User documentation for FishEye 3.4

Contents

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.

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.

FishEye

Supported platforms

This page lists the supported platforms for **FishEye 3.4.x** and its minor releases.

Key: ✓ = Supported ✗ = Not Supported

Java		
Oracle JRE / JDK	✓ 1.6 ✓ 1.7	FishEye requires the **Java Runtime** (JDK or JRE), version as noted. Pre-release/Early access versions of the Java Runtime are *not supported*. You can download an Oracle Java Runtime for Windows/Linux/Solaris. On Mac OS X, the JDK is bundled with the operating system. For the OpenJDK, download and install instructions for Linux flavours are at http://openjdk.java.net/install/. We highly recommend that you use the Oracle JVM or OpenJDK (for Linux only), or use the default Mac OS X JVM. Other implementations have not been tested. Please note: • Once you have installed Java, 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 (`--Xmx`) to a reasonable value, considering the RAM requirements of your system.

OpenJDK	✅ 1.7 (Linux only)	• 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 `-server` parameter. • You'll need the JDK for the JSP source download. 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 .
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. • ⚠️ Although FishEye can be run in virtualised environments, Atlassian is not yet able to provide technical support for performance-related problems in a virtualised environment. If you do choose to run FishEye in a VM, please ensure that you choose a VM with good IO throughput.
Linux	✅	
Apple Mac OS X	✅	
Databases		
MySQL	✅ MySQL Enterprise Server 5.1+ ✅ MySQL Community Server 5.1+ ❌ MySQL 5.0	The FishEye built-in database, running HSQLDB, is somewhat susceptible to data loss during system crashes. External databases (such as MySQL) are generally more resistant to data loss during a system crash. See the FishEye Database documentation for further details. ⚠️ For MySQL: • For 5.1, the version must be 5.1.10 or later • For 5.6, the version must be 5.6.11 or later • FishEye 3.3+ no longer supports MySQL 5.0. See End of Support Announcements for FishEye. ⚠️ For PostgreSQL: FishEye 3.3+ no longer supports PostgreSQL 8.2. See End of Support Announcements for FishEye. ⚠️ For SQL Server: FishEye 3.3+ no longer supports SQL Server 2005. See End of Support Announcements for FishEye.
PostgreSQL	✅ 8.3, 8.4 ✅ 9.0, 9.1, 9.2, 9.3 ❌ 8.2	
Oracle	✅ 11g	
SQL Server	✅ 2008, 2008 R2, 2012 ❌ 2005	
HSQLDB	✅ Bundled; for evaluation use only	
Web browsers		

Microsoft Internet Explorer	9.0, 10.0, 11.0 8.0	FishEye 3.3+ no longer supports Internet Explorer 8. See End of Support Announcements for FishEye.
Mozilla Firefox	Latest stable version supported 3.6, 4.0	
Safari	Latest stable version supported 4, 5	
Chrome	Latest stable version supported	
Version control systems		
Subversion	Server: 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8 Client: SVNKit (bundled & the default) Native JavaHL 1.8 Native JavaHL 1.7 Native JavaHL 1.6	FishEye 3.1, and later, **do not** support the native JavaHL 1.6 client. See Native support for SVN for discussion.
CVS (and CVSNT)	All versions	
Perforce	Client version 2007.3 or later	
Git	1.7.1.1 or later	Externally-hosted repositories only are supported.
Mercurial	1.5.1, 1.5.4, 1.6.4, 1.7.5, 1.8.4, 1.9.3 2.0.2, 2.1.2, 2.2.3, 2.3.2, 2.4, 2.5.2, 2.6.3, 2.7.2, 2.8.2, 2.9.1	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.
Atlassian applications		
Crowd	Crowd server 2.4.x Crowd client library: 2.4.1	From version 2.8.x, FishEye bundles the Crowd 2.4.1 client library, and supports the Crowd 2.4.x server.
JIRA	JIRA 5.0+	FishEye to JIRA communication requires JIRA 5.0.x or later. Communication the other way, from JIRA to FishEye, depends on the JIRA FishEye Plugin. Note that the JIRA FishEye Plugin is bundled with JIRA. If you are using a version of JIRA earlier than JIRA 5.0 you can upgrade the plugin in JIRA to get support for FishEye.

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).
Monitor	Minimum screen resolution of 1024x768. Recommended screen resolution of 1280x768 or above.

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 bundles 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, Internet Explorer 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 upcoming versions of FishEye

Platform	Announcement Date	FishEye End of Support
MySQL 5.0	27 November 2013	As of FishEye 3.3
PostgreSQL 8.2	27 November 2013	As of FishEye 3.3
SQL Server 2005	27 November 2013	As of FishEye 3.3
Internet Explorer 8	27 November 2013	As of FishEye 3.3

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 new versions of databases, web browsers etc. are released, the cost of supporting multiple platforms grows exponentially, making it harder to provide the level of support our customers have come to expect from us. Therefore, we no longer support platform versions marked as end-of-life by the vendor, or very old versions that are no longer widely used.

On this page (most recent announcements first):

Deprecated FishEye support for MySQL 5.0 (announced 27 November 2013)

Atlassian announces the deprecation of FishEye support for MySQL 5.0. We will no longer support MySQL 5.0 in FishEye 3.3. FishEye 3.3 is expected to be released in the first half of 2014.

If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Deprecated FishEye support for PostgreSQL 8.2 (announced 27 November 2013)

Atlassian announces the deprecation of FishEye support for PostgreSQL 8.2. We will no longer support PostgreSQL 8.2 in FishEye 3.3. FishEye 3.3 is expected to be released in the first half of 2014.

If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Deprecated FishEye support for SQL Server 2005 (announced 27 November 2013)

Atlassian announces the deprecation of FishEye support for SQL Server 2005. We will no longer support SQL Server 2005 in FishEye 3.3. FishEye 3.3 is expected to be released in the first half of 2014.

If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Deprecated FishEye support for Internet Explorer 8 (announced 27 November 2013)

Atlassian announces the deprecation of FishEye support for IE8. We will no longer support IE8 in FishEye 3.3. FishEye 3.3 is expected to be released in the first half of 2014.

If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Deprecated FishEye 3.2 support for older versions of JIRA (announced 27 August 2013)

Atlassian announces the deprecation of support for FishEye communication with older versions of Atlassian JIRA. We will stop supporting older versions of JIRA as follows:

- From FishEye 3.2, support for FishEye to JIRA communication for versions of JIRA earlier than 5.0, will end. Please note that communication from JIRA to FishEye will continue to work as it currently does. FishEye 3.2 is expected to be released late in 2013.

If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Deprecated support for internally managed repositories in FishEye (announced 16 August 2012)

Atlassian announces the deprecation of support for internally managed repositories in FishEye. We will stop supporting internally managed repositories as follows:

- From 14 August 2013, Atlassian support for internally managed repositories in FishEye will end. Please read the full announcement for this change.

If you have questions or concerns regarding this announcement, please email `eol-announcement at atlassian dot com`.

Deprecated database support for FishEye (announced 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 (announced 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 at atlassian dot 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 (announced 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 at atlassian dot com`.

End of Life Announcement for Java Platform Support

Java Platform	Support End Date
Java Platform 5 (JDK/JRE 1.5)	When 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 (announced 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 features.
- 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 **ha s 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?

See our Migration guide for Git internally managed repositories for migration strategies and procedures.

Native support for SVN

This page describes an advanced feature of FishEye's Subversion support. It explores the technical background, and some of the issues you may encounter, if you wish to use the native JavaHL access feature.

For most users we recommend that you use the default SVNKit Subversion access client that is bundled with FishEye. You are only likely to need the native JavaHL access described on this page for certain edge case repositories.

FishEye Subversion access

FishEye interacts with Subversion repositories though a layer, defined by the Apache Subversion project, known as JavaHL. This is the high-level Java language binding for Subversion. There are two implementations of the JavaHL interface available:

Bundled SVNKit

The SVNKit implementation is a largely Java-based implementation provided by the SVNKit project. It is bundled with FishEye and is the default JavaHL implementation used. As a Java implementation it operates on all of FishEye's supported platforms.

Native JavaHL

The JNI-based implementation, coupled with a shared dynamic library, is referred to here as native JavaHL. As a native library, native JavaHL is platform-dependent. The shared library is C-based and must be compatible with the remaining Subversion client components installed on the platform. It varies across each platform and distribution.

On this page:

Native vs. bundled

Given that FishEye bundles the SVNKit implementation, why might you want to use the native implementation of JavaHL? In general our recommendation is to stick with the bundled SVNKit implementation. It is the simplest to use and works in the widest variety of scenarios. Nevertheless, there are some scenarios where it may be desirable to use the native implementation, if it is available.

The two implementations have quite different characteristics – these can affect the decision about which to use.

Here is a high-level list of some of the considerations we have encountered over the years:

Aspect	SVNKit - bundled with FishEye	Native JavaHL - platform dependent
Memory Usage	SVNKit uses the Java Heap. It therefore shares the heap that is being used for all FishEye's operations. It does benefit, however, from Java's garbage collection mechanism and we have not seen any memory leaks.	The native JavaHL implementation uses the native process heap and not the Java heap. It can increase the overall process memory usage but does not interfere with the Java heap usage. In some rare instances, we have seen memory leaks in the C-based JavaHL code. As FishEye is a long running service, these can cause problems over the life of the FishEye process.
Speed	In general, when using any of the Subversion network protocols, the JavaHL implementation speed is not a significant factor in the overall speed of Subversion operations as they are dominated by the network latency. Even for file:// access SVNKit is rarely the bottleneck.	If you are using file:// access to talk to a Subversion repository on the same server, then native JavaHL will most likely give the highest performance.
Compatibility	SVNKit has proven to be highly compatible with Subversion across all releases. The project is very responsive to bug reports when any differences become apparent. As an alternative implementation of JavaHL there will be differences between the SVNKit and the native Subversion JavaHL. This may affect some edge case repositories.	JavaHL uses predominantly the same code as Subversion itself so it is virtually 100% compatible.
Availability	SVNKit works on all of FishEye's supported platforms	It can be difficult to get an install of the JavaHL jar and shared library that is compatible with the version of Subversion installed on your platform.

Native JavaHL support

The native JavaHL interface and implementation naturally change with every release of Subversion. Normally these changes are incremental and backward compatible.

The compatibility matrix for recent FishEye versions is:

	SVNKit	Native JavaHL 1.6 client	Native JavaHL 1.7 client	Native JavaHL 1.8 client
FishEye 3.3				
1.8 Subversion Server	✓	✗	✓	✓

1.7 Subversion Server	✓	✗	✓	✓
1.6 Subversion Server	✓	✗	✓	✓
FishEye 3.1 – 3.2				
1.8 Subversion Server (not file:// access)	✓	✗	✓	Unsupported
1.8 Subversion Server (file:// access)	✗	✗	✗	Unsupported
1.7 Subversion Server	✓	✗	✓	Unsupported
1.6 Subversion Server	✓	✗	✓	Unsupported
FishEye 3.0 and earlier				
1.7 Subversion Server (not file:// access)	✓	✓	✗	✗
1.7 Subversion Server (file:// access)	✓	✗	✗	✗
1.6 Subversion Server any access	✓	✓	✗	✗

Note that:

- FishEye 3.1, and later, is not compatible with the JavaHL 1.6 client (or older versions) – it uses the org.apache.subversion package which is not provided in 1.6 JavaHL builds
- From FishEye 3.1, you must use a 1.7 or later JavaHL library if you want native JavaHL access.

Click here to read about compatibility changes from SVN 1.6 to 1.7...

The change from Subversion 1.6 to Subversion 1.7 was much more significant and for a number of FishEye's usages of the interface, it broke compatibility.

In Subversion 1.7, the JavaHL interfaces were updated and moved from the org.tigris.subversion package to the org.apache.subversion package. This coincided with the move of the Subversion project to the Apache Software Foundation. In addition to moving the package, the interface was modernized in a number of ways:

- Extended use of callbacks.
- Use of Java collections rather than native arrays.
- Properties were clarified as byte arrays rather than Java Strings.
- Use of typed Enums rather than primitive integer and char fields.

The existing org.tigris package was retained in most 1.7 distributions and was implemented as an adapter layer over the new org.apache package classes. Unfortunately a number of incompatibilities in the adapter layer meant that FishEye could not use the 1.7 native implementation:

1. Property returning methods were wrapped in a String constructor to convert from the byte[] type of the new interface to the String type used in the old interface. This meant that any null returns would throw NullPointerExceptions rather than returning null Strings.
2. Some of the callbacks were changed from plain interfaces to being interface extensions of the corresponding callback in the new package. This changed the type definition of the callback from an untyped Map to a typed Map. This caused ClassCastExceptions because the code is expecting a map containing byte[] but the underlying code was passing in a map containing strings.

For this reason, FishEye did not support native access using the 1.7 native library prior to FishEye 3.1.

Installing JavaHL for your platform

Atlassian FishEye bundles the SVNKit library to make connecting to your Subversion repository a painless process out of the box. If you do wish to use native JavaHL, it is your responsibility to install it onto your platform. Different organizations have different operating procedures and policies regarding how and what packages they are able to install on production servers.

In some cases the distribution you use will not provide a compatible JavaHL from an official package. In this case you will either need to build everything from source yourself (hard) or use a package from a Subversion vendor. We have used packages from two vendors over time, CollabNet and Wandisco. More recently, we have found it easier to use the Wandisco packages for JavaHL support.

The following sections detail our experiences when we investigated deploying JavaHL 1.7 on a variety of platforms. This is not a definitive list or guide. It is to give you an idea of the some of the issues you are likely to encounter getting a compatible JavaHL install working on a range of platforms and distributions.

Windows 7

Windows does not include a Subversion client by default so you will need to install a Subversion package. We installed the 1.7.11 "client only" install from Wandisco.This installs Subversion, including the javahl components, in C:\Program Files\WANdisco\Subversion. It is interesting to note that the JavaHL package in this install does not include the org.tigris package adapter layer.

Ubuntu 12

Ubuntu provides pckages for both core subversion and the JavaHL library for Subversion. We installed these for Subversion 1.7.5 as follows:

- sudo apt-get install subversion
- sudo apt-get install libsvn-java

Unfortunately the version installed seems to have a consistent assertion failue:

```
java: /build/buildd/subversion-1.7.5/subversion/libsvn_subr/dirent_uri.c:1519:
uri_skip_ancestor: Assertion `svn_uri_is_canonical(child_uri, ((void *)0))' failed.
```

We then removed the two Subversion packages from Ubuntu itself:

- `sudo apt-get remove libsvn-java`
- `sudo apt-get remove subversion`

We installed the Wandisco packages by downloading and running the Wandisco installer:

```
svn1.7_ubuntu_wandisco-precise.sh
```

This configures the Wandisco servers as a source of packages and installs the core Subversion install. At the time of writing this installed 1.7.11. Once installed, reinstall the javahl package:

- `sudo apt-get install libsvn-java`

This will now come from the Wandisco package repository. The location of the shared library and JavaHL jar is:

```
/usr/lib/jni/libsvnjavahl-1.so
/usr/share/java/svn-javahl.jar
```

CentOS 6.4

If you install the Subversion packages (subversion and subversion-javahl) using yum, you will have a 1.6.11 install of Subversion which is not compatible with FishEye if you wish to use JavaHL as descibed above.

If you have previously upgraded to a version of SVN 1.7 before 1.7.11 you may see the message below in your logs. If you do, please upgrade to the Wandisco SVN 1.7 as described below:

```
java: /build/buildd/subversion-1.7.5/subversion/libsvn_subr/dirent_uri.c:1519:
uri_skip_ancestor: Assertion `svn_uri_is_canonical(child_uri, ((void *)0))' failed.
```

You will need to remove the standard yum packages and use a Wandisco install, svn1.7_centos6_wandisco.sh. This installs plain subversion and configures the Wandisco servers as a source of packages. You can then use yum to install subversion-javahl. The following files are installed:

```
$ repoquery --list subversion-javahl
/usr/lib/libsvnjavahl-1.so
/usr/lib/svn-javahl/svn-javahl.jar
/usr/lib64/libsvnjavahl-1.so
/usr/lib64/svn-javahl/svn-javahl.jar
```

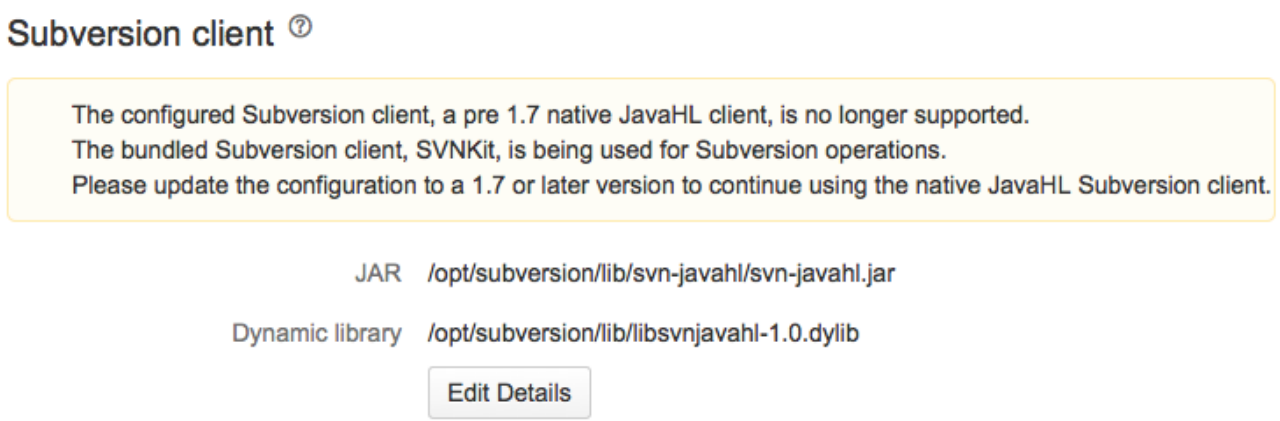
If you are using a 64bit JVM, use the /usr/lib64 library, otherwise use the 32bit library in /usr/lib.

JavaHL considerations when upgrading FishEye

If you are currently using SVNKit with FishEye (the default), then you do not have to do anything when upgrading to FishEye 3.1 and later. FishEye will continue to use the bundled SVNKit library.

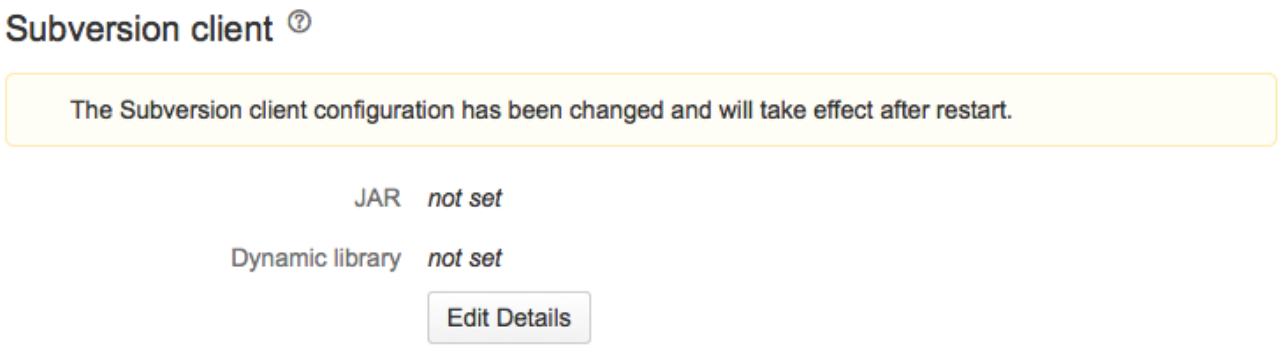
FishEye's Admin UI now displays information about the Subversion client in use – click **Server**, under 'Global Settings'. With no native client, configured, the display would look like:

If you have been using native JavaHL prior to FishEye 3.1, FishEye will detect that you have configured a pre-1.7 version of JavaHL and fallback to the bundled SVNKit client and start up normally. You will see the following in the **Server** section of the admin UI:



You can use the FishEye admin UI to update the JavaHL client information to point FishEye to a 1.7 or later JavaHL jar and shared library. FishEye will perform some checks that the configured library supplies the correct

classes. You will need to restart for the changes to take effect. If there are problems with the JavaHL library on restart, FishEye will again fallback to SVNKit. Once you have updated the configuration, FishEye will show a message that the configuration has been changed and a restart is required:



Upon restart, the display will show the operation of the native library and its version:

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.

