your code reviews (see the [Crucible documentation](#)). Click the arrow to open the dropdown menu. *Logged-in users* will be able to see a list of links to reviews they have recently visited as well as links to the Crucible *'Inbox'* and *'Outbox'*.
- Click the star icon to view your favourite repositories, folders and/or files (see [Using favourites in FishEye](#)).
- 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. The sub-tabs in the 'Source' tab change depending on whether you are viewing a repository or a file.

Viewing a Repository

See [Browsing through a repository](#).

Viewing a File

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 for the file:

Sub-Tab Name	Description
Revisions	Shows the latest revisions of the file. See Viewing a File History .
Files	Shows the contents of the directory.
Activity	<p>Shows recent activity on the item. There are a number of sub-options here (see Viewing the changelog):</p> <ul style="list-style-type: none"> • 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 — 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. <p>¹ If you are using Crucible ² If you are using JIRA</p>
Users	Shows the commit history of the different users that have committed changes on the item.
Reports	Shows activity charts for the item. Various chart options can be selected in the left navigation bar (see FishEye Charts).
Source	Shows the contents of the file.
Query	Allows you to run an advanced search .

Screenshot: The Repositories View

Repositories

Native Repository Access (8) Plugins (5)

FishEye and Crucible natively supports Subversion, CVS, ClearCase, Perforce, Git and Mercurial repositories 🟢

Name	Type	Description	Location	State	Last Update
AGMP	Subversion	a good marketing play	https://studio.atlassian.com/svn/AGMP	Disabled	
applinks	Subversion		https://studio.atlassian.com/svn/APL	Disabled	
atlassian	Subversion		https://svn.atlassian.com/svn/public/atlassian	Running	58 seconds ago
BM	Subversion	Build Monitor	https://labs.atlassian.com/svn/BM	Disabled	
CFS	Subversion		https://studio.plugins.atlassian.com/svn/CFS	Running	50 seconds ago
CGIT	Subversion	Git LightSCM Plugin	https://studio.plugins.atlassian.com/svn/CGIT	Running	50 seconds ago
CLOV	Subversion	Clover	https://studio.atlassian.com/svn/CLOV	Running	28 seconds ago
FE	Subversion	FishEye	https://studio.atlassian.com/svn/FE	Running	5 seconds ago

Info
Indexing has commenced for repository FE

Add...

Screenshot: The Activity View

My Reviews

Inbox (12)

- To Review (4)
- To Summarize (1)
- In Draft (7)
- Require My Approval (0)

Outbox (0)

- Out For Review (0)
- Completed (0)

Archive (5)

- Closed (5)
- Abandoned (0)

My Snippets


- My Open Snippets (1)
- My Snippets (1)

Committer Mappings

0 committer mappings

All Commits Reviews Issues **Expand all** ☒ Show My Activity


Today



Jake
1 min ago


CR-FE-5390 summarized and closed

hey



Seb Ruiz
1 min ago

CR-FE-5599 summarized and closed



Nick Pellow
1 min ago


Re: CR-FE-5593 replied to Tim Pettersen

done. am going to get this merged to default.

25776:e29071439734 changed 1 file 2.6-FECRU-938-review-key-search 1 min ago

FECRU-938: ensure search for review-key doesn't redirect user. even on the review page with a single result.

+5 -0 /test/plugins/fisheye/.../functest/search/ReviewDetailsSearchFuncTest.java



Jason Hinch
1 min ago

CR-FE-5390 finished reviewing

Screenshot: Repository Sub-Tabs — Viewing a File

FE-hg
build.xml

All default Select branch or tag

- idea/
- settings/
- admin/
- amps-jira-plugin/
- artwork/
- build-dependencies/
- bundled-plugins/
- dbdriver/
- dependencies/
- doc/dac/
- etc/
- fe-jar/
- fe-vts-highlighter-plugin/
- fecru-applinks-webitem-plugin/src/
- fecru-gadgets-plugin/src/
- fecru-rest-plugin/
- fisheye-feedback-plugin/
- lib/
- maven-dependencies/
- plugins/

Activity Revisions Users Reports Source

Diff 2 selected Diff latest Filter Include other branches Show all details 1/50

1476 revisions

06 Jun  Pierre-Etienne Poirot
26271:671adca58a4a on 20pc-FECRU-718-user-mgmt
FECRU-729: use a different constant for devmode, as the current one impact all the plugins

06 Jun  Tom Davies
26259:34b5d64024ee on 2.7-davies-FECRU-660-upgrade-to-platform-2.10
FECRU-660: upgrade to atlassian platform 2.10

06 Jun  Jason Hinch
26230:4a05bd93675a on 2.6-FECRU-1054-Quarantine-annotation
FECRU-1054: add Quarantine support to testet aka unit, integration and crucible tests

06 Jun  Michael Studman
26236:48f27811a53 on 2.6-FECRU-1025-gwt-2.3-upgrade
FECRU-1025: Upgrade GWT to 2.3


Browsing through a repository

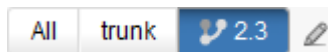
You can use FishEye to [select a repository](#) and browse through it. The repository view provides you with information about the files in the repository, activity occurring related to the repository and users committing to the repository. You can also generate charts and search for specific file revisions in the repository.

On this page:

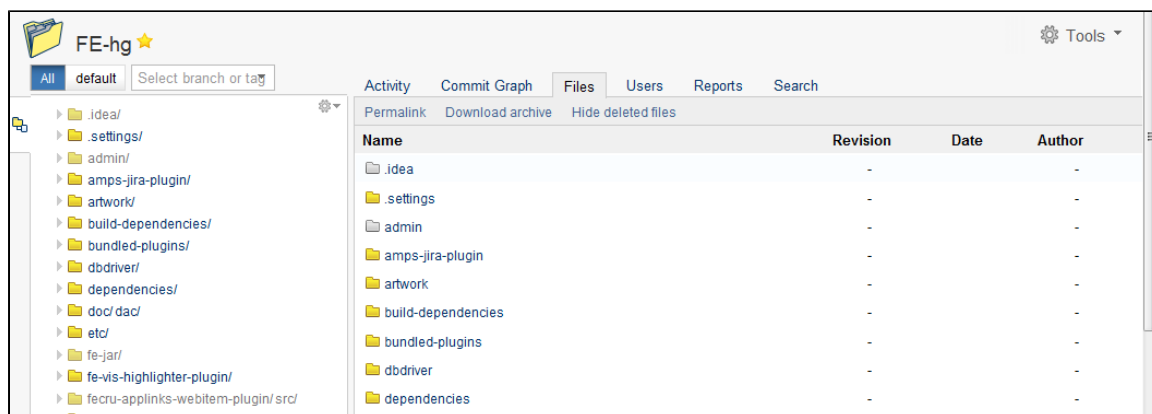
- [Browsing through a repository](#)
- [Hiding empty directories and deleted files](#)
- [Watching a repository](#)

Browsing through a repository**To browse through a repository:**

1. Click the **'Source'** tab. The **'Repositories'** view will be displayed, showing summary information if you have multiple repositories set up.
2. Click the desired repository to view its contents. See the ['Viewing a repository'](#) screenshot below.
3. (optional) Select the [branch](#) or [tag](#) of the repository that you want to view information for, using the branch/tag selector. Only information related to the selected branch/tag will be displayed, when browsing your repositories.
 - Click the tab for the default branch (e.g. 'trunk') to view only information related to it. The default branch (e.g. 'trunk') is determined by FishEye.
 - Click the  icon and enter the desired branch/tag, to view information related to the branch/tag.
 - Click the 'All' tab to view information related to all branches/tags.



4. You can view various information about the repository, as outlined below. If you navigate to a folder, the context of the information below will change. For example, if you navigate to a particular folder in the left navigation tree, the activity, files and users information, reports generated and search results will all relate to that folder.
 - **'Activity'** tab — View the commit, review and issue (requires JIRA) activity related to the repository/folder. The activity stream is similar to the changelog activity stream, see [Viewing the changelog](#) for more information.
 - **'Commit Graph'** tab — Visualise the repository, using the commit graph. See [Viewing the commit graph for a repository](#) for more information.
 - **'Files'** tab — View the contents of the repository/folder being viewed.
 - **'Users'** tab — View the commit history of the users that have committed changes to files in the repository/folder. See [Viewing People's Statistics](#) for more information.
 - **'Reports'** — View activity charts for the repository/folder. See [FishEye Charts](#) for more information.
 - **'Search'** — Search the repository/folder. See [Searching FishEye](#) for more information.
5. Click any file, when browsing the repository, to view information about the file. See [Viewing a File](#) for more information.


Screenshot: Browsing a repository


Hiding empty directories and deleted files

FishEye tracks deleted files for your repository. Deleted files will be greyed out in your left-hand navigation tree. If all of the files in a directory are deleted, the empty directory will also be greyed out. Note, deleted files and empty directories are not removed from your left navigation.

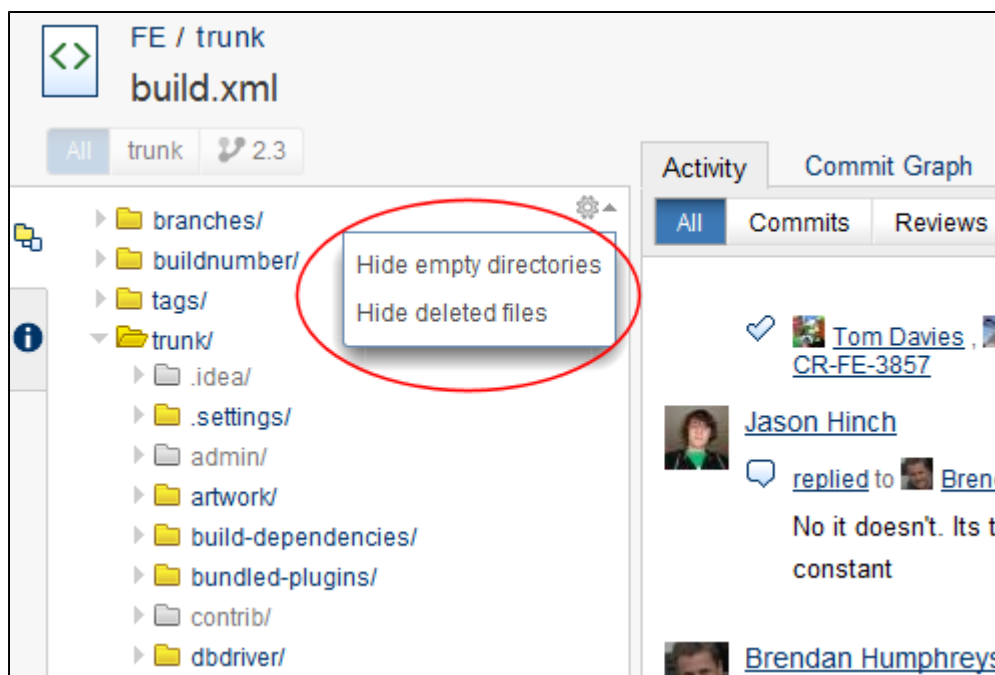
You can choose to hide deleted files and empty directories in the left navigation tree when browsing through a repository, as described below.

To hide deleted directories/files in your navigation tree:

1. Click the **'Source'** tab and browse a repository.
2. In the left hand navigation panel, click the  to show the dropdown menu:
 - Click **'Hide empty directories'** to hide all empty (greyed out) directories and their contents (i.e. deleted files and other empty directories).
 - Click **'Hide deleted files'** to hide deleted (i.e. greyed out) files. This does not affect directories.

 Hence, if you choose to hide both empty directories and deleted files, you will only see files and directories that exist on the [Head](#) of that path. In repositories other than Subversion repositories, this could mean files/directories on any branch.

Screenshot: Hiding deleted files/empty directories



Watching a repository

You can "watch" a repository in FishEye/CRUCIBLE. Watching the repository allows you to receive emails when changes are made to the repository. You can view all of your watches and configure the frequency of your watch emails in your user profile. See [Changing your User Profile](#) for more information.


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

To watch a repository:

1. Navigate to the repository that you want to watch.
2. Click the **'Tools'** menu and click **'Watch'**. The page will reload and a watch will be set up for the repository

(the watch icon will be coloured, not grey).

- If you want to remove the watch, navigate the repository, click the '**Tools**' menu and click '**Watch**'. The watch will be removed (the watch icon will be coloured, not grey).

 You can also remove watches via your user profile.