We recommend that 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. We recommend that this path does *not* contain spaces, and 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 the Java executable as the **Variable value**, that is, something like `C:\Java\bin`. Don't use a trailing backslash. We recommend that JAVA_HOME should point to the Java executable specified in your PATH variable.
4. Now, in a new command prompt, try running '`%JAVA_HOME%\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 – this *is different from* the `<FishEye home`

directory>, which is where you installed FishEye.

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.

For production installations, we recommend that the FishEye instance directory be secured against unauthorised access.

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.

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. See also How to reset the Administration Page password in Fisheye or Crucible.

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:

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 or <FishEye home directory>\wrapper\bin, etc and NOT <FishEye home directory>\wrapper\wrapper or <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 (for example, if JAVA_HOME is defined as C:/Java/bin, then you can
use:
wrapper.java.command=%JAVA_HOME%/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=fa
lse
wrapper.java.additional.7=-Dcom.sun.management.jmxremote.ssl=false
wrapper.java.additional.8=-Dcom.sun.management.jmxremote.password.file=.
/wrapper/jmxremote.password
wrapper.java.additional.9=-Dwrapper.mbean.name="wrapper:type=Java
Service Wrapper Control"

wrapper.java.additional.10=-XX:PermSize=256m
```

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 later (**1.7.0** or later for OpenJDK).

If you don't see a supported version of Java, then get Java...

Download and install the Oracle Java Platform JDK, or OpenJDK.

Now try running '`java -version`' again to check the installation. The version of Java should be **1.6.0** or later (**1.7.0** or later for OpenJDK).

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

Do either of the following:
• If `JAVA_HOME` is not set, log in with 'root' level permissions and run:
`echo JAVA_HOME="path/to/JAVA_HOME" >> /etc/environment`
where `path/to/JAVA_HOME` may be like: `/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/`
• If `JAVA_HOME` needs to be changed, open the `/etc/environment` file in a text editor and modify the value for `JAVA_HOME` to:
`JAVA_HOME="path/to/JAVA_HOME"`
It should look like:
`JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/`
	Insert the following in your `~/.profile` file:
	`JAVA_HOME="path/to/JAVA_HOME"`
	`export JAVA_HOME`
	where `path/to/JAVA_HOME` may be like: `/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/`
	Refresh your ~/.profile in the terminal and confirm that `JAVA_HOME` is set:
	`source ~/.profile`
	`$JAVA_HOME/bin/java -version`
	You should see a version of Java that is **1.6.0** or higher, like this:
	`java version "1.6.0_24"`

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.

For production installations, we recommend that the FishEye instance directory be secured against unauthorised access.

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

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, and 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

This page assumes that:

- You have installed and started the latest version of FishEye. See Installing FishEye on Linux and Mac or Installing FishEye on Windows for details.
- You are using a supported browser.

On this page:

Related pages:

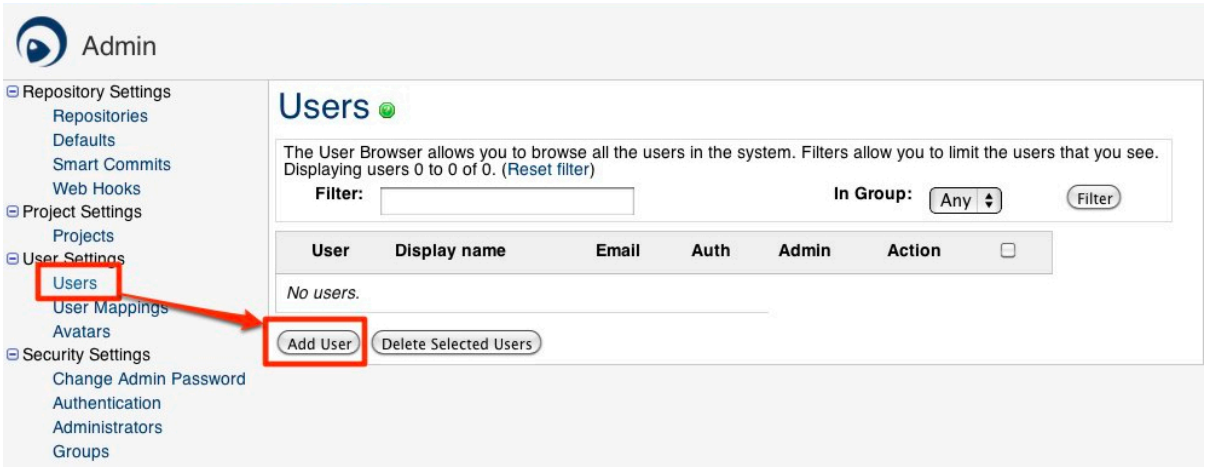
- Installing FishEye on Windows
- Installing FishEye on Linux and Mac
- Supported platforms
- Managing your repositories
- Setting up your Users and Security

Create users in FishEye

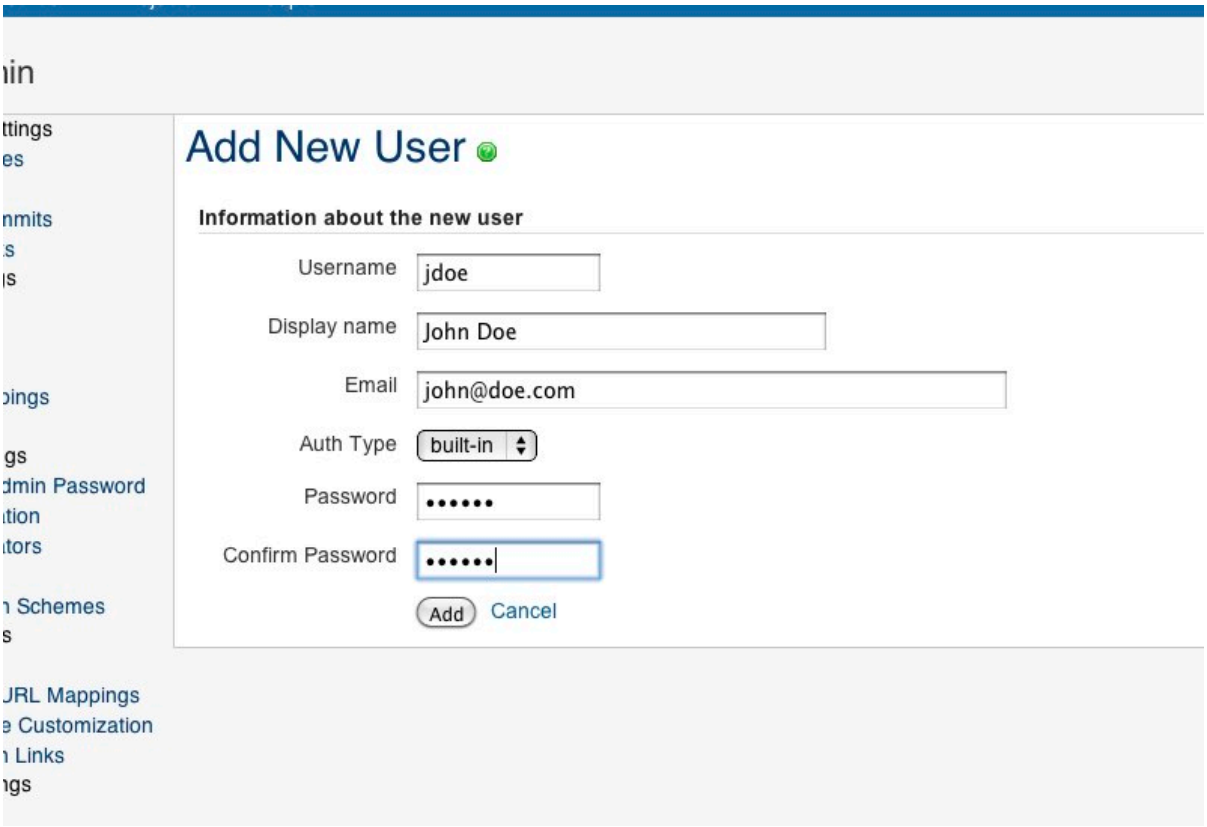
FishEye doesn't have any user accounts after you have installed it for the first time. You need to go to the Administration interface to add the first users of the system.

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

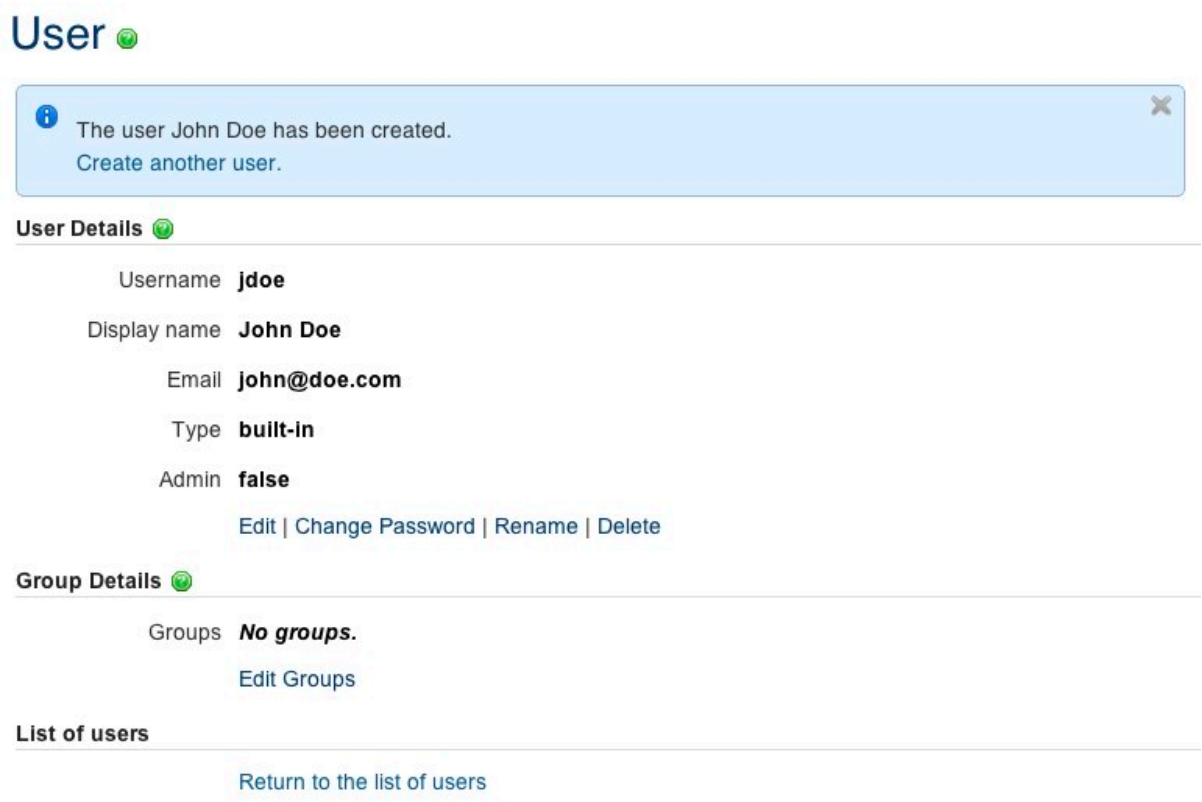
In the **Users** listing page click **Add User** to go to the user creation form:



Fill in the form and create your user:



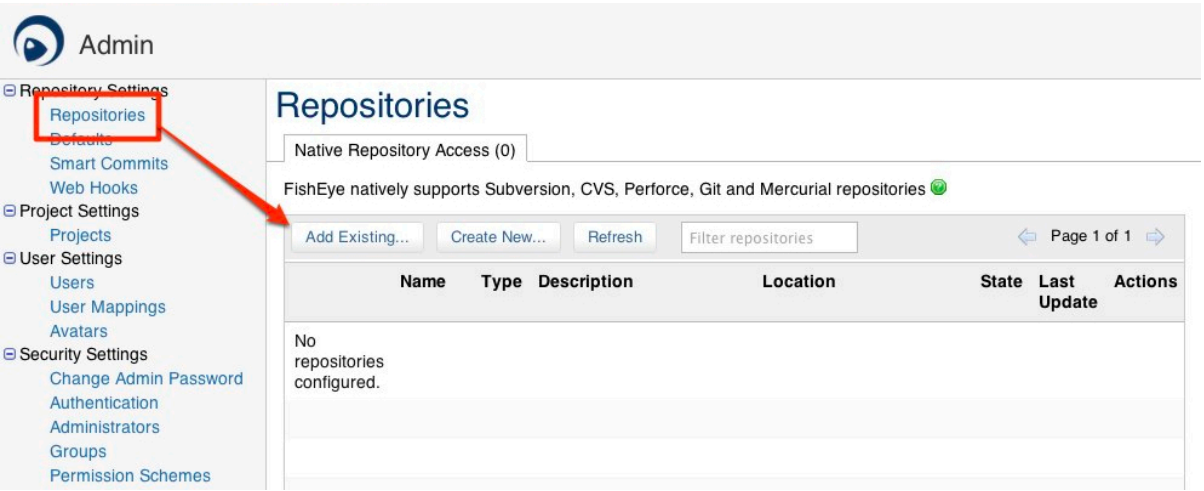
From the **User** page you can click on **Create another user** to repeat this operation:



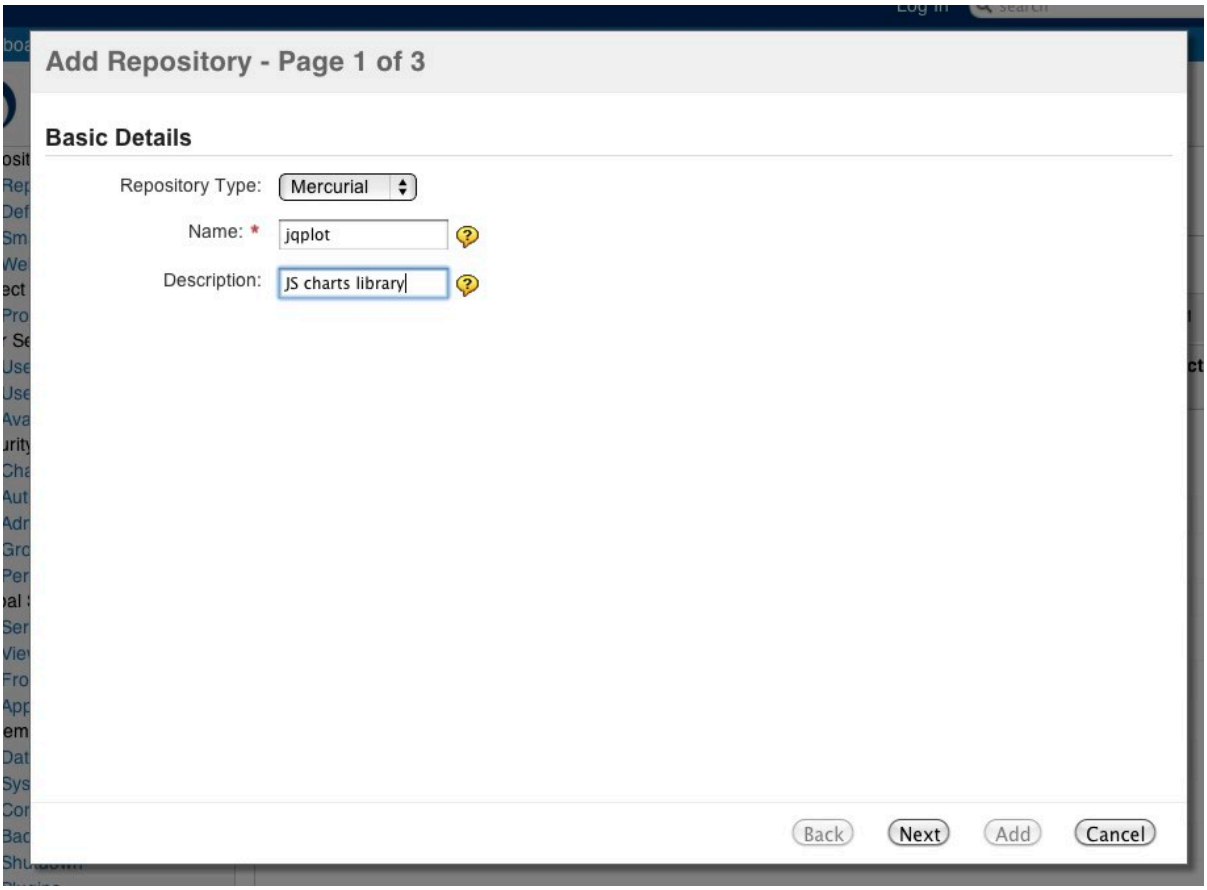
Add a repository

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

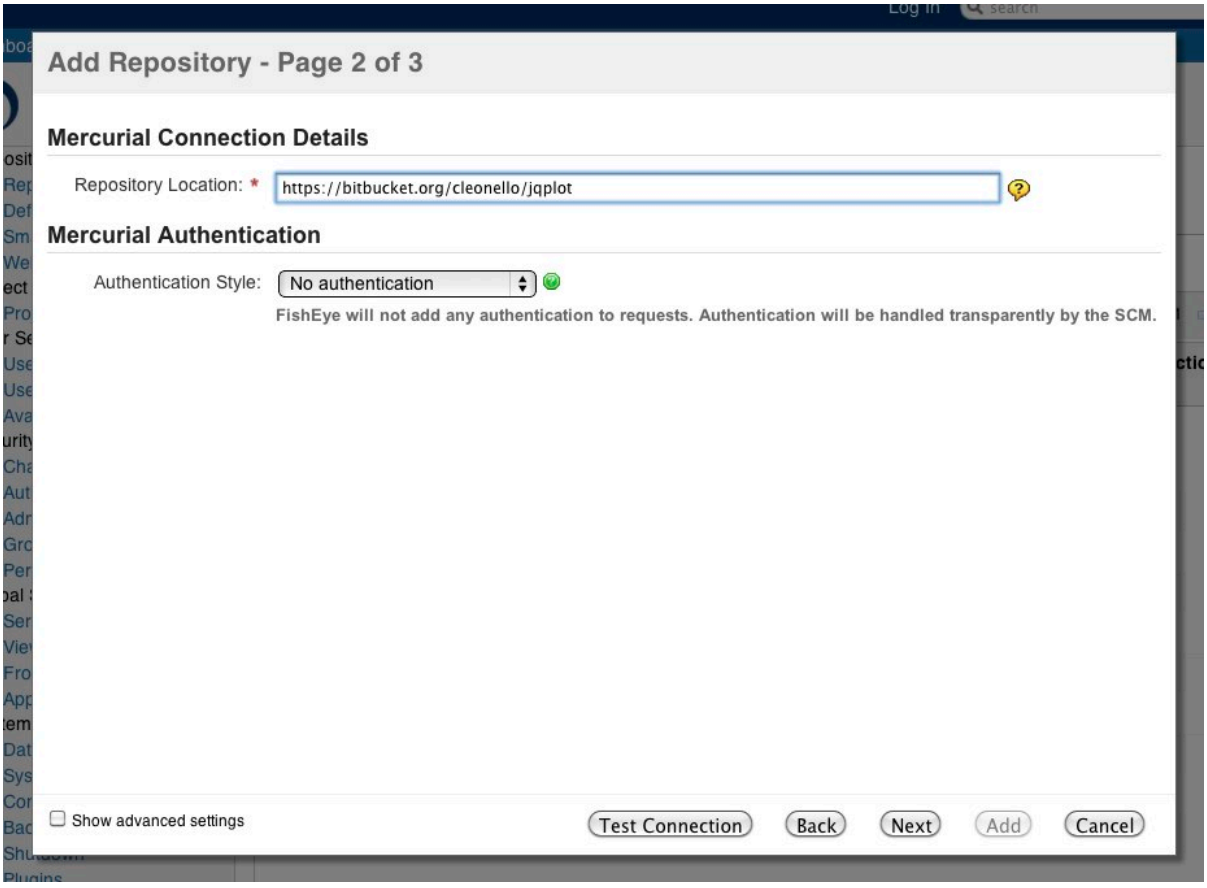
Click on **Add Existing...** in the **Repositories** listing of the Administration:



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

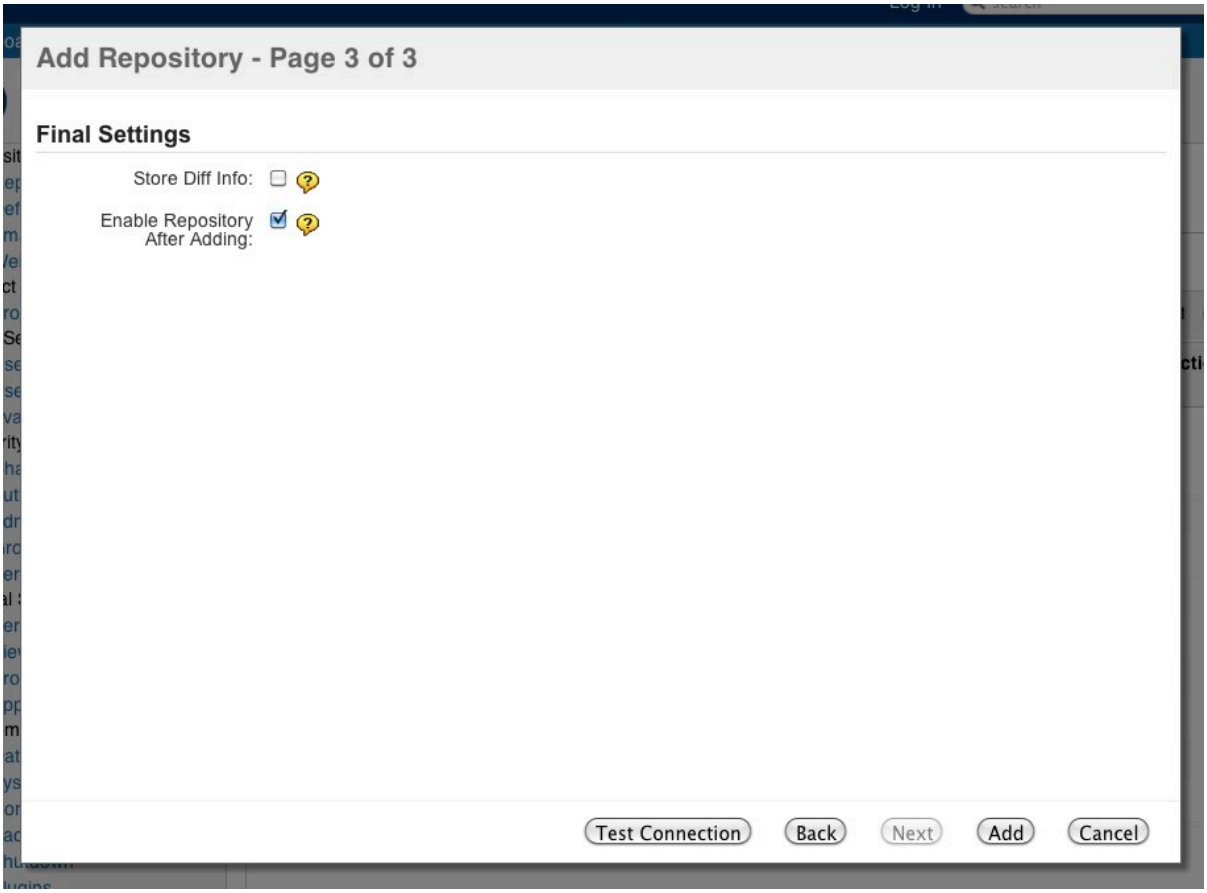


In the repository configuration, add the location of your repository. Fill in the authentication details if necessary.



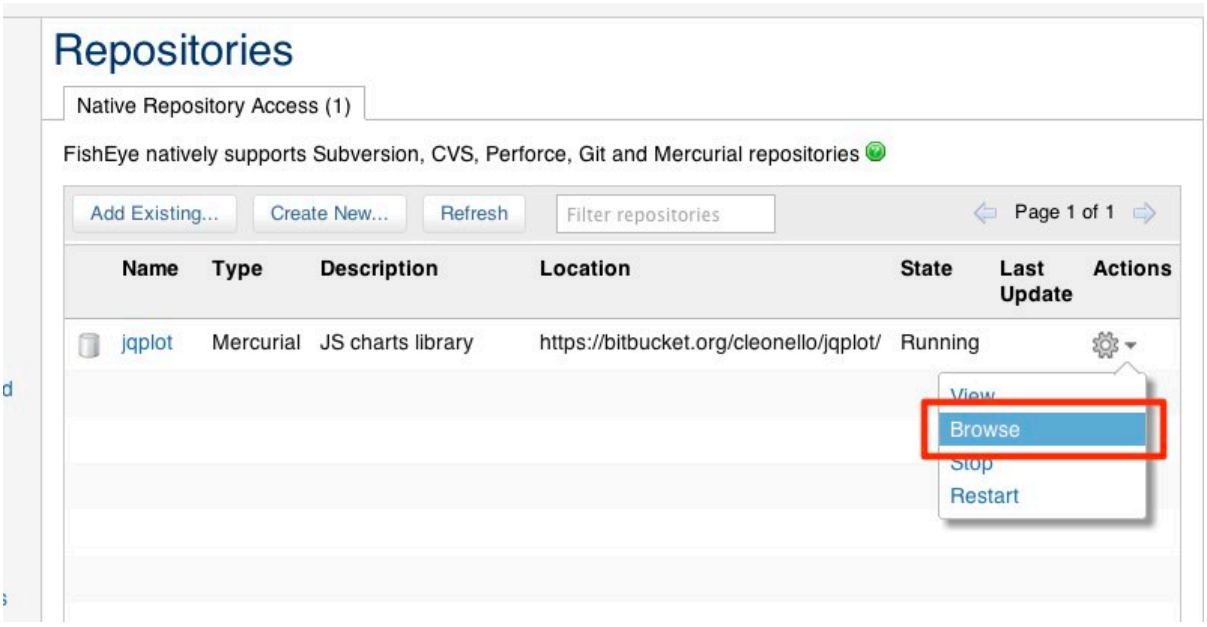
Finally indicate whether or not you would like diff indexing should be turned on and if the repository should not be indexed right away.

Click **Add** to finish the process.

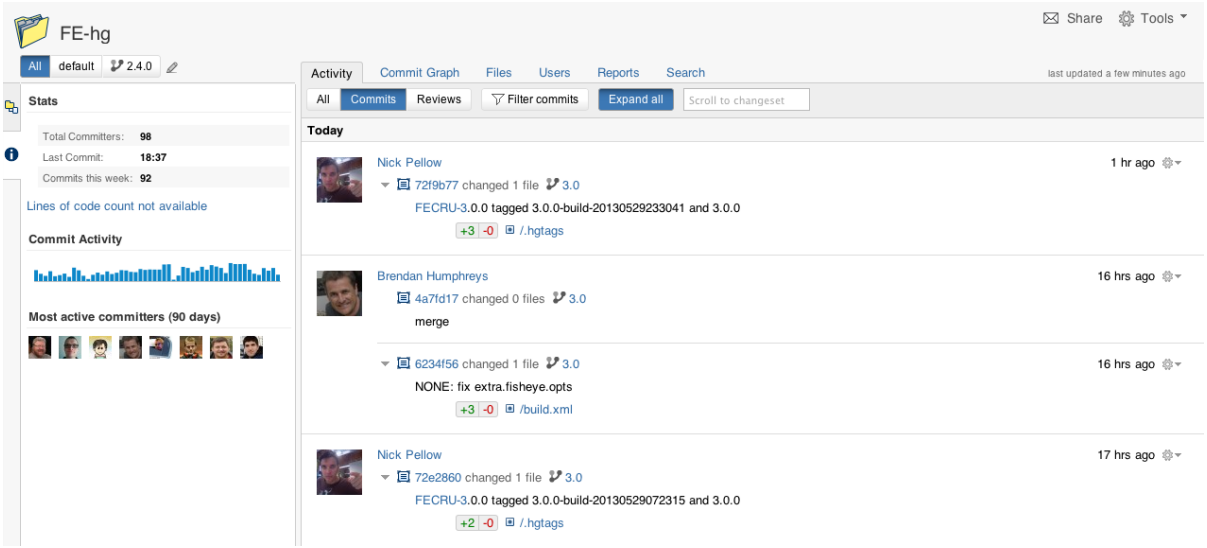


Move forward

Once it's created you can click **Browse**, in 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:

Related pages:

- Starting to use FishEye
- Linking FishEye to JIRA
- 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:

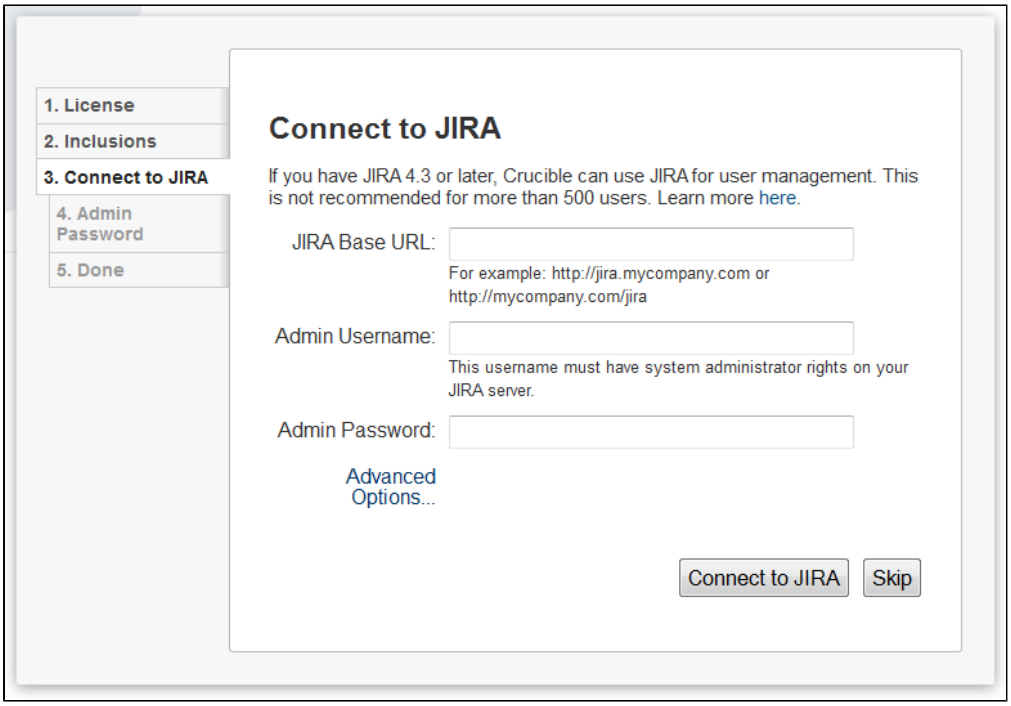
1. Configure the following setting in JIRA: Allow remote API access.
2. Enter the following information on the 'Connect to JIRA' step of the setup wizard:

JIRA base URL	The base URL set for 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 `jira-users`. 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 `jira-administrators`.

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



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

The setup wizard displays one of the following error messages: • 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>.	The setup wizard failed to complete registration of the peer-to-peer application link with JIRA. JIRA integration is only partially configured.	Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.
The setup wizard displays one of the following error messages: • 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>	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.	Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.
The setup wizard displays the following error message: • Error setting Crowd authentication	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 `config.xml` file. This may be caused by a problem in your environment, such as a full disk.	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.
The setup wizard displays the following error message: • Error reloading Crowd authentication	The setup wizard has completed the integration of your application with JIRA, but is unable to start synchronizing the JIRA users with your application.	Restart your application. You should then be able to continue with the setup wizard. If this solution does not work, please contact Atlassian Support.
The setup wizard displays the following error message: • An error occurred: java.lang.IllegalStateException: Could not create the application in JIRA/Crowd (code: 500). Please refer to the logs for details.	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.	Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.

No users can log in after you have set up the application with JIRA integration.	Possible causes: • 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.	Go to JIRA and add some usernames to the group. • 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. If this solution does not work, please contact Atlassian Support.

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 **Delete** next to the application link that you want to delete.
    d. A confirmation screen will appear. Click **Confirm** 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 configruration 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 **Delete** next to the application link that you want to delete.
   d. A confirmation screen will appear. Click **Confirm** 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:
   a. Click **Add Application Link**. Step 1 of the link wizard will appear.
   b. Enter the **server URL** of the application that you want to link to (the 'remote application').
   c. Click **Next**.
   d. Enter the following information:
      - **Create a link back to this server** – Check 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.
   e. Click **Next**.
   f. Enter the information required to configure authentication for your application link:
      - **The servers have the same set of users** – Check this box, because the users are the same in both applications.
      - **These servers fully trust each other** – Check this 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.
   g. Click **Create**.
7. Configure a new connection for user management in JIRA:
   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. **Add** an application.
   c. Enter the **application name** and **password** that your application will use when accessing JIRA.

d. 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:
	`http://www.example.com:8080/jira/`
	`http://jira.example.com`

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 area.
  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 Stash 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

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 each FishEye page provides the following:

- The application navigator, at the left of the header, connects you directly to your other applications, such as JIRA and Bamboo. Admins can configure which apps appear in the navigator – just click **Application navigator** in the admin area.
- FishEye logo – click to go to the dashboard, to see your personal code commits, your reviews (if you are using Crucible) and your activity stream.
- **Repositories** — the list of all FishEye repositories. Click a repository name to browse the repository. A number of sub-tabs become available, as described below (see 'Repository sub-tabs' below).
- **Projects (***when used with Crucible)* – a link to all projects (see the Crucible documentation). Logged-in users can see links to recently visited projects.
- **People** – tab to view statistics about commiters to your FishEye repositories (see Viewing People's Statistics). Logged in users can see links to users they have recently visited.
- **Reviews** (*when used with Crucible)* – go to your code reviews (see the Crucible documentation). Logged-in users can see links to recently visited reviews as well as to the Crucible '*Inbox*' and '*Outbox*'.
- **Create review** (*when used with Crucible)* – click the down arrow to choose **Create snippet**.
- Click your avatar to change your user settings (see Changing your user profile).

Repository tabs

Once you have selected a repository, you can navigate through it by selecting files and folders in the navigation tree on the left. The available tabs change according to whether you are viewing a repository or a file:

Tab	Repo	File	Description
Files	✓		Provides details about the files in the repository. See Browsing through a repository.
Revisions		✓	Shows the latest revisions of the file. See Viewing a file history.

Activity	✓	✓	Shows recent activity on the item. There are a number of sub-options here (see Viewing the changelog): • All — The default view, showing commits, reviews (when used with Crucible) and JIRA issues (when used with JIRA). • Commits — Shows commits in the activity stream. • Reviews — Shows review activity in the activity stream (when used with Crucible). • Filter — Applies constraints to the current activity stream. • Scroll to Changeset — Opens the changeset ID specified in the text field (press Enter to carry out the action).
Commit graph	✓		Provides a visual representation of commits to, and branches in, the repository. See Viewing the commit graph for a repository.
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. See Viewing file content.
Search	✓		Gives access to the advanced search capabilities in FishEye.

Browsing through a repository