Searching FishEye

FishEye has a powerful search engine that allows you to find changesets, committers and files. There are two methods for searching in FishEye:

- **Quick Search** — The Quick Search allows you search all repositories connected to FishEye by entering a single search string. This search is the default search and will suggest "quick nav" results (header search box only). Results are weighted by most recent edit date; files edited within the last twelve months are given greater weighting.
- **Advanced Search** — The Advanced Search allows you to search a single repository by entering search criteria against a range of fields, e.g. commit comments, file contents, etc. These parameters can be selected on the standard search interface or specified using the FishEye's custom query language: EyeQL. This search is more complex to use, however you can define more precise search criteria.

On this page:

- [Using the Quick Search](#)
- [Using the Advanced Search](#)
- [Notes](#)

Using the Quick Search

Before you begin:

- Cross-repository searching has a **100-repository limitation** on searches, to prevent it from becoming unresponsive on FishEye instances that have large numbers of repositories. This means that cross-repository Quick Search is not an exhaustive search if you have more than 100 repositories, as only the first 100 repositories (alphabetically, as defined in FishEye) are included. For faster responses, you should limit your search to a particular repository, if possible. FishEye will also limit the search to the specific repository that you are looking at, if you are already navigating within a specific repository
- The Quick Search will also return code reviews, if you are using [Crucible](#) with FishEye. For information on searching Crucible, see [Searching Crucible](#) in the Crucible documentation.

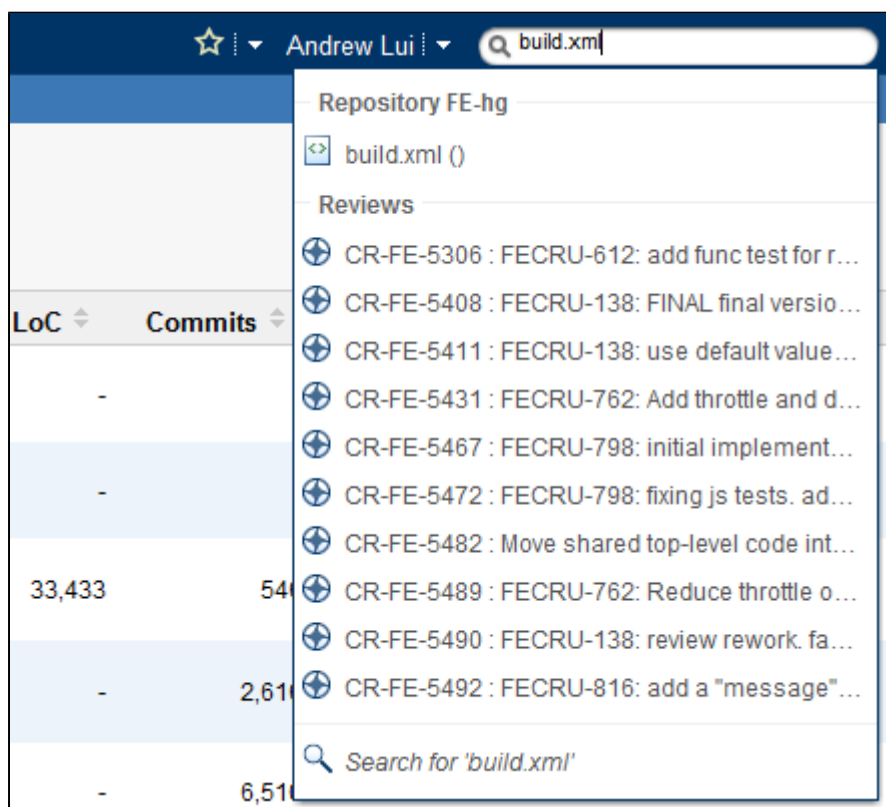
To search FishEye using the Quick Search:

1. Enter your search criteria in the search box in the FishEye header (i.e. Quick Nav). FishEye offers a number of parameters and functions that you can use to refine your expected results, see [Refining your Quick Search Criteria](#) below.
2. "Quick Nav" results will appear in a dropdown, as you type. "Quick Nav" will attempt to match against the file name, repository, committer and username.
 - If you want to use a quick nav result, use the up- and down-arrows on your keyboard and press enter or use your mouse to select the item.
 - If the quick nav results don't have what you are looking for, press enter to run a search. Ensure that no items in the dropdown are selected when you press enter.
3. The Quick Search results page will be displayed. You can filter your results further, as described in [Filtering Quick Search Results](#) below.

Results are sorted by relevance and boosted if they were edited recently. A maximum of 10 results are displayed per page.

 - If you have **integrated your FishEye instance with a JIRA instance**, you can display a summary of any JIRA issues referenced in your search results by hovering over the issue key. For more details, see [JIRA Integration in FishEye](#).
4. If you want to run another search, enter your new criteria in the main search box or in the search box in the header.

Note, only the search box in the header provides "quick nav" results.



Screenshot above: Quick Search displaying "quick nav" matches

Refining your Quick Search Criteria


The FishEye Quick Search has a number of powerful tools that you can use to refine your search criteria before executing the search.

Search Tool	Description	Example
CamelCase Pattern Matching	Enter a CamelCase string pattern and FishEye find files and directories that match the pattern. This is a common search feature in many popular IDEs.	Search for <code>BooQCTest</code> and FishEye will return results like <code>BooleanQueryCoordTest</code> and <code>BooleanQueryClassTest</code> .
Path/File Pattern Matching	Enter a path/file pattern and FishEye will find files and directories that match the pattern.	Search for <code>common/final/Actions</code> and FishEye will return results like <code>/src/common/eu/systemworks/specialprojects/final/Actions.java</code> .

Field Handles	<p>Use a field handle in your criteria to restrict your search to a particular field. Note, you cannot have multiple field handles in a query.</p> <ul style="list-style-type: none"> • <code>file</code> — Search against a file/directory name only. • <code>commit</code> — Search against a commit message only. • <code>diff</code> — Search against a diff (lines added/removed) only. • <code>content</code> — Search against contents of a file only. • <code>committer</code> — Search against a committer only. 	Search for <code>file:build.xml</code> and FishEye will return files that have a name matching <code>build.xml</code> .
Searching within a Directory (AntGlobs)	AntGlobs can be used in the Quick Search to help search within a specific directory.	Search for <code>/src/**/gwt/*.xml</code> and FishEye will return all files with a <code>.xml</code> suffix that are below both a <code>src</code> and a <code>gwt</code> directory. e.g. <code>src/java/com/atlassian/fecru/gwt/FecruCore.gwt.xml</code> but not <code>src/java/com/atlassian/fecru/ApplicationContext.xml</code>
Searching for Discrete Strings	Enter a specific string within quotation marks and FishEye will match against the exact string. Note, this search is not case-sensitive.	Enter <code>"update version in build"</code> and FishEye will only return results that match that exact string, i.e. it will not return a result with <code>update build version</code> or <code>update version in file</code> .

Filtering Quick Search Results

Once you have a set of search results on the Quick Search page, you can filter them to a subset of the original results. The filter controls are in the left panel of the Quick Search page in the 'Source' section.

Filter	Description
Repositories	<p>Select or enter the name of the repository that you want to restrict your results to. For example, if you enter 'FE' then the search results page will refresh to display only files and directories in the 'FE' repository.</p> <p> <i>If you are using Crucible with FishEye, there will be a projects dropdown in the 'Reviews' section. Selecting a Crucible project in this dropdown will not filter the FishEye search results. It is only used to filter reviews returned in the search results. See Searching Crucible.</i></p>

Files and Directories	Click this link to restrict your results to files and directories that have names that match the search criteria.
Changeset Comments	Click this link to restrict your results to changeset comments that match the search criteria.
Diffs	Click this link to restrict your results to diffs (lines added/removed) that match the search criteria.
Content	Click this link to restrict your results to files that have content that match the search criteria.
Committers	Click this link to restrict your results to committers that match the search criteria.

Using the Advanced Search

The Advanced Search can only be run against a specific repository, however you can specify more precise criteria against a number of fields for that repository.

To search a FishEye repository using the Advanced Search:

1. Navigate to the repository that you want to search, as described on [Browsing through a Repository](#).
2. Click the **'Search'** tab.
3. Enter your search criteria, as follows (*Click the **'Switch to EyeQL Search'/'Switch to Standard Search'** link at the bottom of the **'Search Criteria'** panel to switch between the two search methods*):
 - **Standard Search** — Fill in the desired fields in the **'Search Criteria'** panel. See the [Specifying Search Criteria using the Standard Search Interface](#) section below for more details.
 - **EyeQL Search** — Enter your "EyeQL" query. See the [Specifying Search Criteria using EyeQL](#) section below for more details.

Specifying Search Criteria using the Standard Search Interface

The Advanced Search interface allows you to specify search criteria for multiple fields, order the results, group the results and choose the display fields in the results. You can link to the search results, as well as save the results to a CSV file.

Screenshot: Advanced Search Criteria — Standard Search Interface

Please note the following:

- **'Contents of files'** — Files must be non-binary, less than 5MB, and for svn repositories on trunk only. Only HEAD/tip revisions are searched. For older revisions, use the added/removed text search criteria.
- **'File names/paths'** — [Antglobs](#) can be used to specify the criteria for this field.

Specifying Search Criteria using EyeQL

The Advanced Search also allows you to run searches using FishEye's powerful query language, EyeQL.

For information on how to construct an EyeQL query, see the [EyeQL Reference Guide](#). If you haven't built an EyeQL query before, we recommend that you use the [Standard Search Interface](#) interface to build your initial query, then switch to EyeQL to modify that query.

Screenshot above: Advanced Search Criteria — EyeQL Interface

Notes

Related Topics

[Searching Crucible](#) (Crucible documentation)

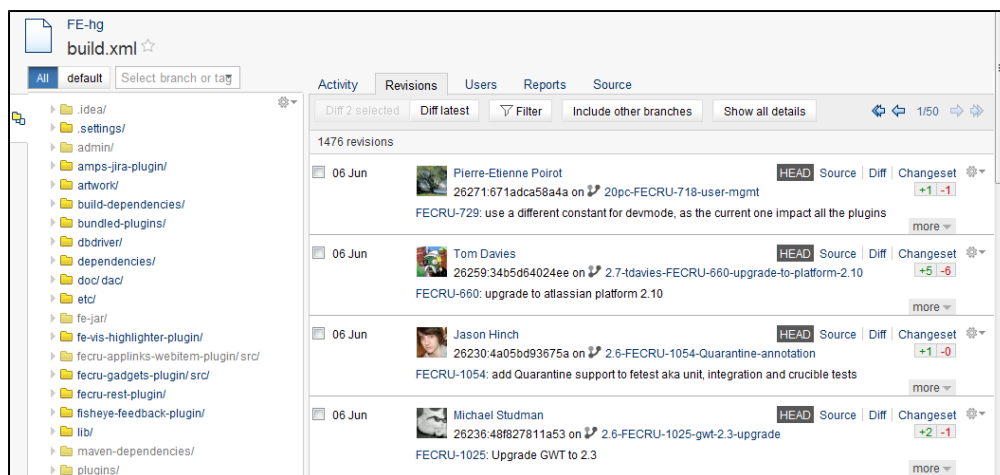
Viewing a File

You can [search](#) or [browse](#) your repositories in FishEye to view a specific file. FishEye provides information about the file history, file content and activity related to the file.

To view a file:

1. Search for a file or browse through a repository to find the file.
2. Click the file name. The file will be displayed, showing the revision history. See screenshot below.
3. View information about the file:
 - **'Activity'** tab — View the changelog for the file. This is the commits, reviews and issues (requires JIRA) activity related to the file. See [Viewing the changelog](#) for more information.
 - **'Revisions'** tab — View the history of revisions for the file. See [Viewing a File History](#) for more information.
 - **'Users'** tab — View the commit history of the users that have committed changes to the file. See [Viewing People's Statistics](#) for more information.
 - **'Reports'** tab — View activity charts for the file. See [FishEye Charts](#) for more information.
 - **'Source'** tab — View the annotated file contents. The raw file can be downloaded via this tab. See [Viewing File Content](#) for more information.