You can use FishEye to select a repository and browse through it. The repository view provides information about the files in the repository and activity related to the repository, including commits to the repository. You

can also generate charts and search for specific file revisions in the repository.

On this page:

Browsing a repository

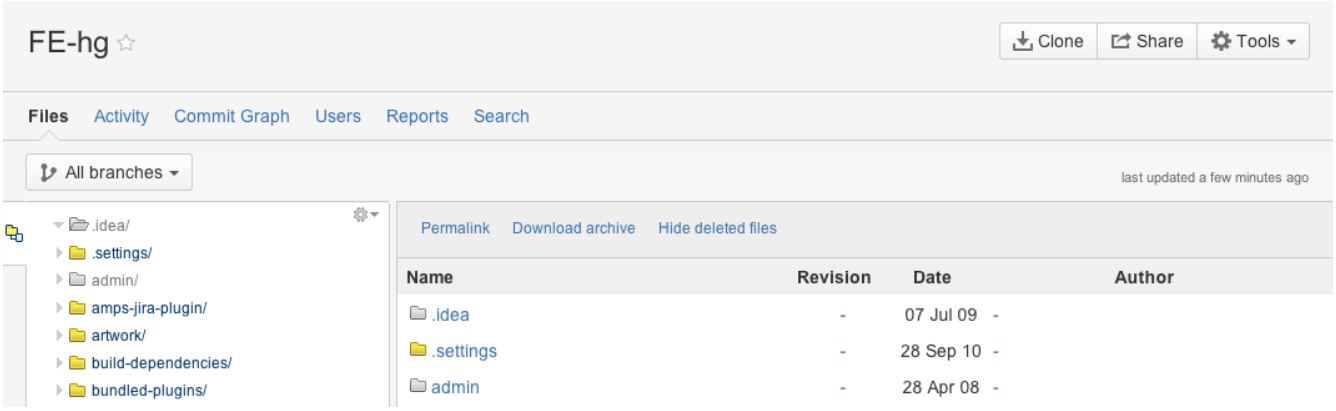
To browse a repository:

1. Click **Repositories** in the header and choose either a recently viewed repository, or **All repositories**.
2. Click the name of a repository to view its contents. See the 'Browsing a repository' screenshot below.
3. If required, use the branch/tag selector (just above the file tree) to choose the branch or tag that you want to browse the changelog for.
4. You can view different 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.
   A greyed out item is either a deleted file or an empty/deleted (folder).

Files tab	View the contents of the repository/folder being viewed.
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.
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.

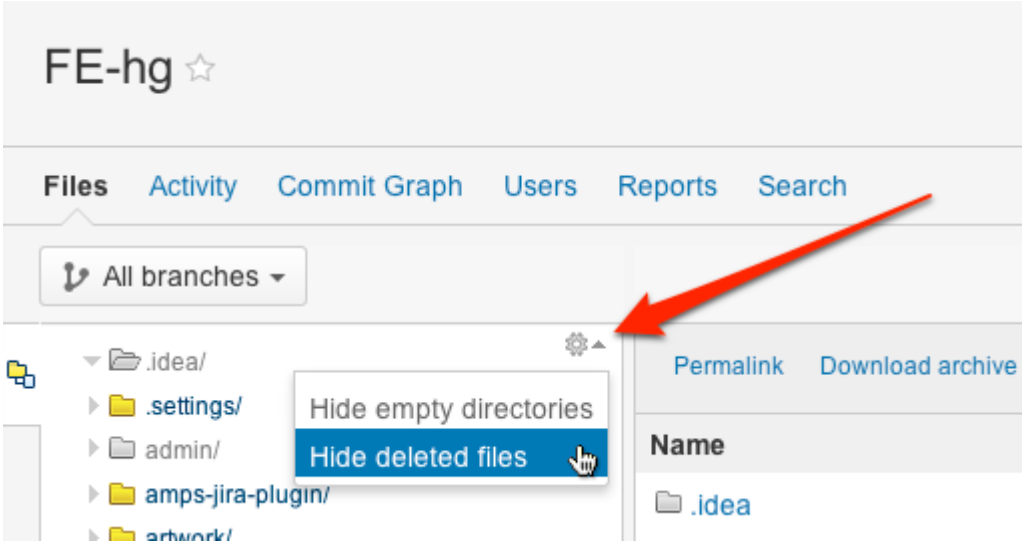
You can choose to hide deleted files and empty directories in the left navigation tree when browsing through a repository, as described below.

To hide deleted directories or files in your navigation tree:

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

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 or 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. Choose **Tools** > **Watch**. (The watch icon becomes coloured, not grey).

To remove the watch, choose **Tools** > **Unwatch**. You can also remove watches from within your user profile.

Searching FishEye

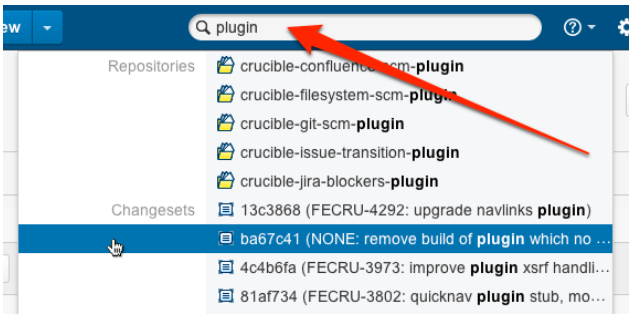
FishEye has a powerful search engine that allows you to find changesets, committers and files. There are two methods for searching in FishEye:

Quick Search

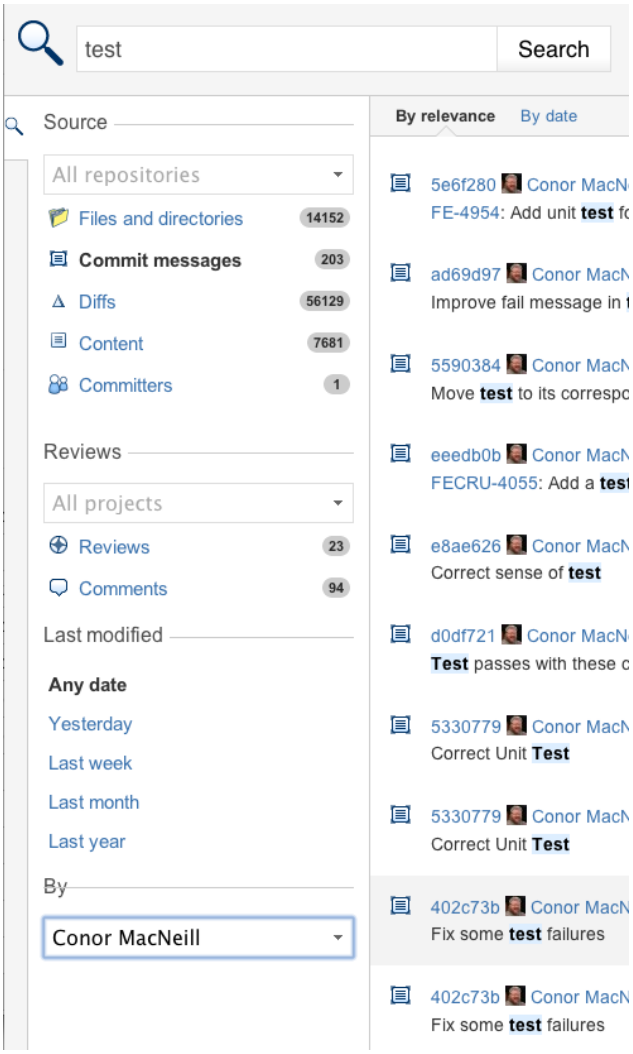
- Searches across all repositories connected to FishEye.
- Suggests "quick nav" results as you type.

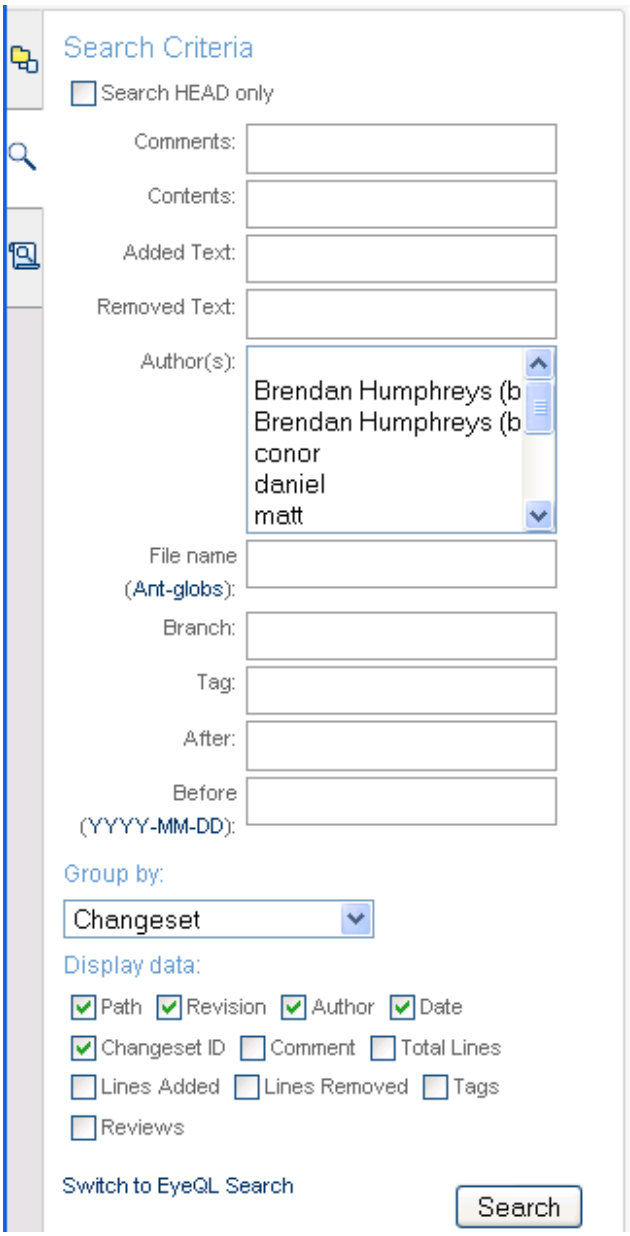
Advanced Search

- Click on the **Search** tab.
- Searches a single repository.
- Enter search criteria for a range of attributes:



- Has extra search tools, such as 'handles'.
- Press <Enter> to see filters:





Using the Quick Search

Before you begin:

- This is the default search and will suggest "quick nav" results (from the header search box only). Results are weighted by most recent edit date; files edited within the last twelve months are given greater weighting.
- There is a 100-repository limitation on searches, to prevent it from becoming unresponsive on FishEye instances that have large numbers of repositories. 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.

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.

Tool	Description	Example
CamelCase Pattern Matching	Enter a CamelCase string pattern to find files and directories that match.	`BooQCTest` returns results like `BooleanQueryCoordTest` and `BooleanQueryClassTest`.
Path/File Pattern Matching	Enter a path/file pattern to find files and directories that match.	`common/final/Actions` returns results like `/src/common/eu/systemworks/specialprojects/final/Actions.java`.
Field Handles	Use a field handle to restrict your search to a particular field: • `file` — file/directory names • `commit` — commit messages • `diff` — lines added/removed • `content` — file contents • `committer` — committer names	`file:build.xml` returns files with names matching `build.xml`.
Search within a Directory (AntGlobs)	AntGlobs can be used in the Quick Search to search within a specific directory.	Search for `/src/**/gwt/*.xml` and FishEye will return all files with a `.xml` suffix that are below both a src and a gwt directory. e.g. **`src`**`/java/com/atlassian/fecru/`**`gwt`**`/FecruCore.gwt.`**`xml`** but not **`src`**`/java/com/atlassian/fecru/ApplicationContext.xml`
Search for Discrete Strings	Enter a string in quotation marks to find the exact string. Not case-sensitive.	`"update version in build"` returns matches for the exact string, i.e. it will not return `"update build version"` or `"update version in file"`.

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
All repositories	By default, searches across all repositories. Choose a repo to restrict the search to just that one. For example, if you enter 'FE' then the search results page will refresh to display only files and directories in the 'FE' repository.
Files and directories	Filter your results to files and directories that have names that match the search criteria.
Commit messages	Filter your results to changeset comments that match the search criteria.
Diffs	Filter your results to diffs (lines added/removed) that match the search criteria.
Content	Filter your results to files that have content that match the search criteria.
Committers	Filter your results to committers that match the search criteria.
All projects	When using Crucible with FishEye, there is a 'Revie ws' section. The **All projects** dropdown allows you to filter reviews returned in the search results. See Searching Crucible.
Reviews	Search in reviews
Comments	Search for review comments
Last modified	Filter by the date of the last change.
Author	Filter by author name.

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 in Browsing through a repository.
2. Click the **Search** tab.
3. Enter your search criteria:
    - **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.

    Use **Switch to EyeQL Search** and **Switch to Standard Search** at the bottom of the 'Search Criteria' panel to toggle between the two search methods.

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.

Please note the following:

- **Contents** — 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** — Antglobs can be used to specify the criteria for this field.

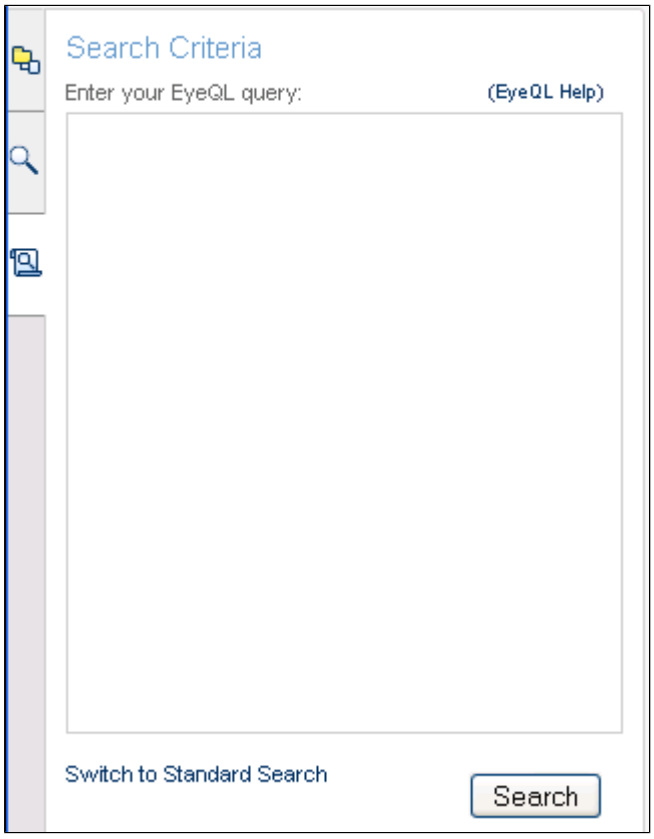
Specifying Search Criteria using EyeQL

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

Click **Switch to EyeQL Search** at the bottom of the 'Search Criteria' panel.

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: Advanced Search Criteria — EyeQL Interface