Screenshot: Viewing a File (Revisions)



Related Topics

- [Viewing File Content](#)
- [Viewing a File History](#)
- [Viewing the changelog](#)

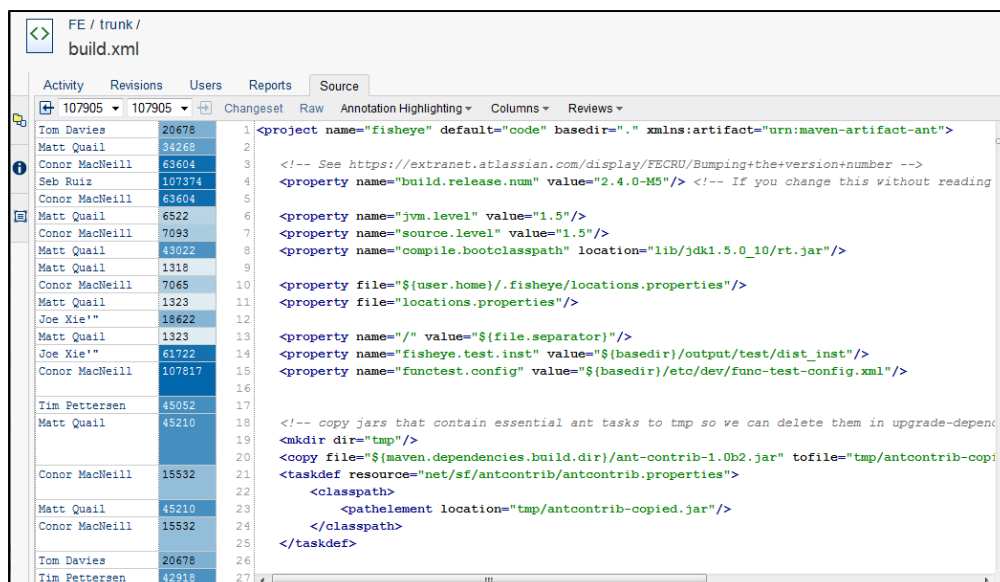
Viewing File Content

You can [search](#) or [browse](#) your repositories in FishEye to view a specific file. FishEye allows you to view and download the source code for the file. You can also view diffs between different revisions of the file and annotations.

To view the source code for a file:

1. Search for a file or browse through a repository to find the file.
2. Click the file name. The file will be displayed, showing the revision history.
3. Click the **'Source'** tab. The source code for the file will be displayed.
 - Select the revision numbers (e.g. '107905') from the two revision dropdowns to [display the diff](#) between the two revisions. By default, the latest revision of the source code is displayed (without any diff).
 - Click the **'Changeset'** link to view the changeset that the revision was a part of.
 - Click the **'Raw'** link to download the raw source code for the file.
 - Click the **'Annotation Highlighting'** dropdown and select **'Age'**, **'Author'** or **'None'** to colour the annotations by age, author or remove highlighting respectively. The highlights are displayed over the revision numbers, next to the authors.
 - Click the **'Columns'** dropdown and select the columns to display: **'Author'**, **'Revision'** and **'Line Number'**.
 - Click the **'Reviews'** dropdown and select **'Create Review'** to create a Crucible review from the file.

Screenshot: Viewing a File (Source)



Using Side by Side Diff View

This page contains instructions for FishEye's innovative '**side by side diff**' view. This is an assistive source code viewing mode where you can see how a file's content has changed, compared on the left and right hand sides of the screen.

On this page:

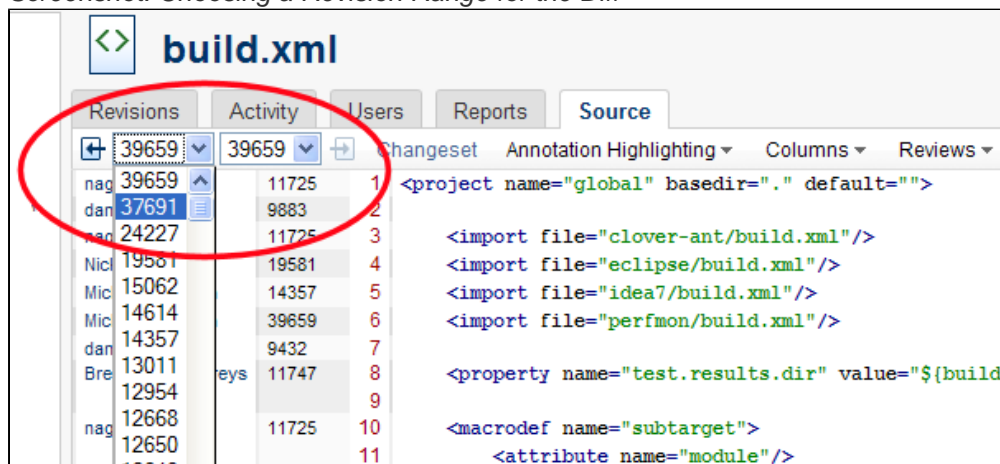
- [Opening side by side diff view](#)
- [Understanding side by side diffs](#)
- [Alternative ways to open side by side diffs](#)
 - [From the FishEye Dashboard](#)
 - [From the Revisions History view](#)

Opening side by side diff view

To open FishEye's side by side diff view:

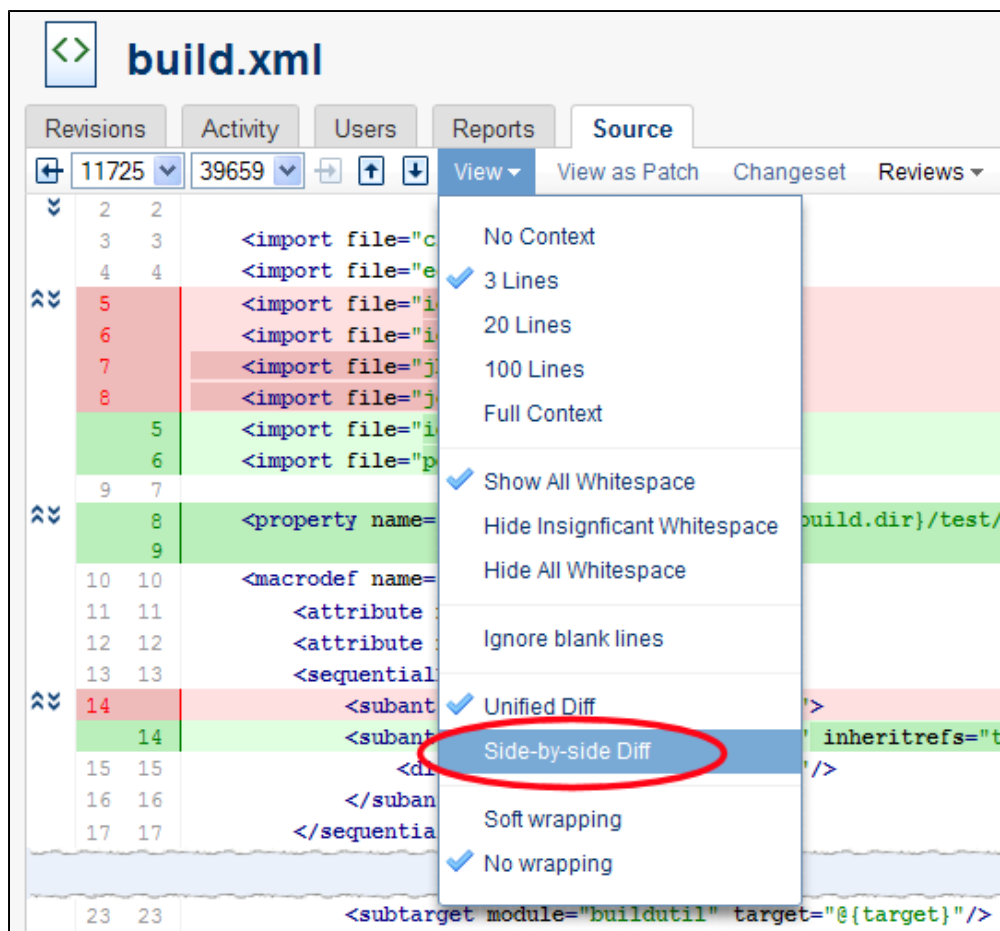
1. Open the source code view for the file in question.
2. Select a range of revisions to compare between.

Screenshot: Choosing a Revision Range for the Diff



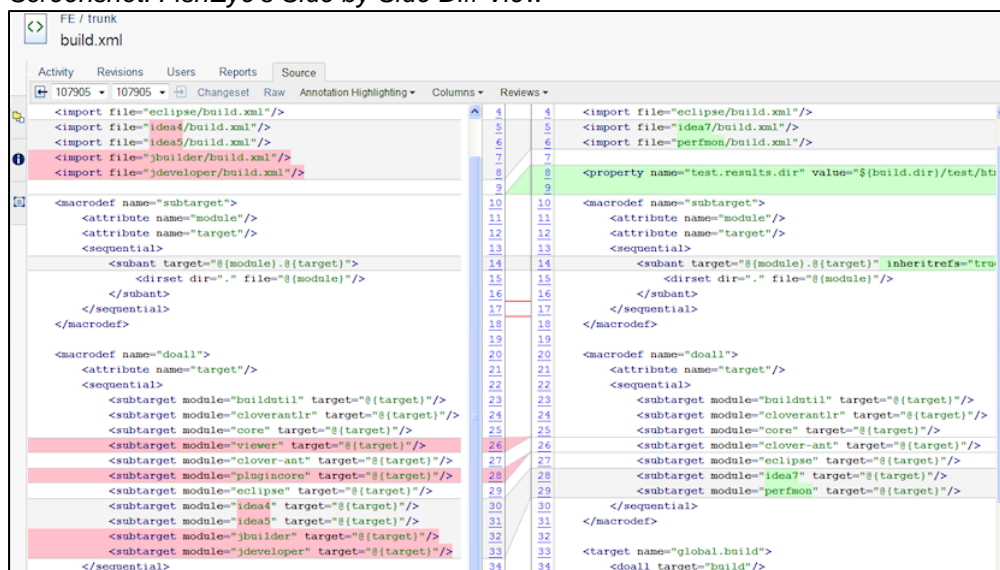
3. Click the '**View**' menu, then select '**Side by Side Diff**'.

Screenshot: Choosing Side by Side Diff Mode



4. **Side by side diff view** opens. The left and right panes can scroll independently, and the view stays anchored around a central point. Colour coding is used to illustrate where lines have been added (green highlights) and where lines have been removed (red highlights). Grey highlights indicate that a line's internal content has changed. Each addition or deletion is linked to the opposite window by a coloured triangle that links to the location of that change in the counterpart file.

Screenshot: FishEye's Side by Side Diff View



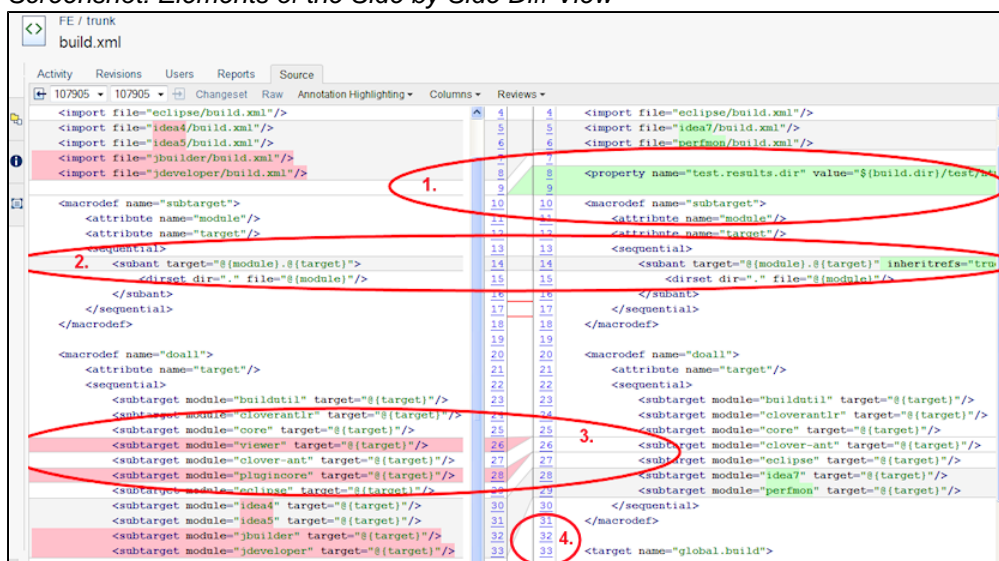
Understanding side by side diffs

Features of the side by side diff screen are referenced in the annotated screenshot below.

1. Added lines are highlighted green, displayed in the right hand pane.

2. Edited lines are highlighted grey, with minor sections highlighted red and green to show deletions and additions.
3. Deleted lines are highlighted red, displayed in the left hand pane.
4. Gutter line numbers are permanent links ("permalinks") that can be saved and sent to colleagues. When they open those links, the view will automatically open in side by side diff mode.

Screenshot: Elements of the Side by Side Diff View



Alternative ways to open side by side diffs

From the FishEye Dashboard

You can also open side by side diffs from the Dashboard screens, by clicking the **'Delta'** triangle icon next to an item when it appears in the stream of events. This will open the file in the diff view. If you have currently selected side by side diff as the viewing mode, then the diff will automatically be displayed in that mode. If not, you can select side by side diff from the **'View'** menu.