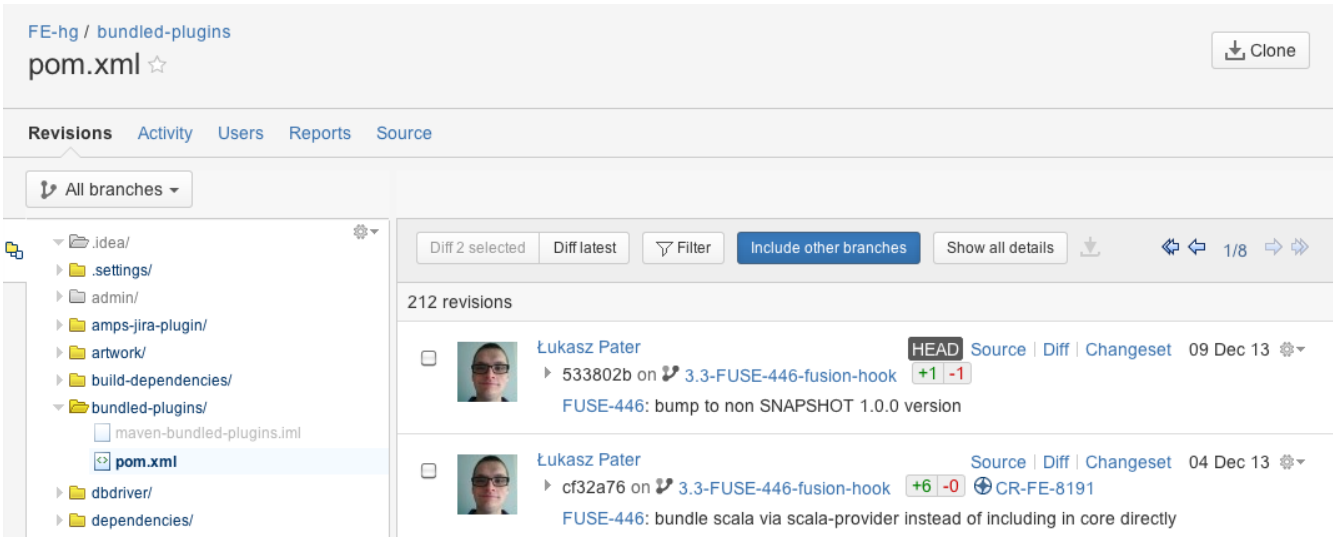
Viewing a file

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, or browse through a repository, to find the file.
2. Click the file name. The revision history for the file will be displayed. See the screenshot below.
3. View information about the file on these tabs:
    - **Revisions** — Displays the history of revisions for the file. See Viewing a file history.
    - **Activity** — Displays the commits and reviews activity related to the file. See Viewing the changelog.
    - **Users** — Displays commit histories for users who have committed changes to the file. See Viewing people's statistics.
    - **Reports** — Displays charts for the file activity. See FishEye Charts.
    - **Source** — Displays the annotated file contents. The raw file can be downloaded from this tab. See Viewing file content.

Screenshot: Viewing a file (Revisions tab)



Related Topics

- Viewing file content
- Viewing a file history
- Viewing the changelog

Viewing file content

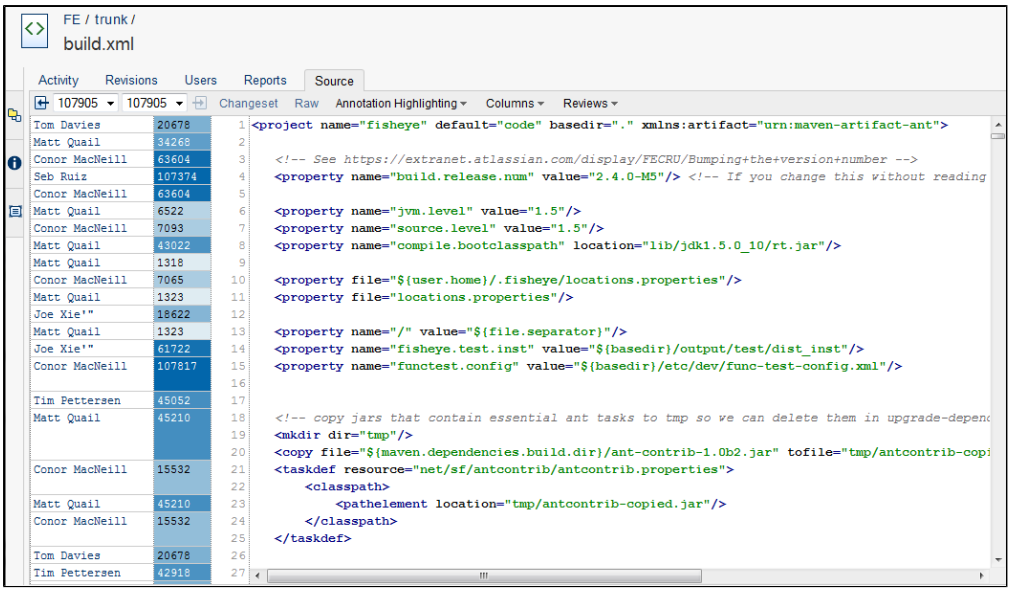
You can search or browse your repositories in FishEye to view a specific file. FishEye allows you to view and download the source code for the file. You can also view diffs between different revisions of the file and annotations.

To view the source code for a file:

1. Search, or browse through a repository, to find the file.
2. Click the file name. The revision history for the file will be displayed.
3. Click the **Source** tab.

Displaying the diff	Select revision numbers (e.g. '107905') from the two revision dropdowns to display the diff for those two revisions.
Changeset	View the changeset that the revision was a part of.
Raw	Download the raw source code for the file.
Annotation Highlighting	Choose **Age**, **Author** or **None** to color the annotations by age, author or remove highlighting respectively. The highlights are displayed over the revision numbers, next to the authors.
Columns	Select the columns to display: **Author**, **Revision** and **Line Number**.
Reviews	Select **Create Review** to create a Crucible review from the file. *(Requires Crucible)*

Screenshot: Viewing a file (Source tab)



Using Side by Side Diff View

This page describes FishEye's innovative 'side-by-side diff' view that shows how a file's content has changed, compared on the left and right hand sides of the screen.

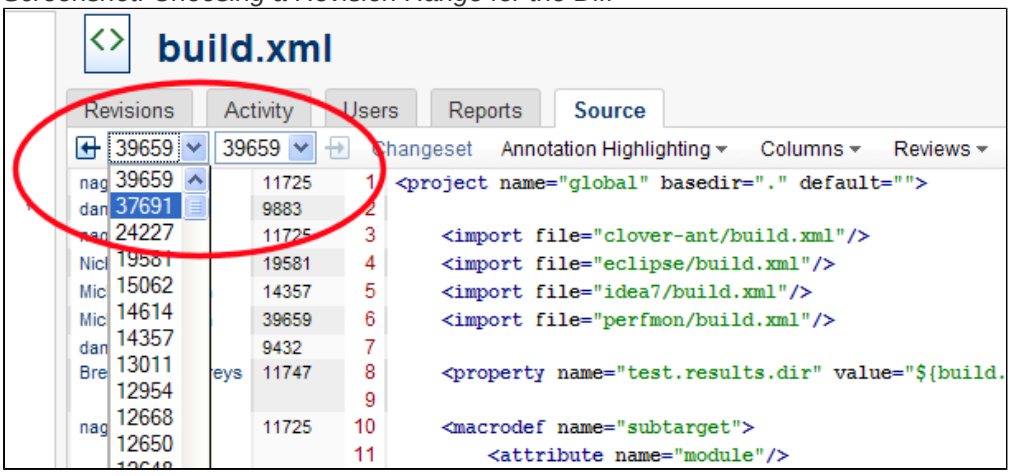
On this page:

Opening the side-by-side diff view

To open FishEye's side by side diff view:

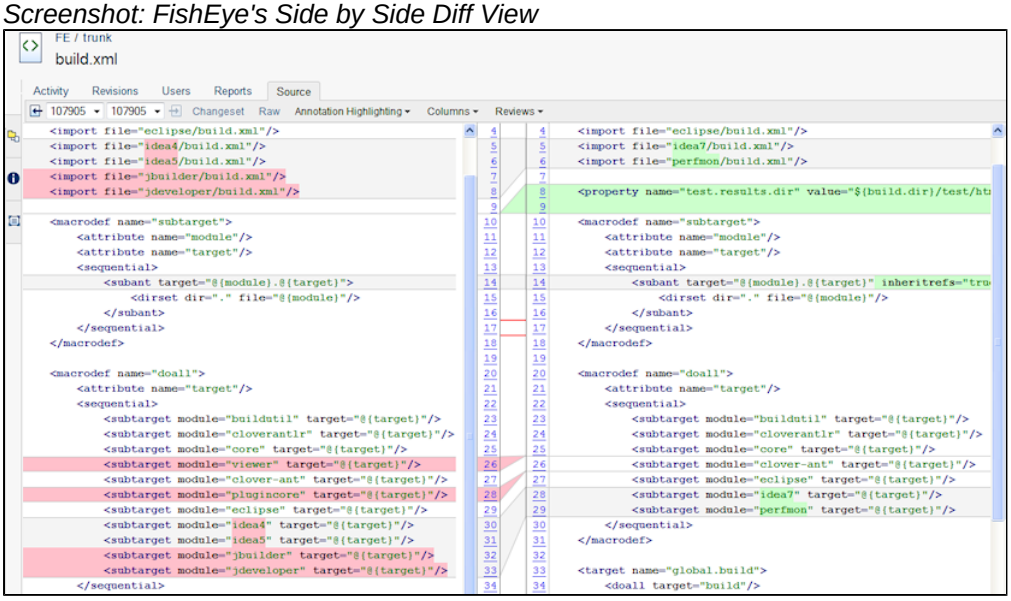
1. Open the source code view for the file in question.
2. Select a range of revisions to compare.
3. Choose **View** > **Side by Side Diff**.

Screenshot: Choosing a Revision Range for the Diff



The left and right panes of the side-by-side diff view 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 adjacent pane by a coloured triangle that shows the location of that change in the comparison file.

Screenshot: FishEye's Side by Side Diff View

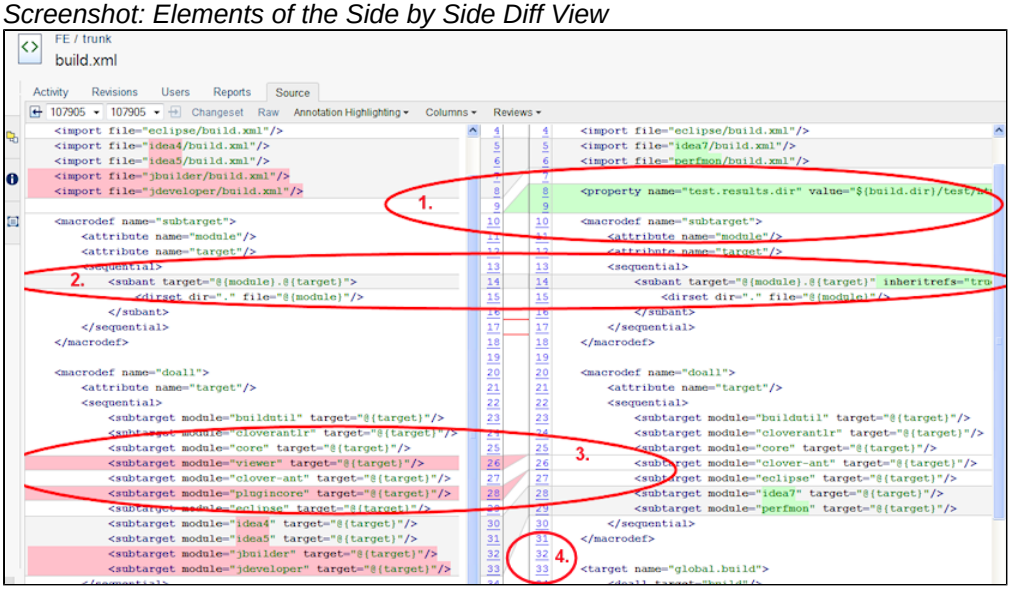


Understanding side-by-side diffs

Features of the side-by-side diff 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. Line numbers in the margin are permanent links ("permalinks") that can be 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 '**Di**

ff' 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.
2. Search, or browse through a repository, to find the file.
3. Click the **Revisions** tab. The history of revisions for the file will be displayed. See the 'File Revisions' screenshot below.

Diff 2 selected	Check boxes for two file revisions and then click to view the diff for those revisions.
Diff latest	View the diff of the two most recent file revisions.
Filter	View the file filter. Enter the desired fields to filter the file history results on.
Include other branches	Include revisions of the same file from 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.*
Show all details	Toggle expand or collapse of 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 an explanation of the information provided about individual revisions.

Screenshot: File Revisions

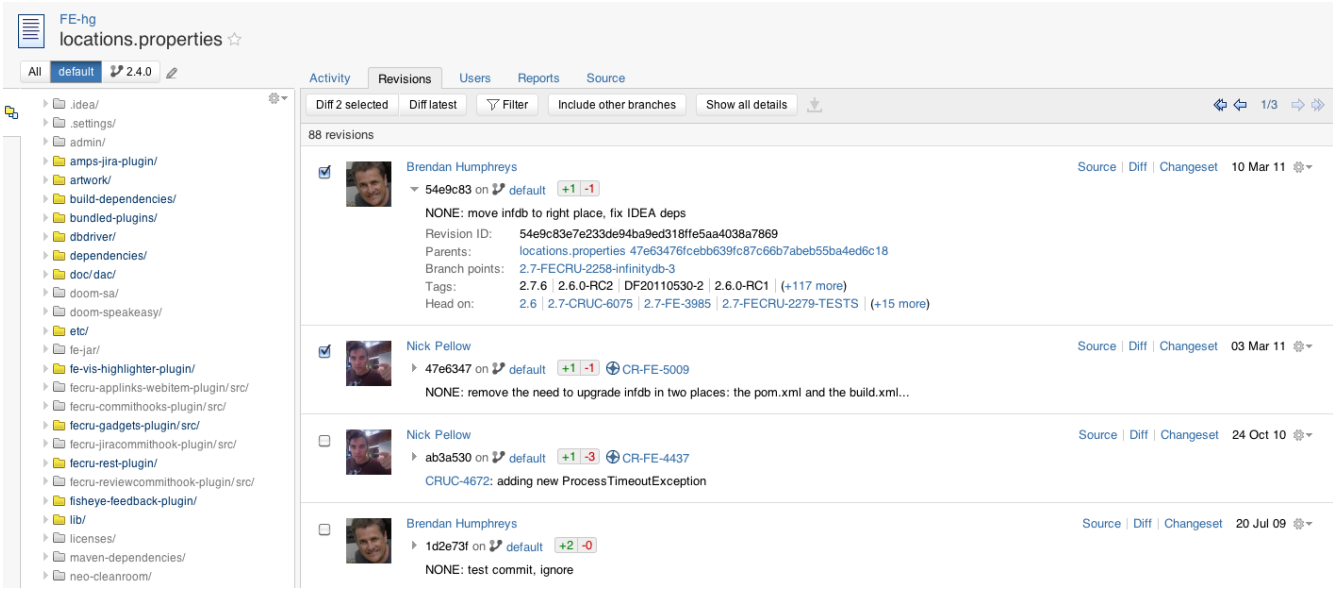
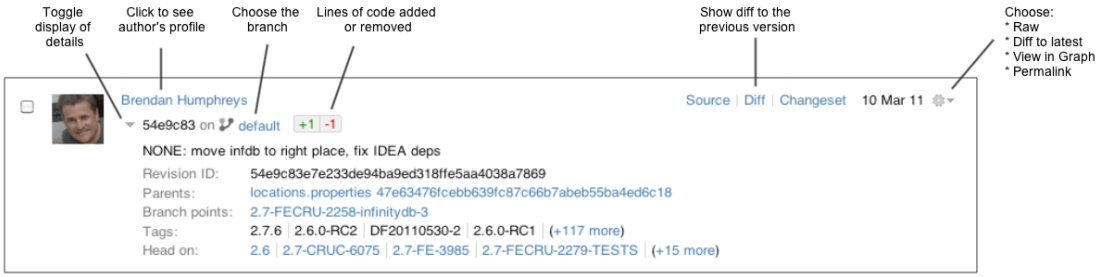


Diagram: Overview of a file revision



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:

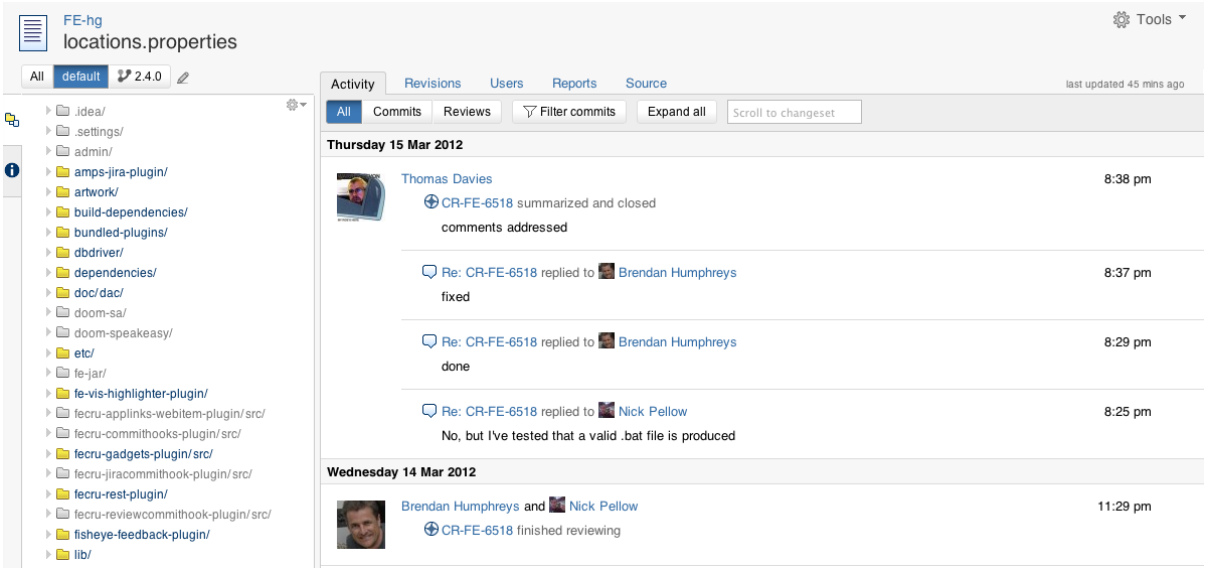
Viewing changelog activity

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 or file 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



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** t abs of the **Activity** tab. Note that you cannot use the commits filter to filter reviews.

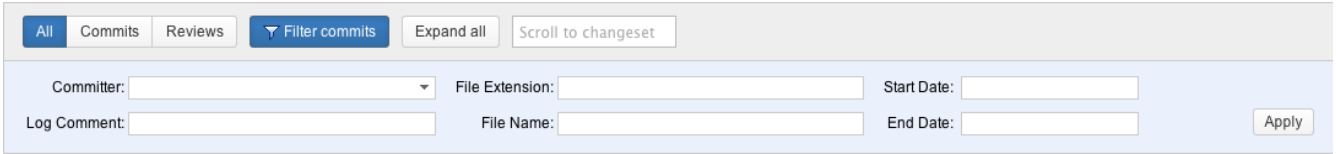
To filter commit activity:

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



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** tab 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

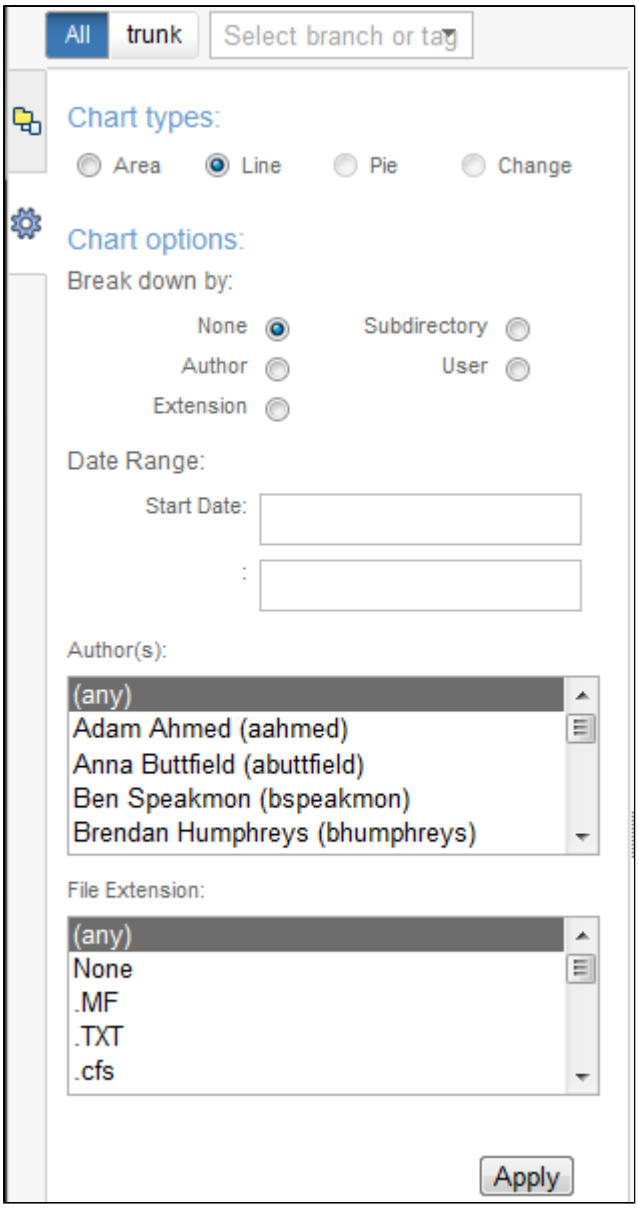
Click **Reports** and then **Charts** when browsing a repository to see charts of activity in the repository.

You can control the chart type that is displayed and various chart options. Click the cog icon on the left, select the required options, and click **Apply**. The available options include:

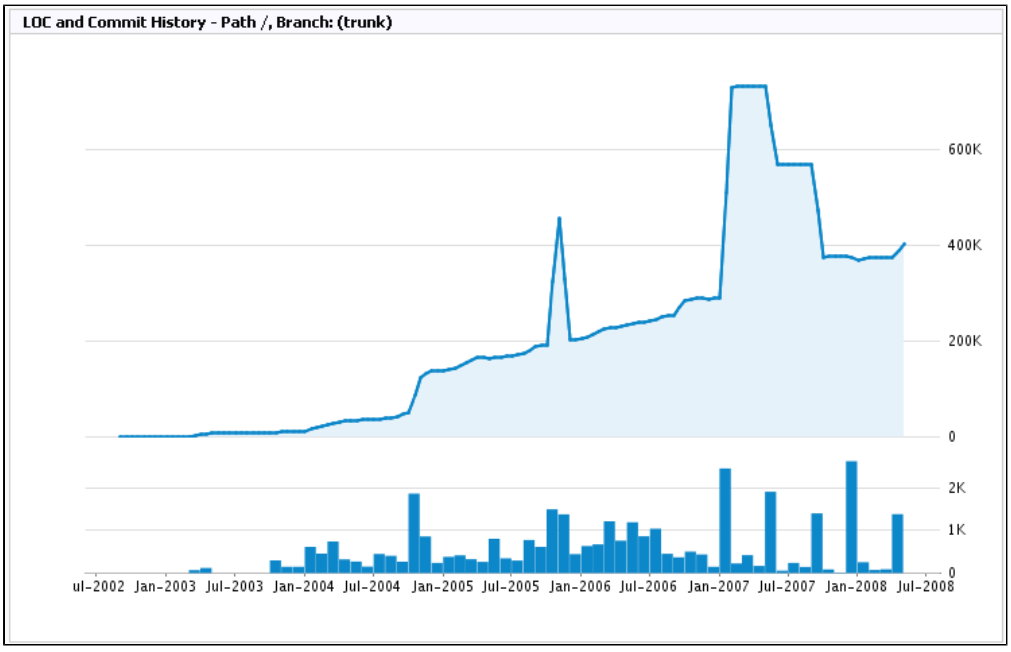
Setting	Explanation	Values	Default
Branch/Tag selector	Limits the chart to the selected branch/tag.	Any branch/tag from the current repository.	Displays the default/trunk.
Chart type	Changes the chart's presentation.	Area, line, pie or change* chart.	Area
Author	Limits the chart to show specific author(s).	Any author configured in the system.	All
Extension	Limits the chart to show specific file type(s).	Any file extension; e.g. '.java'.	All
Subdirectory	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)
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)

*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.

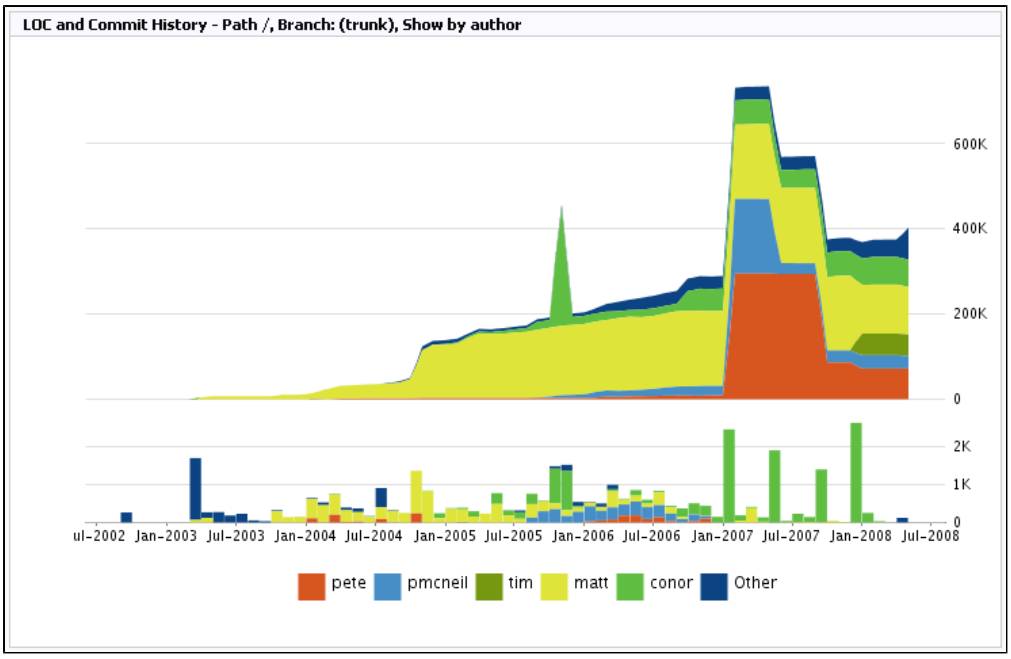
Screenshot: FishEye custom chart settings



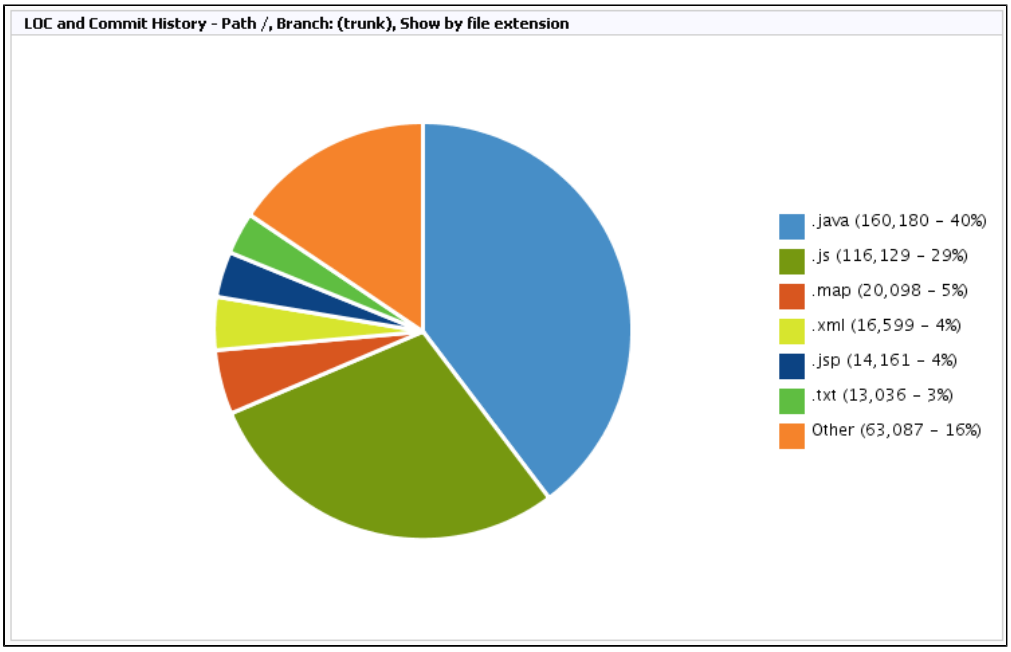
Screenshot: FishEye per-author LOC chart



Screenshot: FishEye per-author LOC chart showing multiple authors



Screenshot: FishEye LOC chart by file extension



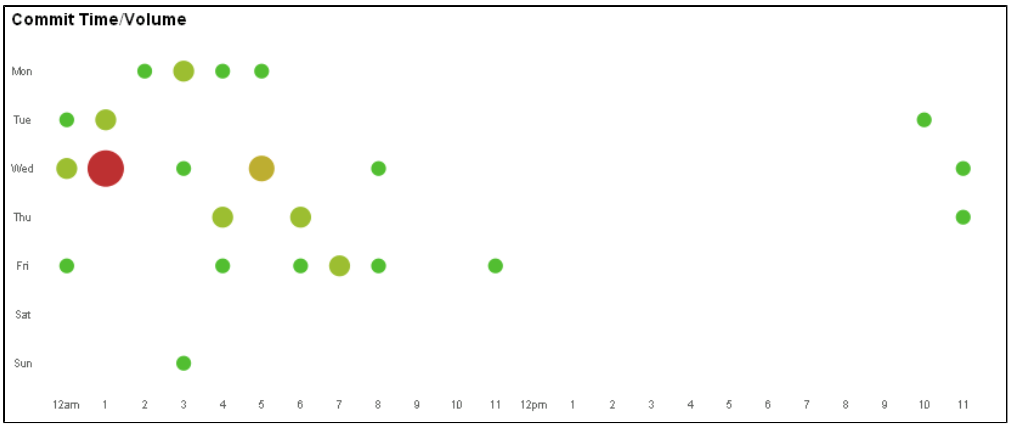
Per-Author Lines of Code Statistics

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

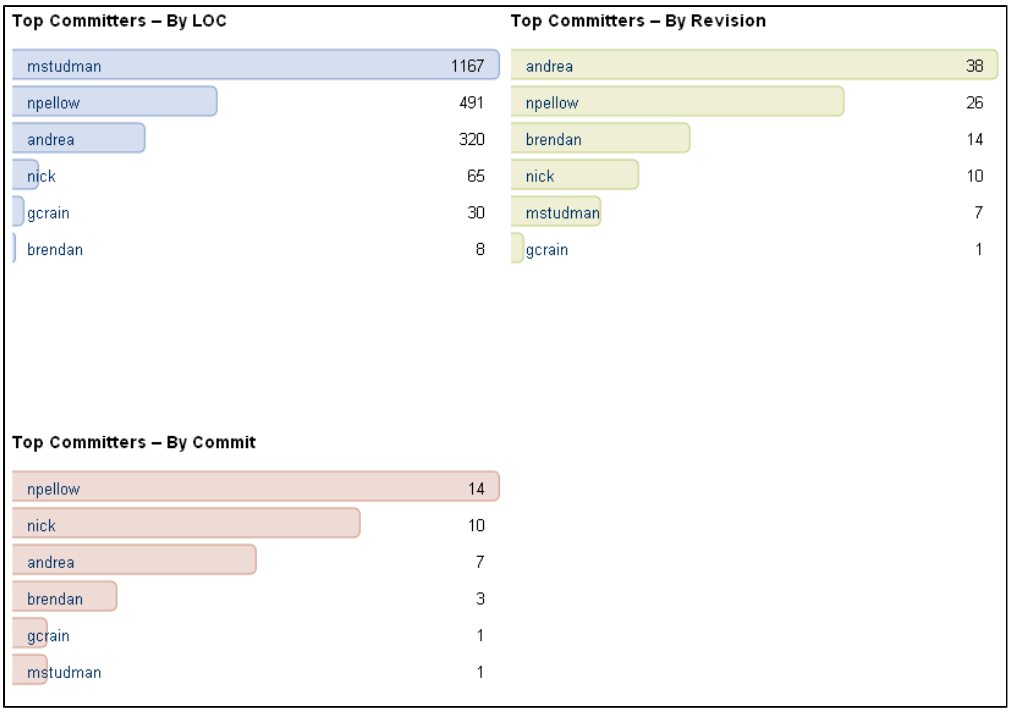
Code Metrics

A number of built-in reports are also provided:

Screenshot: Commit Time/Volume



Screenshot: Top Committers



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.

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

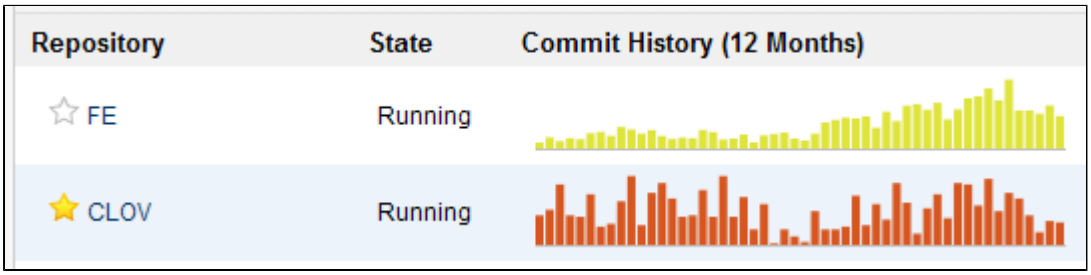
On this page:

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



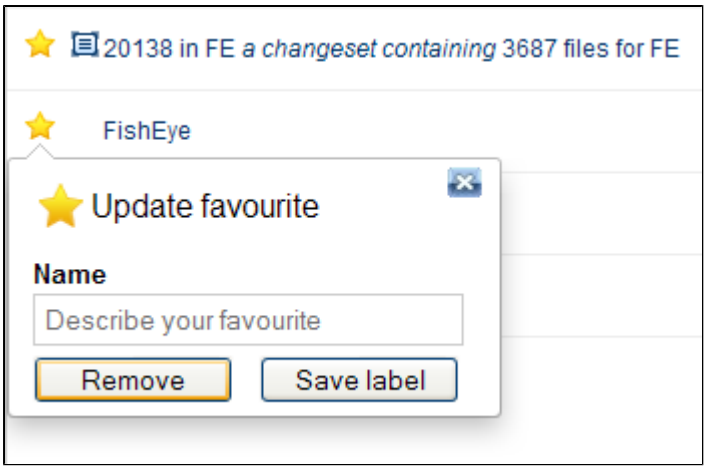
Managing favourites

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 (and if you are using Git or Mercurial, you will be able to see anonymous branches as well).

The Highlight feature of the commit graph allows you to highlight different types of information in the swimlanes

or changeset list:

- ancestors and descendants for a changeset
- commits with JIRA issues
- reviewed and unreviewed changesets.

For example, if you have the **JIRA issues** highlight active, clicking a changeset with a JIRA issue in the commit comment will show all other changesets with the same JIRA issue.

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:

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

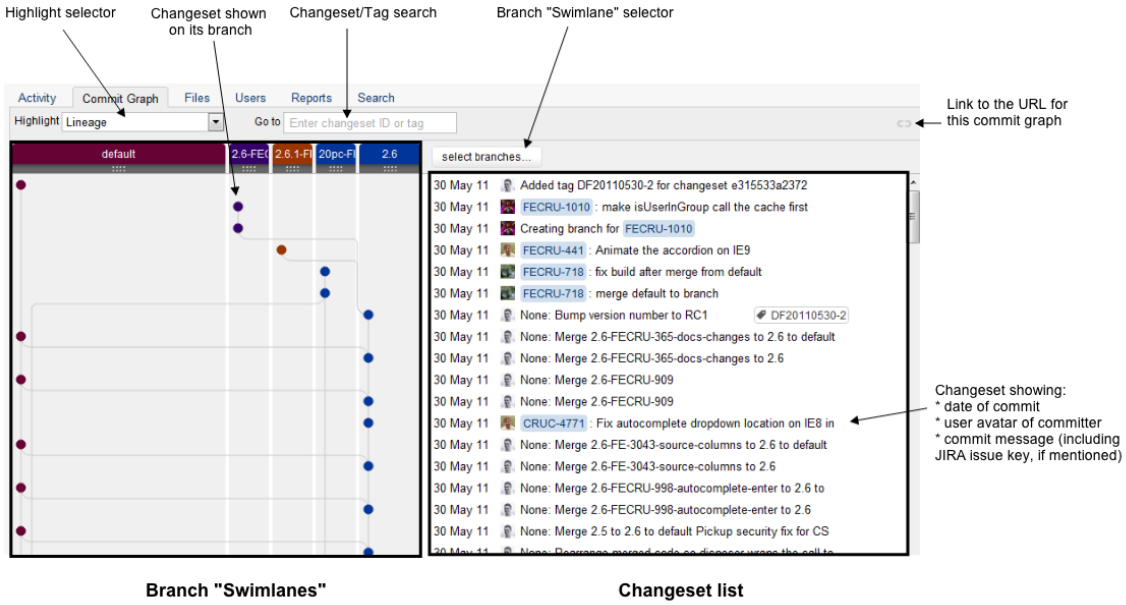
Viewing the commit graph for a repository

To view the commit graph for a repository:

1. Navigate to the desired repository, as described on Browsing through a repository.
2. Click the **Commit Graph** tab.

Selecting, or hovering on, a changeset (regardless of highlight) will display the following in the row for the changeset:

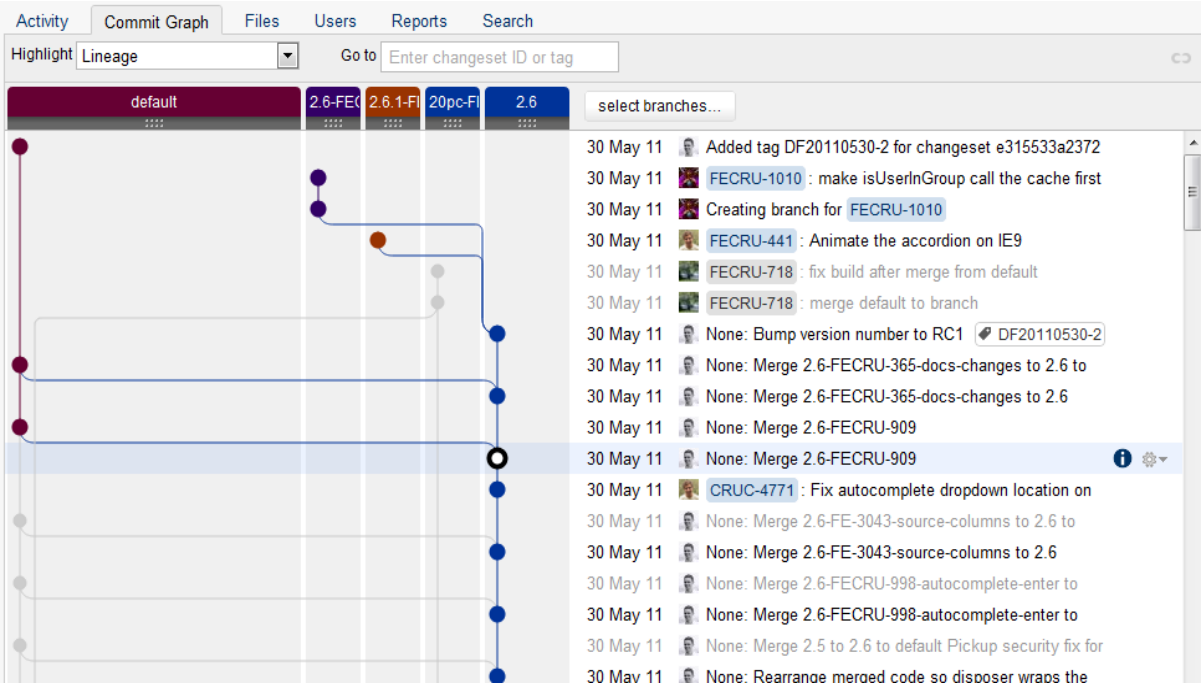
- an icon. Click this icon to see details for the changeset.
- a cog icon with a menu that allows you to see the changeset ID, view the full changeset, view the changeset in the activity stream, or to create a review for the changeset.



Highlighting the lineage of a changeset

Choose **Highlight** > **Lineage** to show the ancestor and descendant changesets for a selected changeset.

Action	Behaviour
Click on a changeset in the changeset list	Highlights where a changeset comes from and where it propagates to, i.e. its ancestors and descendants.
Hover over a changeset in a swimlane	Displays the changeset number and all the branches that the changeset is referenced in. This will include branches that you may not have swimlanes displayed for.



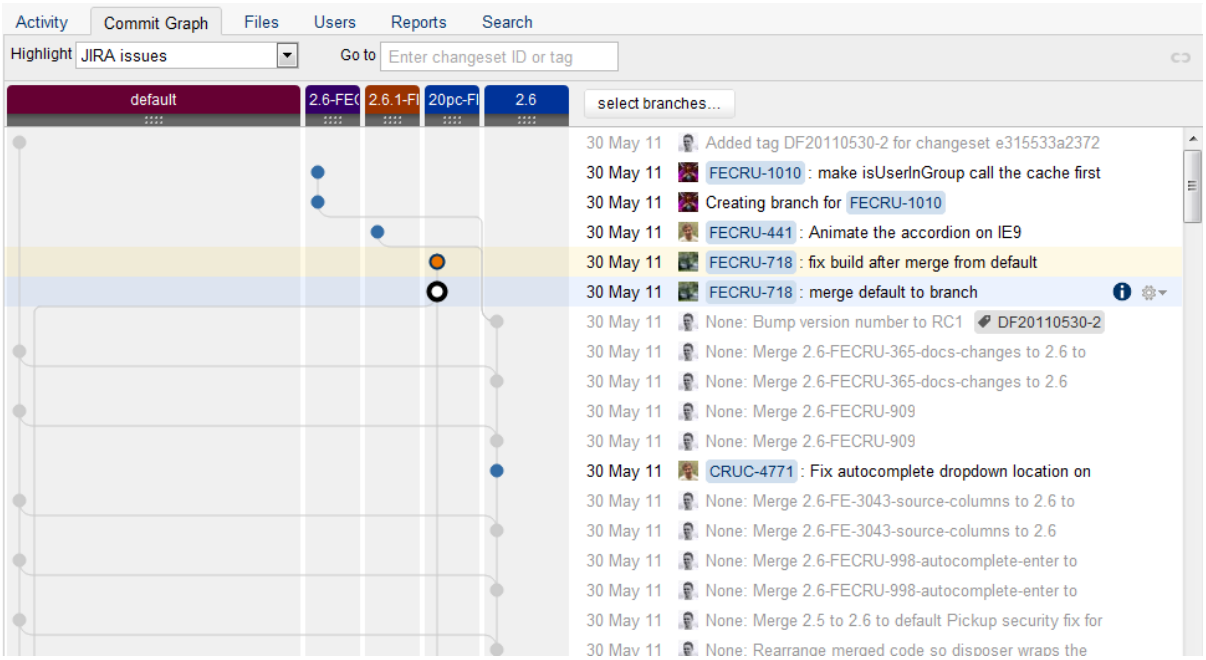
Highlighting JIRA issues

Choose **Highlight** > **JIRA issues** to highlight all the changesets that have a JIRA issue key in the commit

message.

This highlight type is only available if you have integrated FishEye with JIRA and linked your repository to a JIRA project.

Action	Behaviour
Click on a changeset in the changeset list	Highlights all other changesets that have the same JIRA issue key in the commit message.
Hover over a changeset in a swimlane	Displays all branches that the changeset is referenced in, and all referenced JIRA issues.



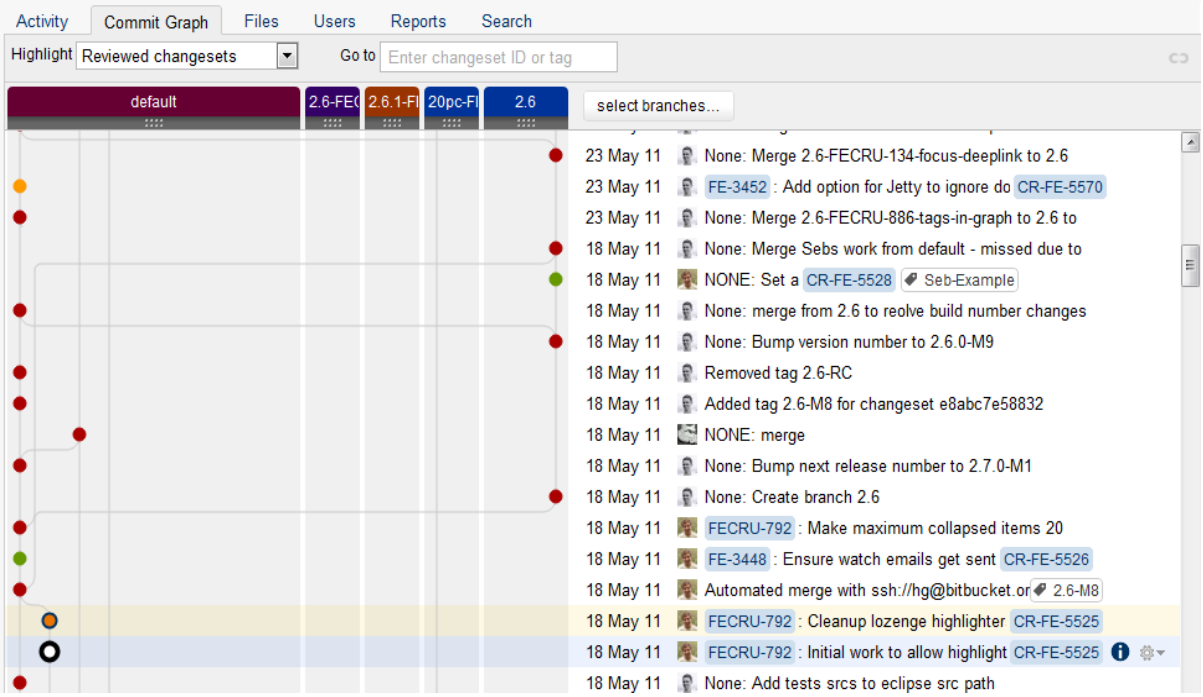
Highlighting reviewed changesets

Choose **Highlight** > **Reviewed changesets** to highlight the changesets that have been reviewed (i.e. included in a Crucible review):

- Red: unreviewed, i.e. the changeset is associated with a review in the 'Dead' or 'Rejected' state, or no review is associated.
- Yellow: under review, i.e. the changeset is associated with a review not in the 'Dead', 'Rejected' or 'Closed' state.
- Green: reviewed, i.e. the changeset is associated with a review in 'Closed' state.

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

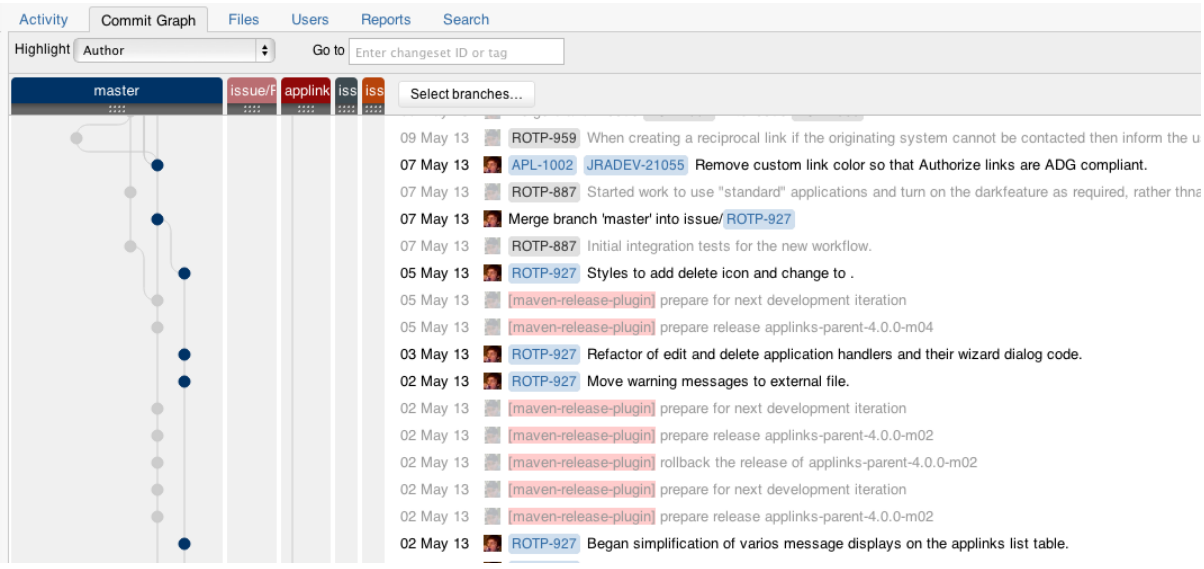
Action	Behaviour
Click on a changeset in the changeset list	Highlights the changesets that are part of the same review as the selected changeset.
Hover over a changeset in a swimlane	Displays all branches that the changeset is referenced in, and the Crucible review key.



Highlighting commits by an author

Choose **Highlight** > **Author** to highlight all the changesets submitted by a particular author.

Action	Behaviour
Click on a changeset in the changeset list	Highlights the changesets that were submitted by the same author.
Hover over a changeset in a swimlane	Displays the changeset number and all the branches that the changeset is referenced in.

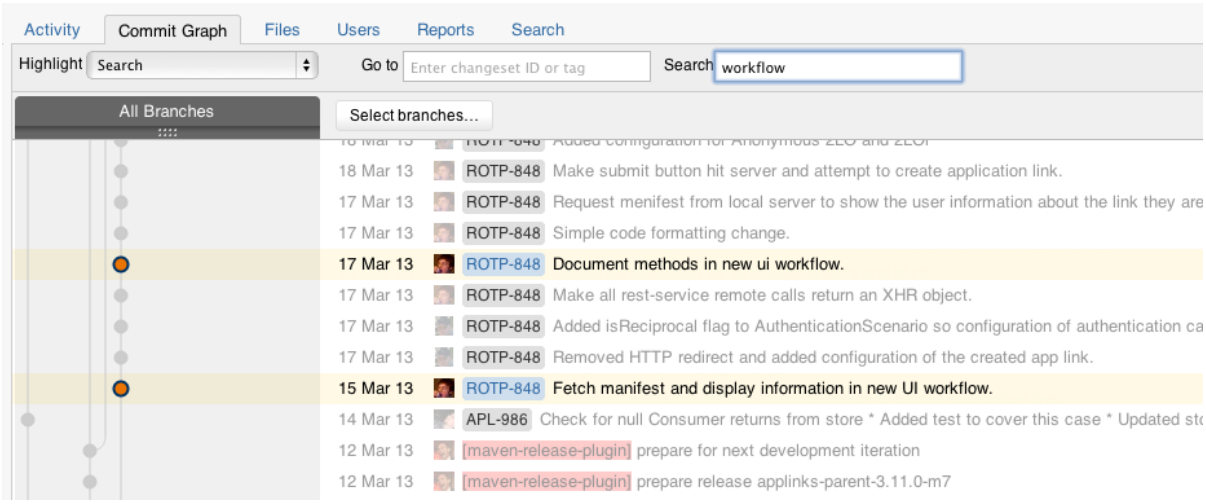


Highlighting search results

Choose **Highlight** > **Search** to highlight all the changesets where the commit message contains the search term.

Action	Behaviour
Click on a changeset in the changeset list	Highlights the changesets that match the search term.

Hover over a changeset in a swimlane	Displays the changeset number and all the branches that the changeset is referenced in.

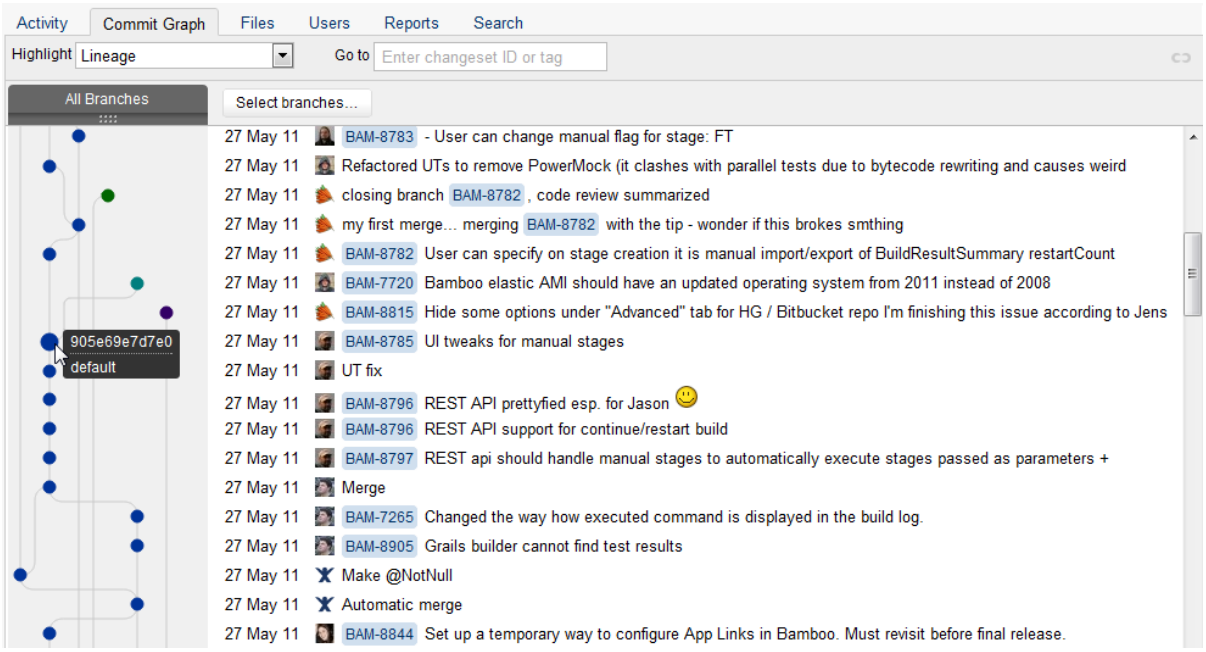


Viewing changesets across all branches

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 hover over any changeset to display information about the changeset, as described in the 'Highlighting Information in the Commit Graph' section above.

To see all the repository's branches in the commit graph:

1. Click **Select branches...** when viewing the commit graph.
2. In the 'Select Branches' dialog, click **Switch to 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 to determine 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 and, for each swimlane, 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

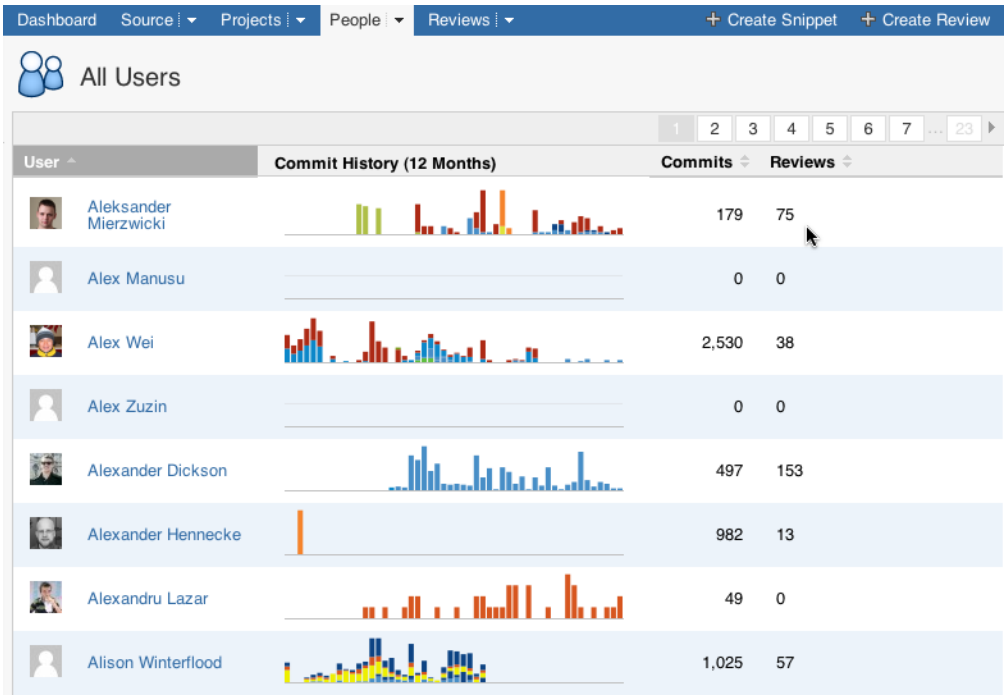
changesets shown in the 'fisheye-2.6' swimlane, as all the commits will be picked up in the 'master' swimlane. However, if you move the 'fisheye-2.6' swimlane to the left of the 'master' swimlane, it will pick up all of the FishEye 2.6 commits.

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.

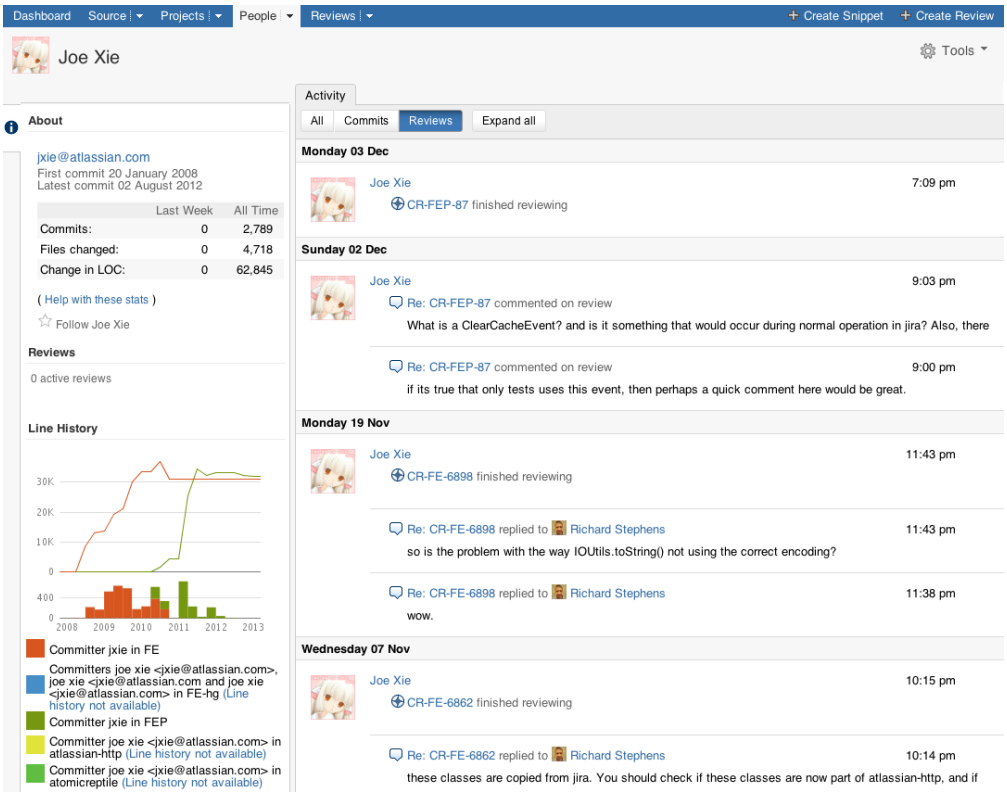
Screenshot: List of all commiters in FishEye



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



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.

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

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:

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
```