From the Revisions History view

When in the revisions view, you can show a diff by checking the boxes next to two revisions, then clicking the **'Diff'** button in the top control bar. If you have currently selected side by side diff as the viewing mode, then the diff will automatically be displayed in that mode. If not, you can select side by side diff from the **'View'** menu.

You can also launch into a diff of the latest revision and the second most recent by clicking **'Latest Diff'** in the top control bar.

Viewing a File History

You can view a specific file when [browsing a repository](#). This allows you to see information about the file, including the history of file revisions.

To view the history of revisions for a file:

1. Log into FishEye/Crucible. The Dashboard will be displayed.
2. Search for the desired file or browse to it as follows:
 - a. Click the **'Source'** tab in the header at the top of your screen. The list of repositories set up in your FishEye instance will be displayed.
 - b. Click the name of the repository that your file is located in.
 - c. Browse to the file using the tree in the left menu and click the file name.
3. Click the **'Revisions'** tab. The history of revisions for the file will be displayed. See the ['File Revisions'](#) screenshot below.
 - Tick the checkboxes next to two file revision and click **'Diff 2 selected'** to [view the diff](#) of the two selected file revisions. Click **'Diff latest'** to [view the diff](#) of the two most recent file revisions.

- Click **'Filter'** to view the file filter. Enter the desired fields to filter the file history results on.
- Click **'Include other branches'** to include revisions of the same file on other branches.
 ⓘ A file can have many physical paths, all of which relate to the same filename in your project structure, or repository's logical structure. This applies to Subversion's branching and tagging directory structure in particular.
- Click **'Show all details'** to expand all file revisions to show additional information including the revision ID, parents and the branch where it is head, denoted with this graphic: **HEAD**.
- See the ['Overview of a File Revision'](#) diagram below for information about the individual revisions.

Screenshot: File Revisions

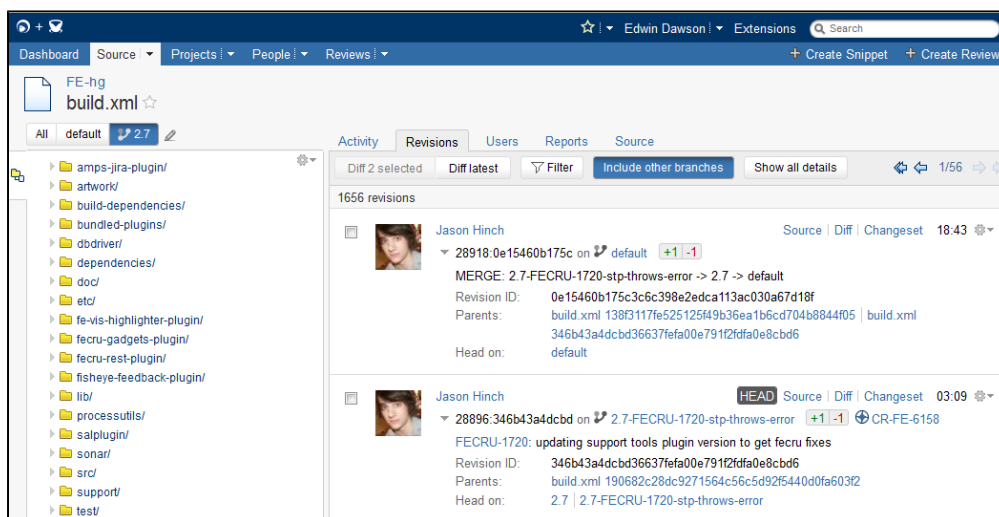
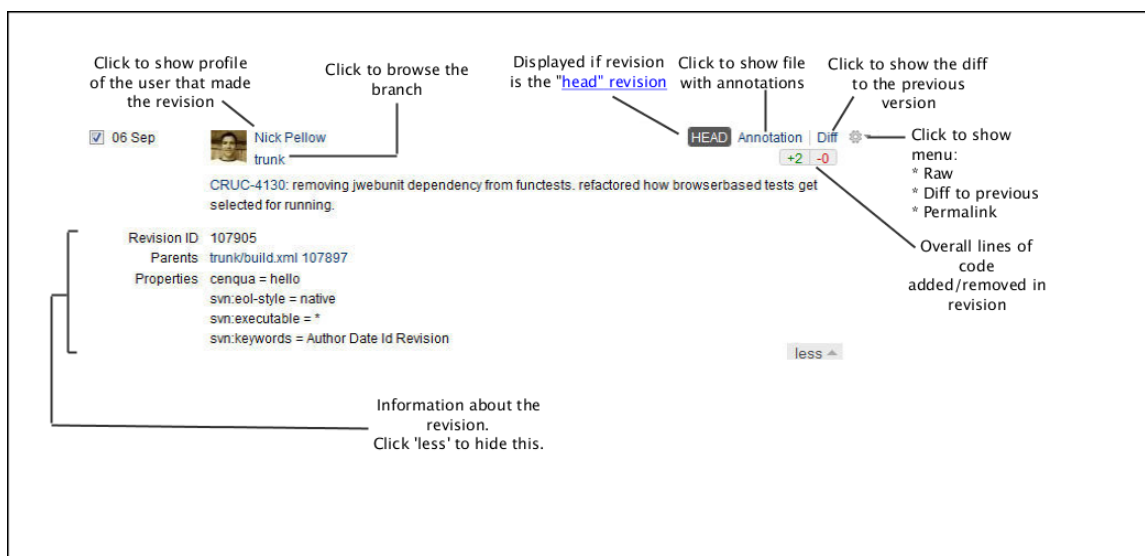


Diagram: Overview of a File Revision



Viewing the changelog

The changelog is a record of the commits and reviews for a repository, branch, directory or file. You can view the recent activity in the changelog when [browsing a repository/branch/directory](#) or [viewing a file](#).

On this page:

- [Viewing changelog activity](#)
- [Filtering commit activity in the changelog](#)
- [Watching the changelog activity](#)

Viewing changelog activity

To view the changelog activity for a repository, branch, directory or file:

1. [Browse to the desired repository, branch, directory](#) or [view the desired file](#).
2. If required, use the selector (under the repository name) to choose the branch or tag that you want to browse the changelog for.
3. Click the **Activity** tab. The recent changelog activity of your repository/branch/directory or file will be displayed.
4. Use the following functions of the **Activity** tab to see different aspects of the changelog activity:

All	Click to show commits, reviews (requires integration with Crucible) and JIRA issues (requires integration with JIRA) activity in the activity stream.
Commits	Click to show only commits in the activity stream.
Reviews	Show only review activity. (Requires integration with Crucible)
Filter commits	See Filtering commit activity in the changelog (below) for more information.
Expand all	Show all modified files related to each changeset.
Scroll to changeset	Type a changeset ID (e.g. 107856) and press Enter to scroll to the that changeset in the activity stream.

Screenshot: Viewing the changelog activity for a file

The screenshot shows the FishEye web interface. The top navigation bar includes links for Dashboard, Source, Projects, People, and Reviews. The main content area is titled 'FE-hg build.xml' and shows the 'Activity' tab selected. The activity stream displays a list of recent changesets, including commits and reviews, with details such as the user, changeset ID, and description. The sidebar on the left shows a file tree structure for the repository.

Filtering commit activity in the changelog

You can filter the commits that are displayed in the activity stream, that is, the commits in the **All** and **Commits** tabs of the **Activity** tab. Note that you cannot use the commits filter to filter reviews.

To filter commit activity:

1. Go to the **Activity** tab, as described above.
2. When viewing either the **All** or **Commits** tab, click **Filter commits**.
3. Enter filtering criteria for the commits to be displayed:

Committer	Changesets checked in by the given committer/author.
Log Comment	Changesets where the commit comment matches the given text.
File Extension	Changesets that contain files with the specified file extension.
File Name	Changesets that contain a given file.
Start Date	Changesets created on or after that date. Must be of the form YYYY-MM-DDTHH:mm:ss, YYYY-MM-DD, YYYY-MM or YYYY (you can use '/' instead of '-').
End Date	Changesets created on or before that date. Must be of the form YYYY-MM-DDTHH:mm:ss, YYYY-MM-DD, YYYY-MM or YYYY (you can use '/' instead of '-').

4. Click **Apply**.
 - Use **Clear** to clear the filter.
 - Click **Filter commits** again to turn off the filter.

Screenshot: Using the Filter

The screenshot shows the FishEye web interface. At the top, there are tabs for 'Activity', 'Revisions', 'Users', 'Reports', and 'Source'. The 'Activity' tab is selected. Under 'Activity', there are sub-tabs: 'All', 'Commits', 'Reviews', and 'Issues'. The 'Filter commits' button is highlighted in blue. Below these tabs, there is a form with the following fields: 'Committer:' (a dropdown menu), 'Log Comment:' (a text input), 'File Extension:' (a text input), 'File Name:' (a text input), 'Start Date:' (a date picker), and 'End Date:' (a date picker). An 'Apply' button is located at the bottom right of the form.

Watching the changelog activity

You can "watch" a changelog's activity stream in FishEye and Crucible. Watching the activity stream allows you to receive emails when updates occur in the activity stream. You can view all of your watches and configure the frequency of your watch emails in your user profile. See [Changing your User Profile](#) for more information.

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

To watch an activity stream:

1. Navigate to the activity stream that you want to watch.
2. Choose **Tools** > **Watch**. The page will reload and a watch will be set up for the activity stream (the watch icon will now be coloured, not grey).
 - To remove the watch, from the activity stream, choose **Tools** > **Watch**. The watch will be

removed. You can also remove watches from your user profile.

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](#)
- [Notes](#)

Charts

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

Setting	Explanation	Values	Default
Branch/Tag selector	Limits the chart to the selected branch/tag.	Any branch/tag from the current repository.	Displays the default/trunk.
Author	Limits the chart to show specific author(s).	Any author configured in the system.	All
Extension	Limits the chart to show specific file type(s).	Any file extension; e.g. '.java'.	All
Chart type	Changes the chart's presentation.	Area, line, pie or change* chart.	Area
Show by	Secondary data by which to refine the chart.	Subdirectory, author or extension.	None
Start Date	Date of the earliest data to show.	Date in format YYYY-MM-DD.	None (show all)
End Date	Date of the latest data to show.	Date in format YYYY-MM-DD.	None (show all)
Y Axis	Choosing ' Tight ' zooms in the charts view to the limits of the range that the data covers. Only applies to Line charts.	Full or Tight	Tight
Sub Directories	Limit the chart to a folder under the current branch. Files in the current directory are represented by an element labelled ' .(this dir) '.	A single folder.	None (show all)

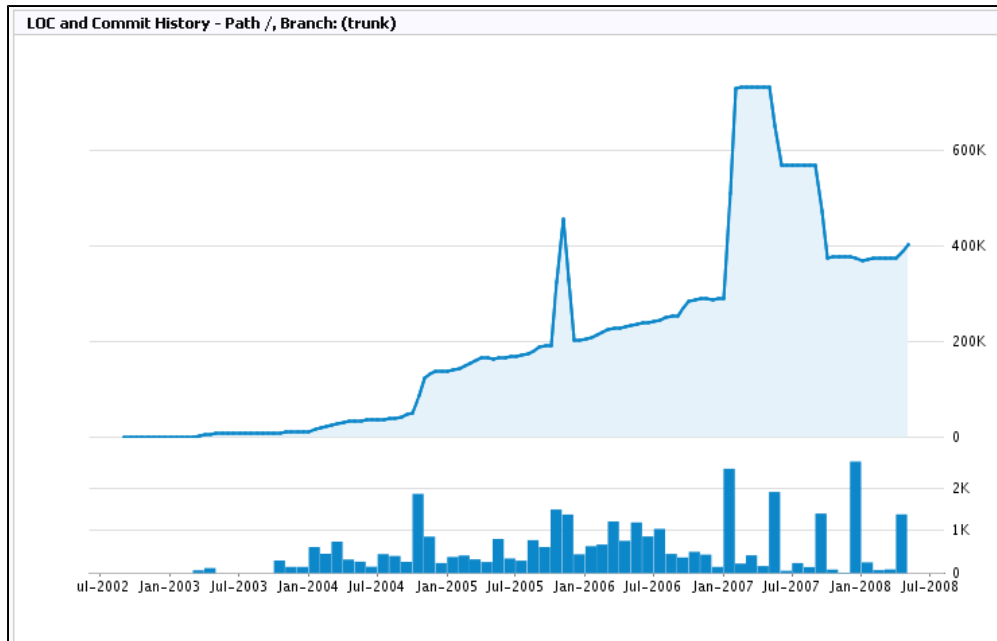
i *The '**change**' chart displays the change in lines of code, for a specific date range, expressed as a line graph. For example, if the lines of code at the start date is 100, the start point will be zero and the rest of the graph shifted by 100 lines.

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

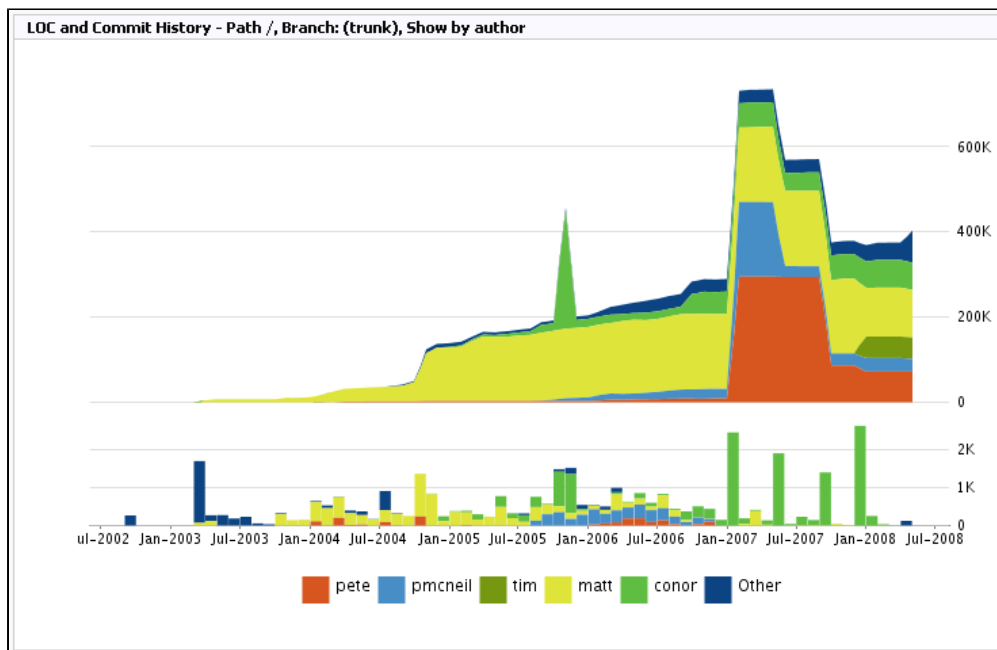
Screenshot: FishEye custom chart settings

The screenshot shows the 'Chart types' and 'Chart options' settings in FishEye. At the top, there are tabs for 'All' and 'trunk', and a dropdown for 'Select branch or tag'. Under 'Chart types', there are radio buttons for 'Area', 'Line' (selected), 'Pie', and 'Change'. Under 'Chart options', there is a 'Break down by:' section with radio buttons for 'None' (selected), 'Subdirectory', 'Author', 'User', and 'Extension'. Below this is a 'Date Range:' section with 'Start Date:' and an empty input field, followed by a colon and another empty input field. There are two dropdown menus: 'Author(s):' with options '(any)', 'Adam Ahmed (aahmed)', 'Anna Buttfield (abuttfield)', 'Ben Speakmon (bspeakmon)', and 'Brendan Humphreys (bhumphreys)'; and 'File Extension:' with options '(any)', 'None', '.MF', '.TXT', and '.cfs'. An 'Apply' button is at the bottom right.

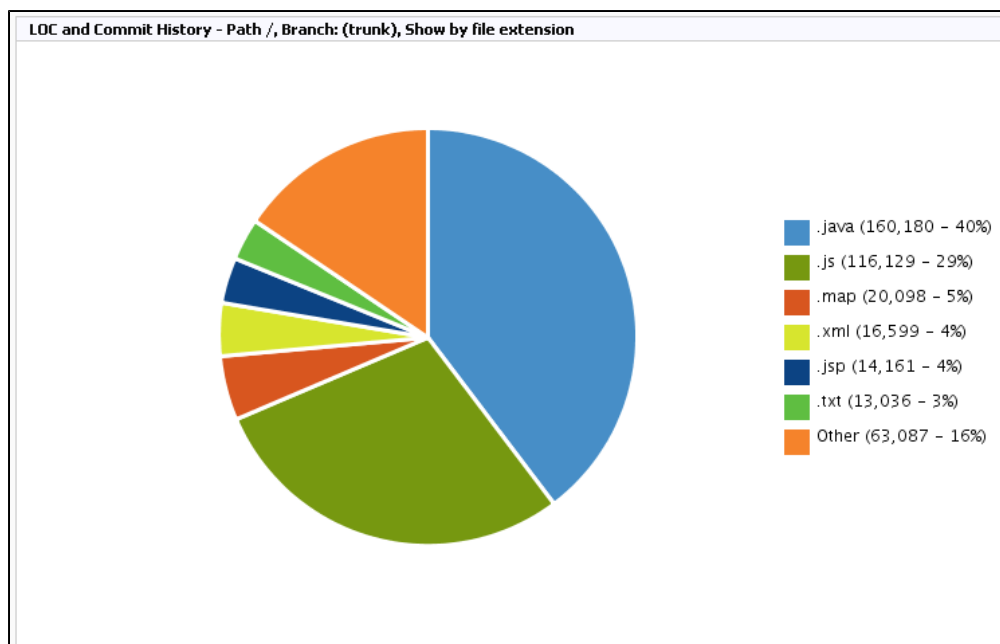
Screenshot: FishEye per-author LOC chart



Screenshot: FishEye per-author LOC chart showing multiple authors



Screenshot: FishEye LOC chart by file extension



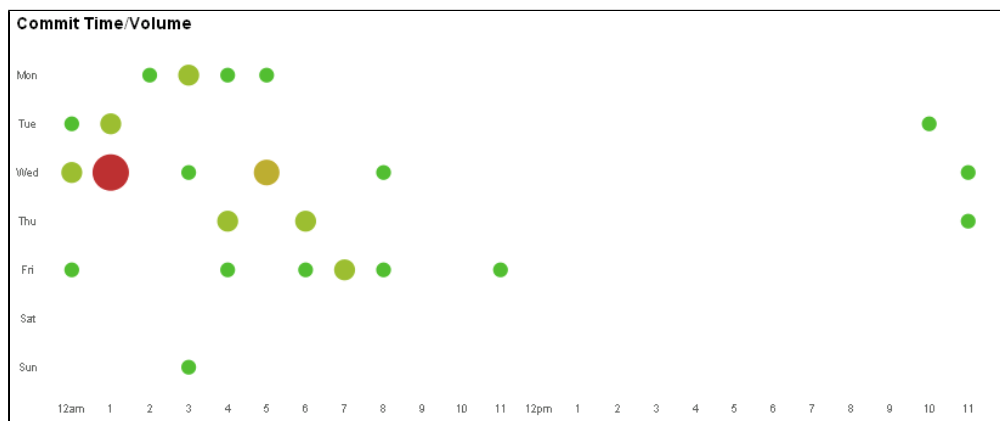
Per-Author Lines of Code Statistics

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.

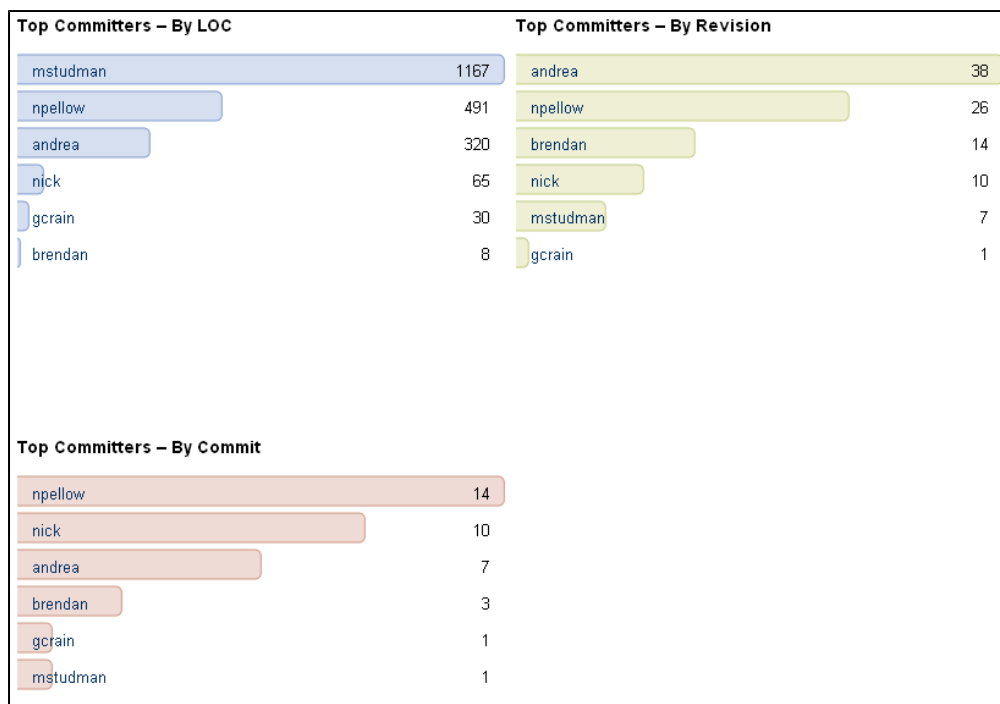
Code Metrics

A number of built-in reports are also provided:

Screenshot: Commit Time/Volume



Screenshot: Top Committers



Notes

Related Topics

[Browsing through a repository](#)

Using favourites in FishEye

FishEye allows you to add changesets, files, people and repositories as favourites. You can view your favourites, or see a stream of all activity relating to your favourites. We suggest that you select items that you are currently working on as favourites, to create a more relevant personalised view.

You can always view your favourites from the menu at the top of the screen, next to your username.

i If you are using Crucible, you can also add [code reviews](#) to your favourites.

On this page:

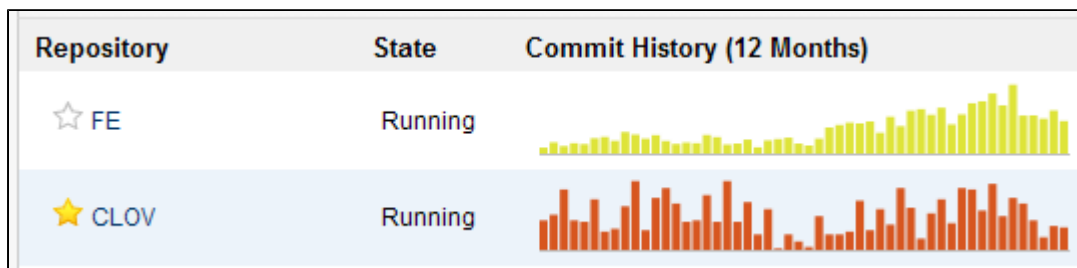
- [Adding favourites](#)
- [Managing favourites](#)

Adding favourites

To add an item to your favourites, follow one of the options below:

People	Hover the mouse cursor over their avatar or username. In the context menu, click Follow .
Changesets	Open the changeset and click the grey star icon next to its name, near the top of the screen.
Files or folders	Open the file or folder and click the grey star icon that appears next to its name. The name appears in the breadcrumb links at the top of the screen.
Repository	Click the Source tab and then the grey star icon that appears next to the name of the desired repository.

Screenshot: Adding a repository to your favourites



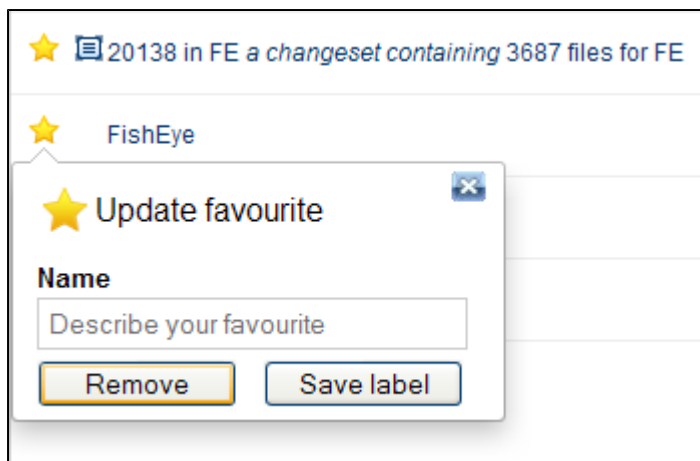
Managing favourites

To change the display name for, or remove, a favourite:

1. Click the Favourites menu (at the top of the FishEye screen, next to your username) and choose **Manage favourites**.
2. Click the yellow star beside the favourite, and either:
 - a. edit the display name, and click **Save label**
 - b. click **Remove**.

Due to [FE-2348](#) you cannot currently rename favourite directories, users or committers

Screenshot: Renaming a favourite



Changeset Discussions

Please see the [Crucible documentation](#) for instructions on this feature.