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:**

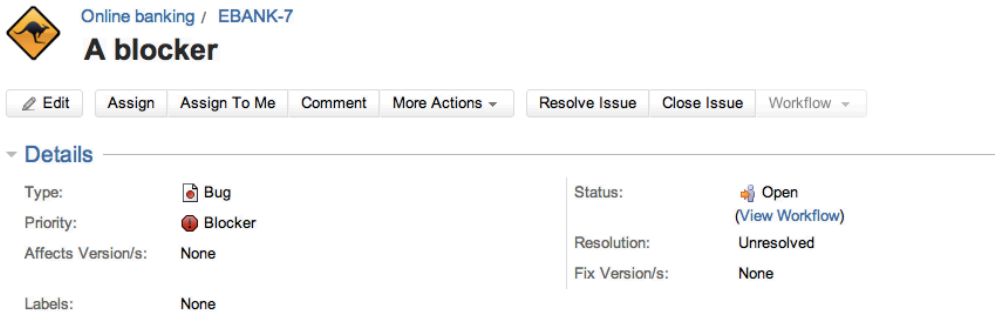
  <ISSUE_KEY1> <ISSUE_KEY2> <ISSUE_KEY3> #<COMMAND1> <optional COMMAND1_PARAMETERS> #<COMMAND2> <optional COMMAND2_PARAMETERS> #<COMMAND3> <optional COMMAND3_PARAMETERS> etc.

  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).



Command	Command Parameters	Description	Example

`#time`	**<n>w <n>d <n>h <n>m <work log comment>** *where **<n>** is a user-specified time period.*	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. *Please note:* • *Work log comments cannot be set using smart commits. See FE-3757.*	`#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.	`#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'. `#start` — this command would execute the 'Start Progress' workflow transition for an issue in the default JIRA workflow

		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. We are tracking this improvement request here:* ***FE-3873*** *- Smart Commits: Cannot set the resolution when using the "#resolve" command* OPEN	

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 in Crucible using a smart commit requires that the author of the changeset has already been mapped to a Crucible username. See 'Author mapping' on Changing your user profile.

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.

Review objectives

When creating a new review using a smart commit the default project objectives are added to the review, and since Fisheye/Crucible 2.10.2, the commit message is also copied to the review objectives.

Note that you cannot add arbitrary objectives to the review from the smart commit.

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 **Profile settings** from the User Menu (with your avatar) 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

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. This is disabled if avatars are served from an external server – see Configuring avatar settings.

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, changelogs 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.

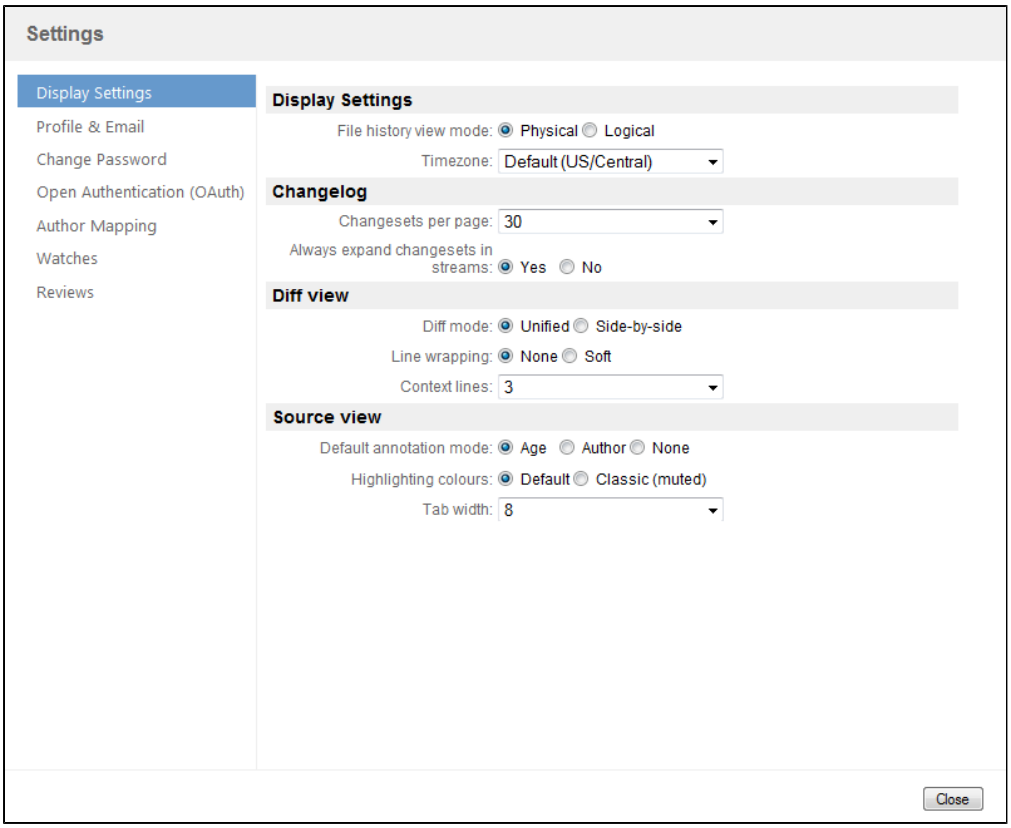
If the SMTP server is set up, then you will receive emails when different actions occur within Crucible.

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



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.

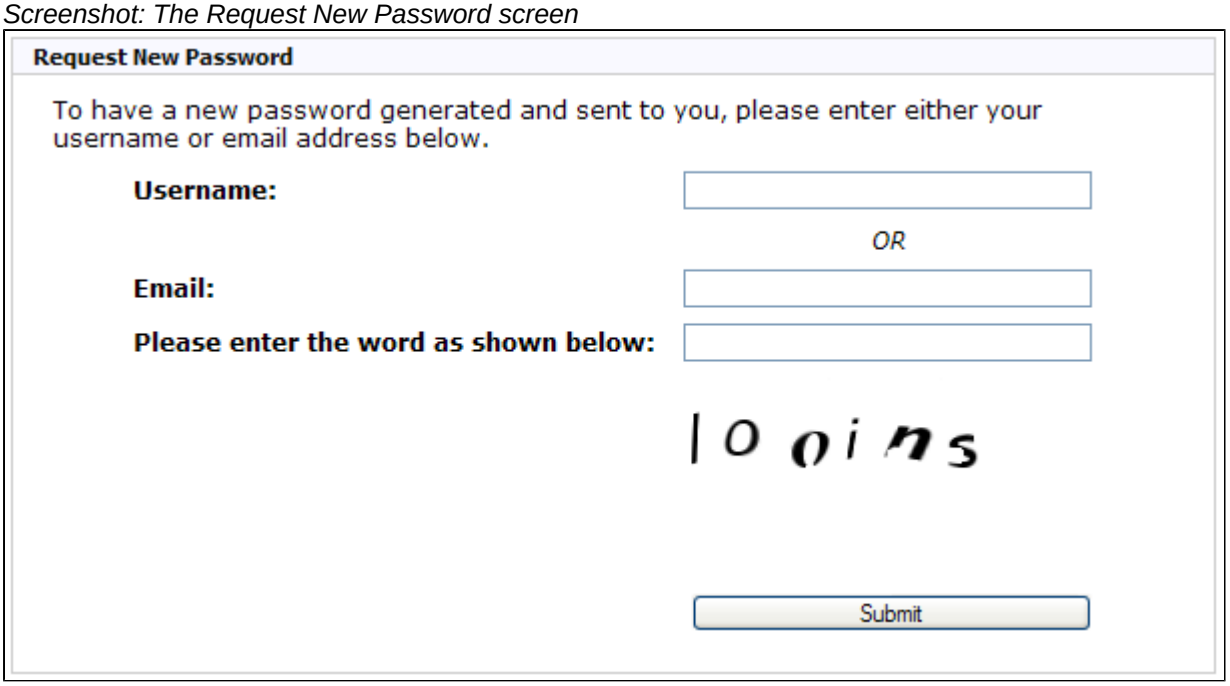
To reset your password:

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 **Plea se enter the word as shown below** and type the graphical letters shown above the **Submit** button.
3. An email is then sent to the email address specified in your profile. When it arrives, click the link supplied to complete the password reset.
4. On the resulting web page, you will receive the message 'A new password has been sent to your account.'
5. An email will arrive in your inbox, containing your new password.