Viewing the commit graph for a repository

The commit graph shows changesets in their respective branches, using configurable "swimlanes". This allows you to see key information such as branching and merging (if you are using Git or Mercurial, you will be able to see anonymous branches as well). You can also use highlights to identify changesets in the same branch, commits with JIRA issues, and reviewed/unreviewed changesets. Clicking a changeset with the appropriate highlight active shows you related changesets, such as changesets with the same lineage, the same JIRA issue or same Crucible review.

Before you begin

- Subversion repositories currently do not show lines between branch swimlanes (i.e. merging). But in some cases, FishEye might pick up associations based on SVN branch points.
- Some features of the commit graph are only available if you are using [Crucible](#) with FishEye. For details, see the description below.

- Some features of the commit graph are only available if you are using [JIRA](#) with FishEye. For details, see the description below.

On this page:

- [Before you begin](#)
- [Browsing to the commit graph for a repository](#)
- [Highlighting information in the commit graph](#)
- [Viewing changesets across all branches](#)
- [Reordering swimlanes for Git repositories](#)

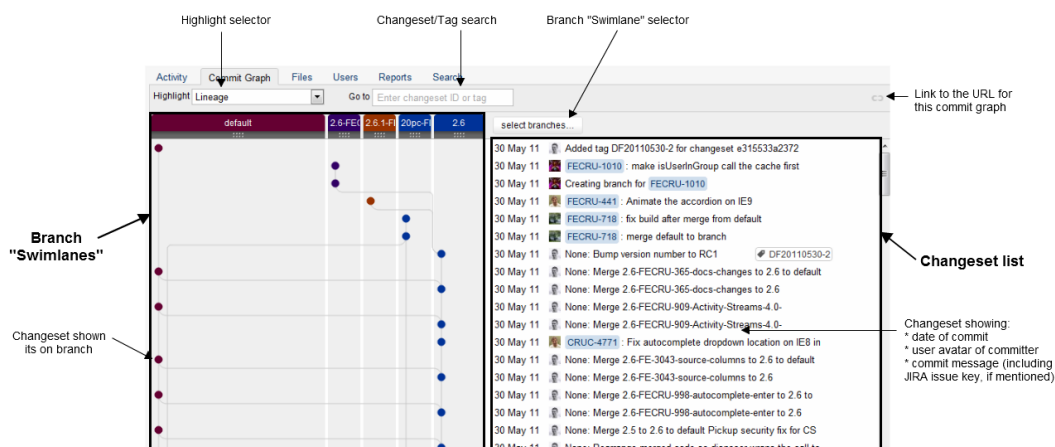
Related pages:

- [Subversion Changeset Parents and Branches](#)
- [What are Subversion root and tag branches?](#)
- [Perforce Changesets and Branches](#)
- [Using the FishEye Screens](#)
- [Browsing through a repository](#)
- [JIRA Integration in FishEye](#)

Browsing to the commit graph for a repository

To view the commit graph for a repository:

- Navigate to the desired repository, as described on [Browsing through a repository](#).
- Click the **Commit Graph** tab.
- The commit graph for the repository will be displayed.




Annotated screenshot above: Commit graph for a repository – default view

Highlighting information in the commit graph

The **'Highlight'** dropdown of the commit graph allows you to highlight different types of information in the swimlanes. When a highlight is active, you can also select a changeset in the changeset list to show related changesets. For example, if you have the 'JIRA Issues' highlight active, selecting a changeset with a JIRA issue (in the commit comment) will show which other changesets have the same JIRA issue.

Selecting a changeset (regardless of highlight) will display the following in the changeset list:

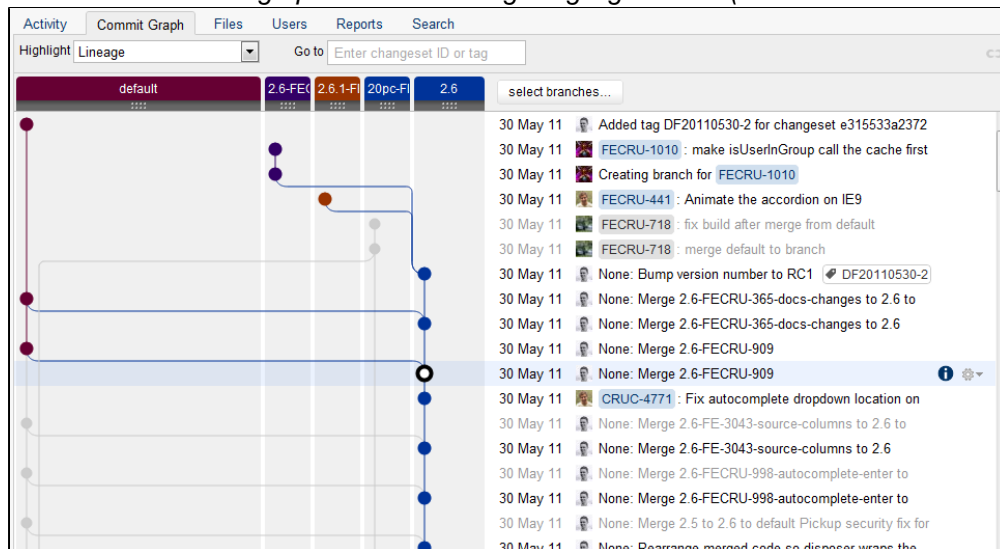
- Display an  icon next to the changeset. Clicking this changeset will display a changeset summary
- Display dropdown menu (accessible via a cog icon) allowing you to view the full changeset, view the changeset in the activity stream or create a review for the changeset.

In this section:

- [Highlighting the lineage of a changeset](#)
- [Highlighting JIRA issues](#)
- [Highlighting reviewed changesets](#)

Highlighting the lineage of a changeset

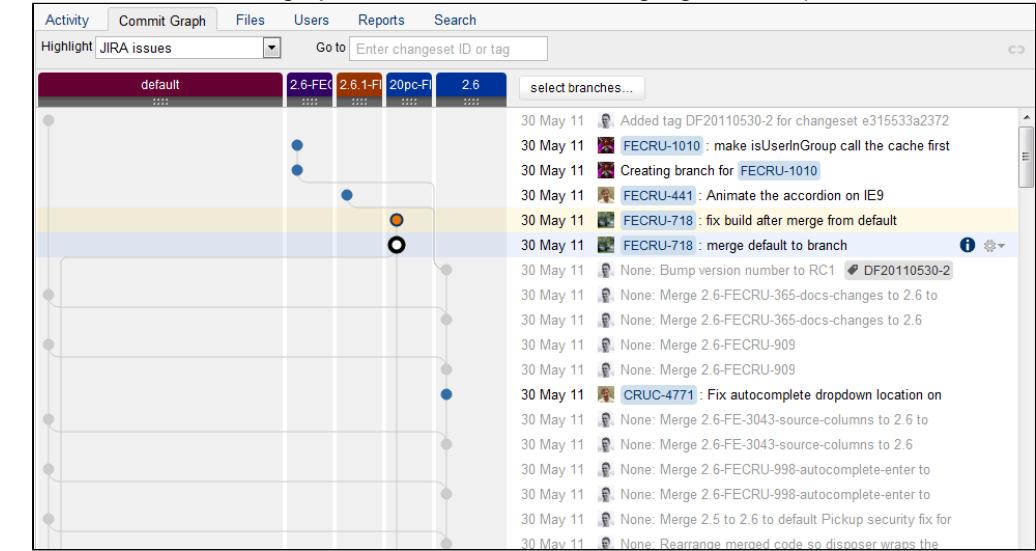
Action	Behaviour
Select the 'Lineage' highlight	Colours the changesets in the swimlanes according to which branch they are in.
Select a changeset in the changeset list with the 'Lineage' highlight active	Shows where a changeset comes from and where it propagates to, i.e. its ancestors and descendants.
Mouseover a changeset in the swimlanes with the 'Lineage' highlight active	Shows a pop-up displaying all branches that the changeset is referenced in. This will include branches that you may not have swimlanes displayed for.

Screenshot: Commit graph with the 'Lineage' highlight active (with mouseover on a changeset)**Highlighting JIRA issues**

This highlight is only available if you have [integrated FishEye with JIRA](#) and [linked your repository to a JIRA project](#).

Action	Behaviour
Select the 'JIRA issues' highlight	Colours the changesets in the swimlanes that have a JIRA issue key in the commit message.
Select a changeset in the changeset list with the 'JIRA issues' highlight active	Colours the changesets in the swimlanes that have the same JIRA issue key in the commit message, as the changeset selected.
Mouseover a changeset in the swimlanes with the 'JIRA issues' highlight active	Shows a pop-up displaying all branches that the changeset is referenced in and all referenced JIRA issues.

Screenshot: Commit graph with the 'JIRA Issues' highlight active (with mouseover on a JIRA issue key)

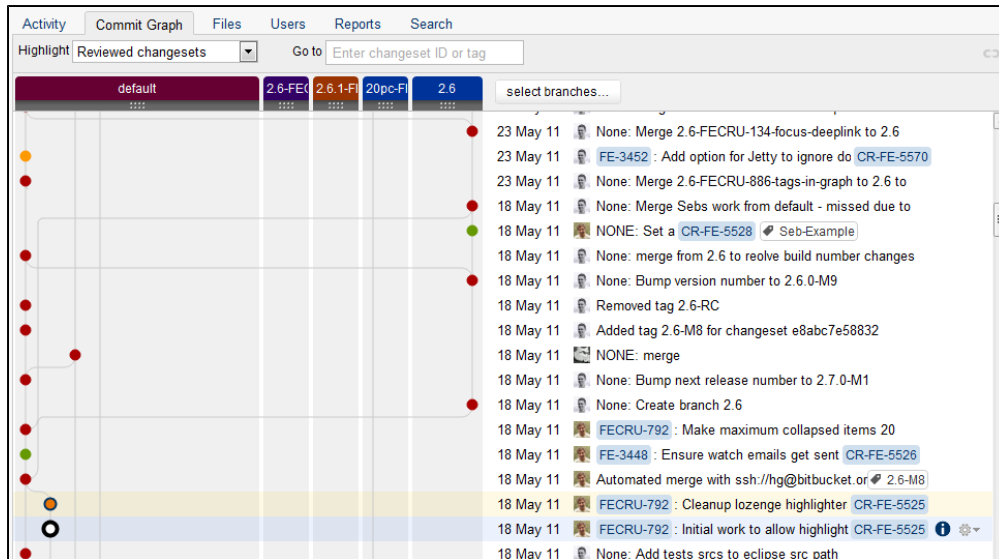


Highlighting reviewed changesets

This highlight is only available if you are using FishEye with Crucible.

Action	Behaviour
Select the 'Reviewed changesets' highlight	Colours the changesets in the swimlanes that have been reviewed (i.e. included in a Crucible review), as follows: <ul style="list-style-type: none">* Red dot: Unreviewed changeset, i.e. associated with review in 'Dead' or 'Rejected' state or no review associated.* Yellow dot: Changeset under review, i.e. associated with review not in 'Dead', 'Rejected' or 'Closed' state.* Green dot: Changeset reviewed, i.e. associated with review in 'Closed' state.
Select a changeset in the changeset list with the 'Reviewed changesets' highlight active	Colours the changesets in the swimlanes that are part of the same review as the changeset selected.
Mouseover a changeset in the swimlanes with the 'Reviewed changesets' highlight active	Shows a pop-up displaying all branches that the changeset is referenced in and the Crucible review key.

Screenshot: Commit graph with the 'Reviewed changesets' highlight active (with mouseover on a changeset)



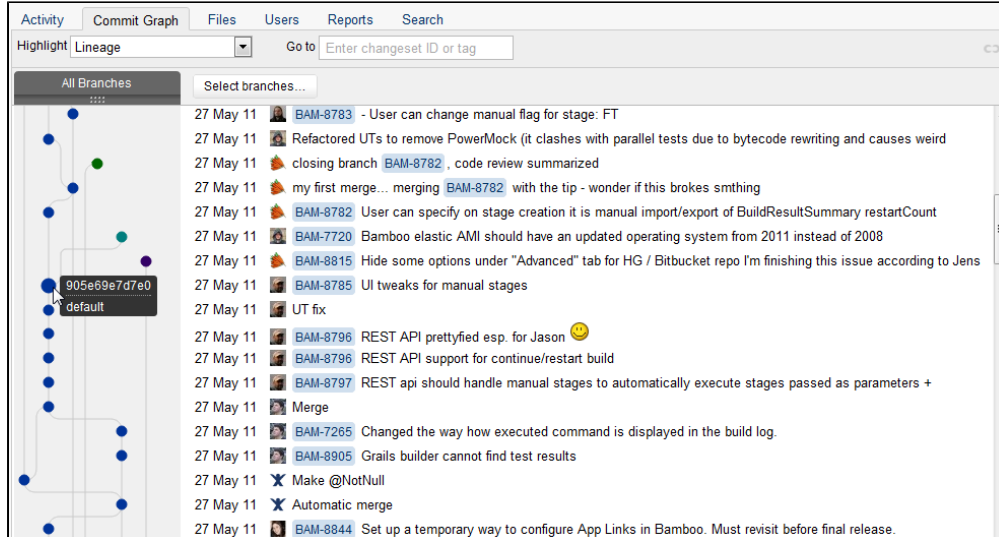
Viewing changesets across all branches

The 'All Branches' mode allows you to view commit activity across all branches of a repository. In this mode, the swimlane headers are not displayed. However, you can mouseover any changeset to display information about the changeset, as described in the 'Highlighting Information in the Commit Graph' section above.

To view the 'All Branches' mode of the commit graph for a repository:

1. Click the **'select branches...'** button when viewing the commit graph.
2. In the 'Select Branches' pop-up, click **'Switch to all branches mode'**.

Screenshot: Commit graph – 'All Branches' mode.



Reordering swimlanes for Git repositories

Reordering swimlanes is useful if you just want to show branches in a certain order. However, ordering swimlanes is vital for Git repositories, as it is the only way of determining which branch a commit is from.

When you view the commit graph for a Git repository, FishEye works from the leftmost swimlane to the right doing the following:

- For each swimlane, FishEye checks if the commit is in that branch. If the commit is in the branch, a dot is shown representing the commit.
- If the commit is not in the branch, the dot for the commit is moved to the next column on the right.

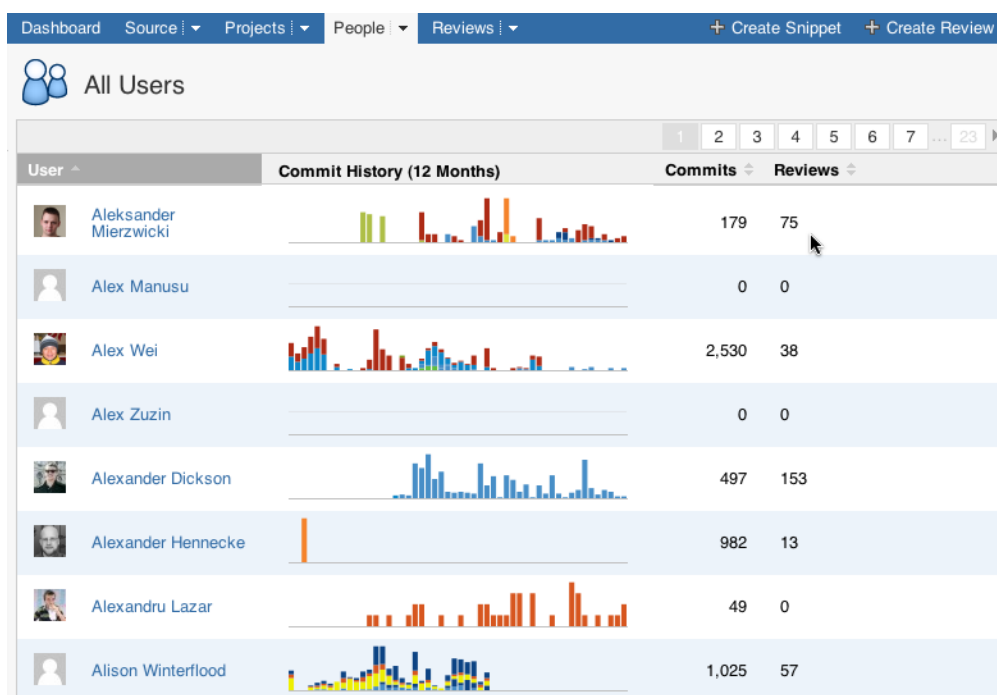
For example, if the 'master' swimlane is to the left of another swimlane, e.g. 'fisheye-2.6' branch, there will be no changesets shown in the 'fisheye-2.6' swimlane, as all the commits will be picked up in the 'master' swimlane. However, if you move the 'fisheye-2.6' swimlane to the left of the 'master' swimlane, it will pick up all of the FishEye 2.6 commits.

For more information, read this Knowledge Base article: [Ordering of Branches Important When Visualising Git Changeset](#)

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 screen.

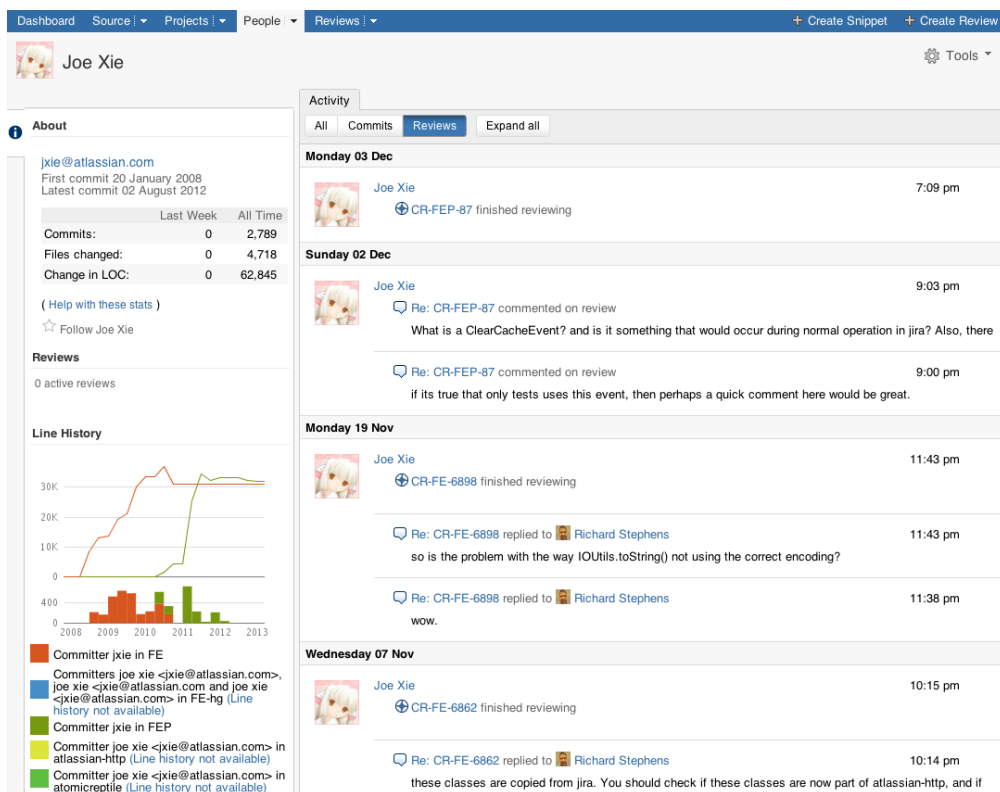
Screenshot: List of all committers in FishEye



The All Users screen shows all those with accounts on the system. You can see their commit history (expressed as a bar graph) and their total number of commits.

Click on a person's name to see detailed information about their additions to the repository, and issue updates. If you are using FishEye with Crucible and have JIRA integration set up, you can see their review activity.

Screenshot: Statistics on a Person in FishEye



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

Avatars

By default, each user has a unique avatar that is randomly formed from the text in their email address. You can add your own avatar by uploading an image to an external service such as Gravatar, which Crucible supports. See [Changing your User Profile](#).

i If you are using [Crucible](#), statistics for each person's code reviews are also available.

Using smart commits

Smart commits allow repository committers to perform actions like transitioning JIRA issues or creating Crucible code reviews by embedding specific commands into their commit messages. Multiple smart commits can be used in one commit message, however they must be on separate lines. Note that smart commits don't provide for field-level updates in JIRA issues.

Note that Smart commits require the following:

- An application link must be configured between FishEye/Crucible and JIRA. See [Adding an application link](#).
- If you have **JIRA 5.0 or later**, and the **JIRA FishEye Plugin** (at least version **5.0.10**), a project/entity link is unnecessary. Otherwise, a project link must be configured between FishEye/Crucible and JIRA. See [Adding project links between applications](#).
- Smart commits must be enabled in FishEye. See [Enabling smart commits](#).

Transition your JIRA issues

i Compatibility

- In order to use smart commits with JIRA you need to have the [JIRA FishEye Plugin version 3.4.5](#) or above installed on your JIRA instance.
- Note that smart commits only support the default JIRA issue key format (that is, two or more uppercase letters, followed by a hyphen and the issue number, for example BAM-123).

On this page:

- [Transition your JIRA issues](#)
 - [Basic command line syntax](#)
 - [Advanced command line syntax](#)
 - [Commands](#)
- [Integration with Crucible](#)
 - [Creating a review](#)
 - [Adding reviewers](#)
 - [Updating an existing review](#)
- [Linkers](#)
- [Error handling](#)

Related pages:

- [Enabling smart commits](#)
- [Configuring web hooks](#)
- [Transitioning JIRA issues](#)
- [Write your own smart commit](#)
- [Linkers](#)

Basic command line syntax

The basic command line syntax for your commit comment is:

<ISSUE_KEY> #<COMMAND> <optional COMMAND_PARAMETERS>

Please note, commit commands cannot span more than one line (i.e. you cannot use carriage returns).

For example, if you include the following text in your commit message, FishEye will record 2 days and 5 hours of work against issue JRA-123, when you perform your commit:

```
JRA-123 #time 2d 5h
```

i Please see the section below for further information on the command line parameters.

Advanced command line syntax

If you wish to perform multiple actions on issues, you can create composite commands by combining keywords, as described below. Please note, commit commands cannot span more than one line (i.e. you cannot use carriage returns).

- **To perform multiple actions on a single issue:**

<ISSUE_KEY> #<COMMAND1> <optional COMMAND1_PARAMETERS> #<COMMAND2> <optional COMMAND2_PARAMETERS> #<COMMAND3> <optional COMMAND3_PARAMETERS> etc

For example, if you include the following text in your commit message, FishEye will log 2 days and 5 hours of work against issue JRA-123, add the comment 'Task completed ahead of schedule' and resolve the issue, when you perform your commit:

```
JRA-123 #time 2d 5h #comment Task completed ahead of schedule #resolve
```

- **To perform a single action on multiple issues:**

<ISSUE_KEY1> <ISSUE_KEY2> <ISSUE_KEY3> #<COMMAND> <optional COMMAND_PARAMETERS> etc

For example, if you include the following text in your commit message, FishEye will resolve issues JRA-123, JRA-234 and JRA-345, when you perform your commit:

```
JRA-123 JRA-234 JRA-345 #resolve
```

- **To perform multiple actions on multiple issues:**

<ISSUE_KEY1> <ISSUE_KEY2> <ISSUE_KEY3> #<COMMAND1> <optional COMMAND1_PARAMETERS> #<COMMAND2> <optional COMMAND2_PARAMETERS> #<COMMAND3> <optional COMMAND3_PARAMETERS> etc.

For example, if you include the following text in your commit message, FishEye will log 2 days and 5 hours of work against issues JRA-123, JRA-234 and JRA-345, add the comment 'Task completed ahead of schedule' to all three issues, and resolve all three issues, when you perform your commit:

```
JRA-123 JRA-234 JRA-345 #resolve #time 2d 5h #comment Task completed ahead of schedule
```

Commands

Note that you can see the custom commands available for use with smart commits by visiting the JIRA issue and seeing its available workflow transitions (in an issue, click **View Workflow**, near the issue status).



Online banking / EBANK-7

A blocker

[Edit](#)
[Assign](#)
[Assign To Me](#)
[Comment](#)
[More Actions](#)
[Resolve Issue](#)
[Close Issue](#)
[Workflow](#)

Details

Type: Bug
 Priority: Blocker
 Affects Version/s: None
 Labels: None

Status: Open
 (View Workflow)
 Resolution: Unresolved
 Fix Version/s: None

Command	Command Parameters	Description	Example
---------	--------------------	-------------	---------

#time	<n>w <n>d <n>h <n>m <work log comment> <i>where <n> is a user-specified time period.</i>	<p>This command records time tracking information against an issue. Please note, time tracking must be enabled for your JIRA instance to use this command. Please check with your JIRA administrator, if you cannot record time tracking information against issues.</p> <p><i>Please note:</i></p> <ul style="list-style-type: none"> • <i>Work log comments cannot be set using smart commits. See FE-3757.</i> 	#time 1w 2d 4h 30m Total work logged — this command would record 1 week, 2 days, 4hours and 30 minutes against an issue, and add the comment 'Total work logged' in the Work Log tab of the issue.
#comment	<comment text>	This command records a comment against an issue.	#comment My comment. — this command would create the comment, "My comment", against the issue.
#<workflow command> e.g. #resolve	<workflow> <comment text>	This command transitions an issue to a particular workflow state. Please see the documentation for Configuring Workflow in JIRA.	<p>#close Fixed the issue — this command would execute the 'Close Issue' workflow transition for an issue in the default JIRA workflow and adding the comment 'Fixed the issue'.</p> <p>#start — this command would execute the 'Start Progress' workflow transition for an issue in the default JIRA workflow</p>

FishEye will do prefix matching for issue transitioning. For example, if you have transition name with spaces, such as `finish work` then specifying `#finish` is sufficient. Hyphens replace spaces: `#finish-work`

FishEye will only execute issue transitions if there is no ambiguity in valid workflow transitions. Take the following example:
An issue has two valid transitions:

- `Start Progress`
- `Start Review`

A smart commit with action `#start` is ambiguous as FishEye will not be able to determine which transition to execute. In order to execute one of these transitions, the smart commit specified will need to be fully qualified `#start-review`

Please note: If you want to resolve an issue using the `#resolve` command, you will not be able to set the resolution via smart commits.

Integration with Crucible

Please note that each commit command in the commit message must not span more than one line (i.e. you cannot use carriage returns). You can use multiple commands in the same message as long as they are on separate lines.

Creating a review

With smart commits, it is also easy to create a Crucible review from a commit:

```
Fix a bug +review CR-TEST
```

The command "+review" tells FishEye to create a new review in the project CR-TEST with the content of the changeset. The review will be in a draft state unless the project has default reviewers or reviewers are explicitly mentioned. If you only have one project in Crucible, or a repository is a project's default repository, it is not necessary to mention the project key. Just use "fix a bug +review".

Adding reviewers

Reviewers can be added to a new review using a smart commit:

```
Fix a bug +review CR-TEST @jcage @skhan
```

That command will create a new review in *PROJ* and add the users *jcage* and *skhan* to the review. The review will be automatically started if reviewers are specified.

Note, you cannot add reviewers to existing reviews using smart commits.

Updating an existing review

Often, reviews require rework and changes in response to comments left by the team. When committing these changes, adding the review key will iteratively add these new changes to the review:

```
Implement rework on past work +review CR-TEST-123
```

With this command FishEye will add the changeset content to the review *CR-TEST-123*.

Linkers

When using smart commits you can use linkers that create a hyperlink to the JIRA issue. See [Linkers](#) for more information.

Error handling

If there are any errors during the processing of smart commits, they will be logged to FishEye's error console, as well as emailed to the actioning users. Please speak to your FishEye administrator about [Configuring SMTP](#).

Changing your User Profile

You can change FishEye (and Crucible) settings such as password, notifications, profile image and display settings.

To change your FishEye settings:

1. Log into FishEye.
2. Choose **Settings** from the User Menu (labelled with your username) at the top of the screen.
3. Update your user settings as required. Each tab is described in more detail below.

4. Click **Close**.**On this page:**

- [Display Settings](#)
- [Profile and Email](#)
- [Change Password](#)
- [Open Authentication \(OAuth\)](#)
- [Author Mapping](#)
- [Watches](#)
- [Reviews](#)

Display Settings

Display Settings	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	Changesets per page	The default is 30 per page.
	Always expand changesets in stream	Default is Yes .
Diff view	Diff mode	Default is Unified .
	Line wrapping	Default is None i.e. long lines will never word-wrap. Soft is when long lines will word-wrap.
	Context lines	Default is 3. The number of lines to show (for context), if the diff contains more than three lines of code.
Source view	Default annotation mode	Default is Age .
	Highlighting colours	The default scheme uses bright colours for highlighting diffs in the code. If you prefer more muted colours, select Classic (muted) .


	Tab width	Default is 8. Can be changed to a number between 1 and 10.
--	------------------	--

Profile and Email

Email settings	Display Name	Name displayed for the user currently logged in.
	Email Address	The address to which all email notifications will be sent.
	Email Format	Default is Text . Can be sent as HTML .
Email watches	Send Watch Emails	The frequency at which emails will be sent for watch notifications. Immediately is the default value. Daily sends a summary of changes.
Profile Picture	Choose picture	Upload an avatar image of your choice. This image will be displayed next to your username throughout FishEye/Crucible. Accepted formats are JPG, GIF and PNG. Image file size limit is 2Mb. Images will be automatically cropped on upload.

Change Password

Change your password from this tab, if required. Please note that passwords are case-sensitive.

 This tab is not displayed if your FishEye instance is connected to an external LDAP authentication source, such as LDAP. You will need to contact your administrator for assistance.

Open Authentication (OAuth)

Configure your OAuth settings on this page. You can choose to allow gadgets/applications to access FishEye data using your account.

Read more about [OAuth](#).

Author Mapping

The **Author Mapping** tab allows you to make an association between you (as a logged-in user) and a committer, for each repository.

This is only necessary if the name or email of the user within FishEye is different from the committer name or email within the repository. By default, FishEye will automatically match users to committers where it can.

Watches

By adding a 'watch', you can ask to receive emails about changes made to the repository. Any watches that you have set up in FishEye/Crucible will be displayed on this tab. You can watch the [dashboard activity stream](#), [changes](#) and [repositories](#). Watching an activity stream/repository allows you to receive emails when updates occur. Note, the option to add a watch may only be available if the administrator has [enabled watches](#) for the repository.

You can delete any of your watches by clicking **Delete** next to the watch.

Reviews

This functionality is used by [Crucible](#).

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

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

Auto-mark files as 'read'	Default is Yes .	
Review Notifications Events	State change	Default is Immediate . A Crucible review moves through different states e.g: 'Draft', 'Under Review'. An email is sent when the state changes.
	Comment added	Default is Immediate . An email is sent when a comment is added to a review.
	Comment reply added	Default is Immediate . An email is sent (to the Moderator only) when any reviewer has completed their review .
	Participant finished	Default is Immediate . An email is sent when a reviewer is added or removed from a review, after it has gone into the 'Under Review' state .
	General message	Default is Immediate . An email is sent when a reviewer is added or removed from a review, after it has gone into the 'Under Review' state .
	File revision added	Default is Immediate .
	Uncomplete review if defect is raised:	Default is Yes . This allows reviews to be resurrected automatically to deal with new code or defects.
	Uncomplete review if revision is added:	Default is Yes . This allows reviews to be resurrected automatically to deal with new code or defects.

	My actions	Default is No . If set to Yes , an email is sent every time you perform an action on a review.
--	-------------------	--

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

Screenshot: User Profile Settings

Settings

Display Settings

File history view mode: ☒ Physical ☐ Logical

Timezone:

Changelog

Changesets per page:

Always expand changesets in streams: ☒ Yes ☐ No

Diff view

Diff mode: ☒ Unified ☐ Side-by-side

Line wrapping: ☒ None ☐ Soft

Context lines:

Source view

Default annotation mode: ☒ Age ☐ Author ☐ None

Highlighting colours: ☒ Default ☐ Classic (muted)

Tab width:

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.

i 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

Login Required

Username:

Password:

[Forgot your password?](#)

Screenshot: The Request New Password screen

Request New Password

To have a new password generated and sent to you, please enter either your username or email address below.

Username:

OR

Email:

Please enter the word as shown below:

| o o i n s

Pattern matching 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 wild cards:

?	Matches one character (any character except path separators)
*	Matches zero or more characters (not including path separators)
**	Matches zero or more <i>path segments</i> .

Remember that Ant globs 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 wild cards.

Also see the [Ant documentation](#).

Examples

<code>*.txt</code>	Matches <code>/foo.txt</code> and <code>/bar/foo.txt</code> but not <code>/foo.txtty</code> or <code>/bar/foo.txtty/</code>
<code>/*.txt</code>	Matches <code>/foo.txt</code> but not <code>/bar/foo.txt</code>
<code>dir1/file.txt</code>	Matches <code>/dir1/file.txt</code> , <code>/dir3/dir1/file.txt</code> and <code>/dir3/dir2/dir1/file.txt</code>
<code>**/dir1/file.txt</code>	Same as above.
<code>/**/dir1/file.txt</code>	Same as above.
<code>/dir3/**/dir1/file.txt</code>	Matches <code>/dir3/dir1/file.txt</code> and <code>/dir3/dir2/dir1/file.txt</code> but not <code>/dir3/file.txt</code> , <code>/dir1/file.txt</code>
<code>/dir1/**</code>	Matches all files under <code>/dir1/</code>

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:

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

`DATECONSTANT` can be any of:

<code>now</code>	This very instant (at the time the expression was evaluated).
<code>today</code> <code>todaygmt</code>	The instant at 00:00:00 today. (server-time* or GMT)
<code>thisweek</code> <code>thisweekgmt</code>	The instant at 00:00:00 on the first day of this week. Sunday is considered the first day. (server-time* or GMT)

thismonth thismonthgmt	The instant at 00:00:00 on the first day of this month. (server-time* or GMT)
thisyear thisyeargmt	The instant at 00:00:00 on the first day of this year. (server-time* or GMT)

* 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

2005-01-02	The start of the day on January 1, 2005 (server's timezone)
2005-01-02-0500	The start of the day on January 1, 2005 at GMT offset -0500 (New York)
2005-01-02T12:00:00Z	Midday, January 1, 2005 GMT
today-P1D	Yesterday (start of day)
today+P1D	Start of tomorrow
thismonth-P1M	Start of last month
thisyear+P1Y	Start of next year
now-PT1H	One hour ago
now+PT1H2M3S	One hour, two minutes and three seconds from now

EyeQL Reference Guide

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 when programming the FishEye API.

query:

```

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|csid|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 [or] is used.

exclusive if (or) 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** | **dir** | **directory** | **revision** | **author** | **date** | **comment** | **csid** | **isBinary** | **totalLines** | **linesAdded** | **linesRemoved** | **isAdded** | **isDeleted** | **isCopied** | **isMoved** | **tags** | **reviews** | **aggregate**)
(**as** *word*)?

The attribute to return, optionally followed by a name to use for the column.

Notes: **reviews** applies to [Crucible reviews](#) .

aggregate-return-field:

(**count**(**revisions**) | **count**(*binary-field*) | **count**(**distinct** *other-field*) | **sum**(*numeric-field*) | **average**(*numeric-field*) | **max**(*numeric-field*) | **min**(*numeric-field*))

The aggregate field to return.

Notes:

binary-fields are **isBinary**, **isAdded**, **isDeleted**, **isCopied**, **isMoved**. e.g. **count(isAdded)** will return the number of added files.

numeric-fields are **totalLines**, **linesAdded**, **linesRemoved**.

other-field can be **path**, **dir**, **author**, **date**, **csid**, **tags** or **reviews**. e.g. **count(distinct path)** will return the number of unique paths. **count(distinct tags)** will return the number of unique tags.

If a **group by** is given, give sub-totals for each group.

With no **group by** clause, you can have:

- return *normal columns*
- return *aggregates*

With a **group by** **changeset** | **csid** clause:

- return *normal columns*
- return *csid, comment, date, author, aggregates*

With a group by file|path clause:

- return *normal columns*
- return path, *aggregates*

With a group by dir|directory clause:

- return *normal columns*
- return dir, *aggregates*

i.e. The EyeQL can contain a **returns** clause that contains all non-aggregate columns, or all aggregate column. Non-aggregate and aggregate columns can only be mixed if the columns are unique for the grouping.

limit-clause:

(*length* | *offset, length* | *length offset offset*)

Notes: Limits the number of results to return. *offset* specifies the starting point of the truncated result set and *length* 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]
```

Find revisions that do not have any associated review:

```
select revisions where (not in any review)
```

Return number of matched revisions, the number of files modified, authors who modified code, changesets, tags, and reviews:

```
select revisions
where date in [ 2003-10-10, 2004-12-12 ]
return count(revisions), count(distinct path), count(distinct author),
count(distinct csid), count(distinct tags), count(distinct reviews)
```

As Sub-totals for each distinct changeset, Return csid, the author, date, comment, number of matched revisions, the number of files modified, the lines added/removed:

```
select revisions
where date in [ 2003-10-10, 2004-12-12 ]
group by changeset
return csid, author, date, comment, count(revisions), count(distinct path),
sum(linesAdded), sum(linesRemoved)
```

For each matched file, return the file name, number of matched revisions, the lines added/removed:

```
select revisions
where date in [ 2003-10-10, 2004-12-12 ]
group by file
return path, count(revisions), sum(linesAdded), sum(linesRemoved)
```

Show all the changesets with no review:

```
select revisions
from dir /
where not reviewed
group by changeset
return csid, author, count(revisions), comment
```