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

Screenshot: The Log In dialog

Screenshot: The Request New Password screen



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 *path segments*.

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

*.txt	Matches `/foo.txt` and `/bar/foo.txt` but not `/foo.txty` or `/bar/foo.txty/`
/*.txt	Matches `/foo.txt` but not `/bar/foo.txt`
dir1/file.txt	Matches `/dir1/file.txt`, `/dir3/dir1/file.txt` and `/dir3/dir2/dir1/file.txt`
**/dir1/file.txt	Same as above.
/**/dir1/file.txt	Same as above.

/dir3/**/dir1/file.txt	Matches `/dir3/dir1/file.txt` and `/dir3/dir2/dir1/file.txt` but not `/dir3/file.txt`,`/dir1/file.txt`
/dir1/**	Matches all files under `/dir1/`
/dir1*	Matches all files as `/dir11`, `/dir12`, `/dir12345` a nd so on.
/dir??	Matches all files as `/dir11`, `/dir22` and so on, replacing just 2 characters.

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:

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

`DATECONSTANT` can be any of:

`now`	This very instant (at the time the expression was evaluated).
`today` `todaygmt`	The instant at 00:00:00 today. (server-time* or GMT)
`thisweek` `thisweekgmt`	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-field*s are **isBinary**, **isAdded**, **isDeleted**, **isCopied**, **isMoved**. e.g. **count(isAdded)** will return the number of added files.

*numeric-field*s 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 columns.
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:

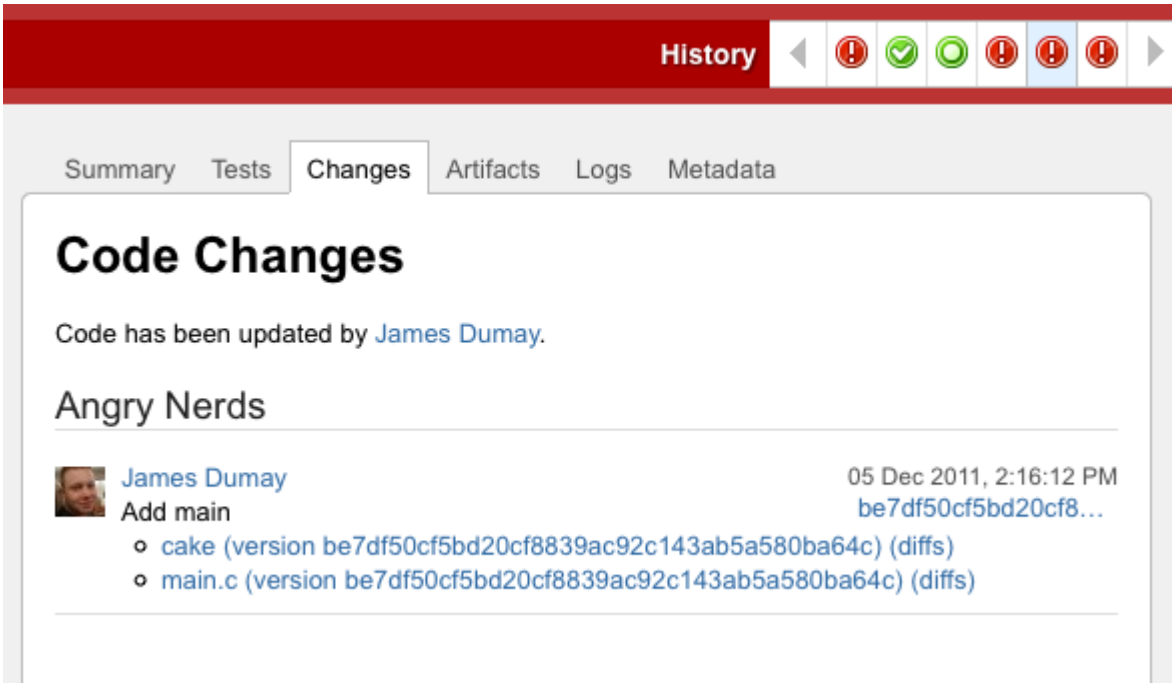
```
select revisions
from dir /
where not reviewed
group by changeset
return csid, author, count(revisions), comment
```

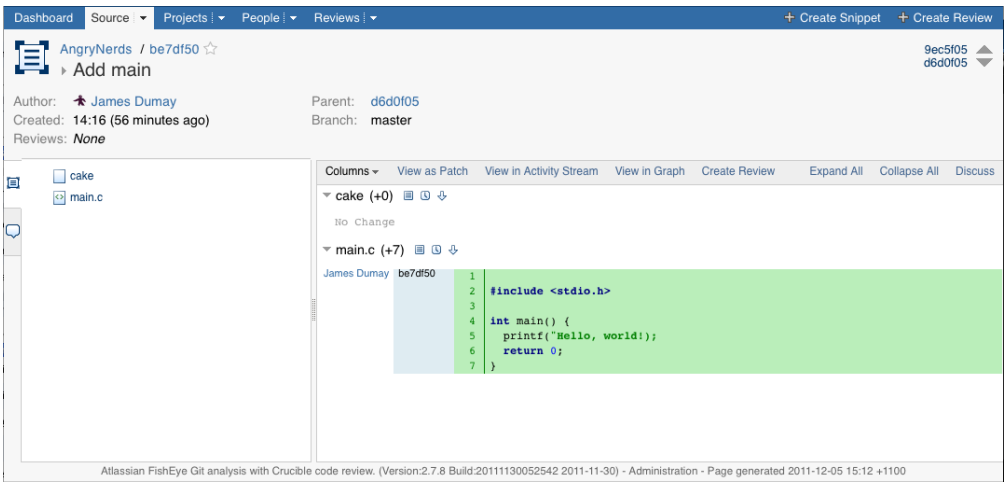
Integrating FishEye with Atlassian applications

You can integrate FishEye with the following Atlassian applications:

<table>
<tr>
<td>  </td>
<td>When FishEye is integrated with JIRA you can:
Use smart commits to transition JIRA issues
Delegate user and group management to JIRA
When Crucible is integrated with JIRA, you can:
Transition JIRA issues</td>
<td>When JIRA is integrated with FishEye, you can:
View an issue's FishEye changesets
Browse a project's FishEye changesets
Add the FishEye Charts Gadget to your JIRA dashboard
Add the FishEye Recent Changesets Gadget to your JIRA dashboard
View an issue's Crucible reviews
Browse a project's Crucible reviews
Add the Crucible Charts Gadget</td>
</tr>
<tr>
<td>  </td>
<td colspan="2">When FishEye is integrated with Crucible, you can:
In FishEye:
Use smart commits to create Crucible reviews, add reviews to new reviews and update an existing review.
In Crucible:
When using Iterative reviews in Crucible, you will be prompted when a new version of a file is available.
Files and changesets displayed in activity streams (e.g. the dashboard activity stream) render as links to the relevant files/changesets.
See your content roots and repositories associated with projects.
See repository lists and browse repositories using the Files tab.
View charts or code metrics.
See also What happens if I decide to stop using FishEye with Crucible.
You'll need the same number (or higher) of users in FishEye as Crucible.</td>
</tr>
<tr>
<td>  </td>
<td colspan="2">When FishEye is integrated with Stash (v2.11.4, and later) you can easily add Stash repositories to FishEye with a single click. Once added, the repository behaves just like a native repository in FishEye, so your team gets all the benefits of FishEye indexing, browsing and searching. Furthermore, the repository becomes available to Crucible, so you can perform in-depth code reviews for changes in the repository.</td>
</tr>
<tr>
<td>  </td>
<td colspan="2">See:
Integrating Crowd with Atlassian FishEye
Integrating Crowd with Atlassian Crucible</td>
</tr>
<tr>
<td>  </td>
<td colspan="2">When FishEye is integrated with your Bamboo continuous integration server, you can view the code changes that triggered a build. When a build fails due to a compilation error or failed test, you can explore the failed build in FishEye and jump directly into the changeset that broke the build. You can view the history of that changeset to see what the author was trying to fix, take advantage of the the side-by-side diff view to analyze the change and then open the correct files in your IDE.
For more details see Viewing the Code Changes that Triggered a Build.</td>
</tr>
</table>

Screenshots: Code changes listed in a Bamboo build (top), and viewed in FishEye (bottom)





JIRA Integration in FishEye

JIRA is Atlassian's issue tracking and project management application – when FishEye is integrated with JIRA, you and your team get all the benefits described on this page. Go straight to Linking FishEye to JIRA if you want to connect FishEye to a JIRA server.

You can also use JIRA for delegated management of your FishEye users. See JIRA and Crowd Authentication.

See Streamlining your development with JIRA for the full story of how Atlassian tools work together to give you a fast and guided software development process.

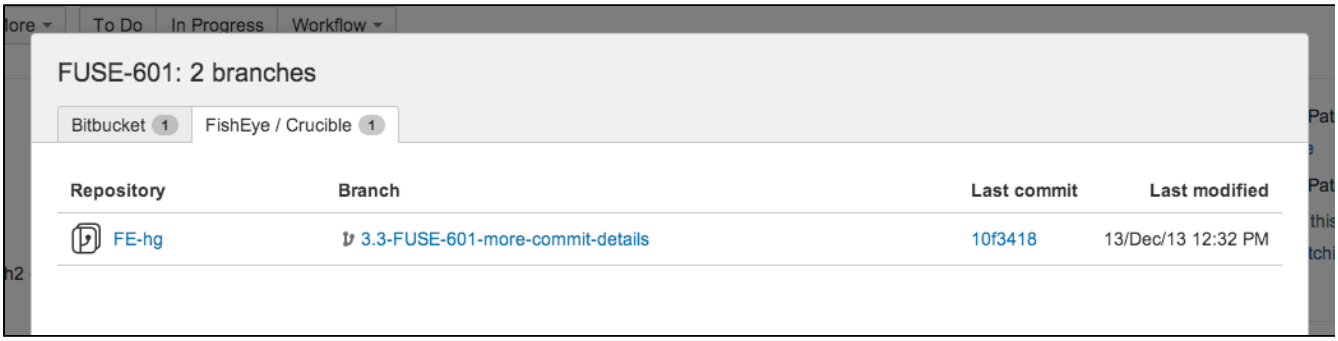
Related pages:

- Linking FishEye to JIRA
- Enabling Smart Commits

Your user tiers don't need to match between JIRA and FishEye/Crucible in order to integrate them. JIRA users that are not FishEye users will see the same view as FishEye users within JIRA, but will not be able to log in to FishEye to view the source/reviews.

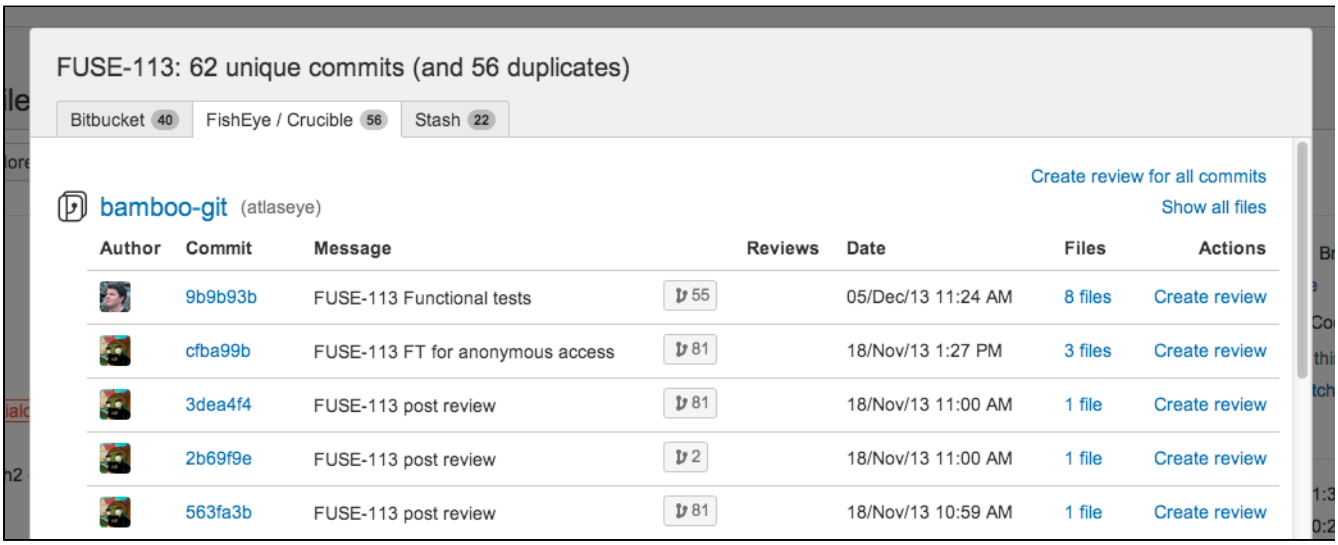
See in JIRA the FishEye repository branches related to an issue

For FishEye 3.3 and later, the FishEye repository branches related to a JIRA issue are summarized in the Development panel for the issue, when JIRA and FishEye are connected with an application link. To see details of the branches, simply click the **branches** link. You can see which repository each branch is in and when the last commit was made. As long as the issue key is included in the branch name the branch is automatically linked to the JIRA issue.



See in JIRA the commits related to an issue

For FishEye 3.3 and later, the FishEye repository commits related to a JIRA issue are summarized in the Development panel for the issue, when JIRA and FishEye are connected with an application link. You can click the **commits** link to see detailed information such as who made each commit, when they commited, and how many files were changed. Click through to see a particular commit in the FishEye instance where the commit was made. A developer only needs to add the issue key to the commit message for that commit to be automatically linked to the JIRA issue.



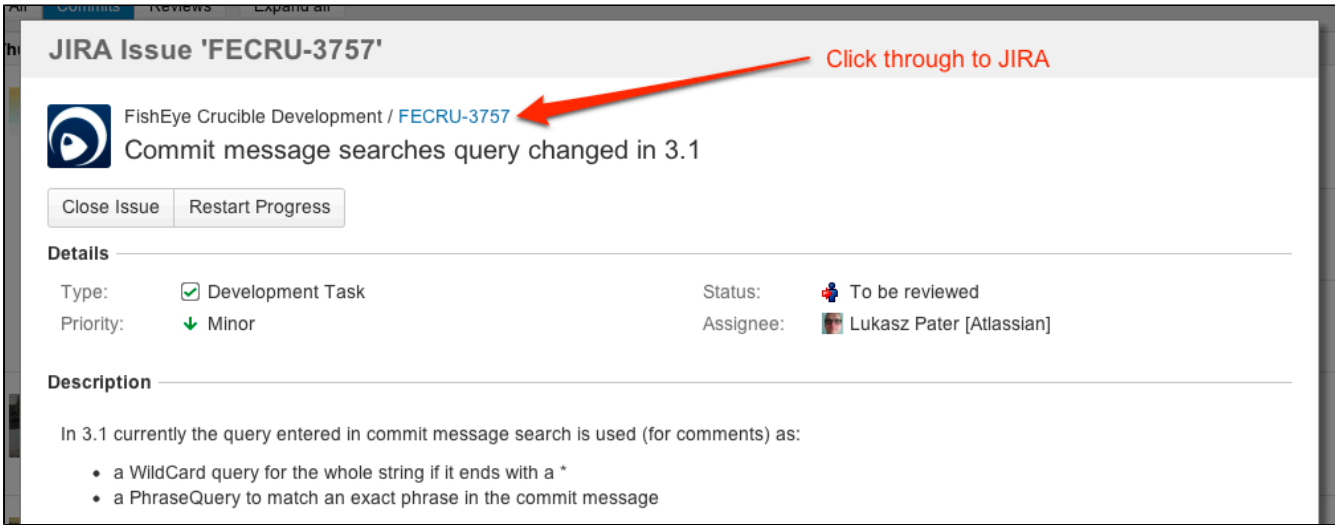
See the JIRA issues related to commits

FishEye recognises JIRA issue keys, and displays those as links in places such as the activity stream, side-by-side diffs, and commit messages:

Click on the linked issue key to see details for the issue, as described next.

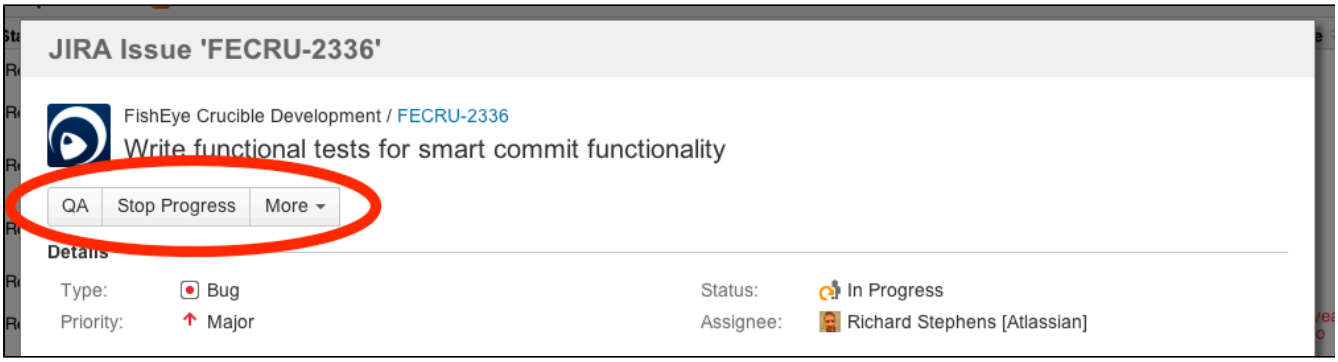
See the details for JIRA issues

Click a linked issue key anywhere in FishEye to see the details of that issue in a dialog. And you can click the issue key at the top of the dialog to go straight to the issue in JIRA:



Transition JIRA issues from within FishEye

You can easily transition a JIRA issue from within FishEye. For example, when viewing a commit, you may want to transition the related JIRA issue into QA. Click on a linked JIRA issue anywhere in FishEye to see a dialog with the available workflow steps:



Click on a step in the dialog, and complete any displayed fields as required. If there are custom required fields that are unsupported by FishEye, just click **Edit this field in JIRA** to transition the issue directly in JIRA.

See issues from multiple instances of JIRA

FishEye can link to more than one JIRA server at a time, so different teams can work with their own projects in different JIRA instances, or a single team can link to issues across multiple JIRA servers.

Integrating FishEye with Stash

This page...

... describes the benefits for you and your team, when FishEye is integrated with Stash.

Set it up...

... with our short guide to help FishEye admins connect FishEye to Stash.

Stash is...

... Atlassian's on-premises Git repository management solution for enterprise teams. Read about getting started with Git and Stash.

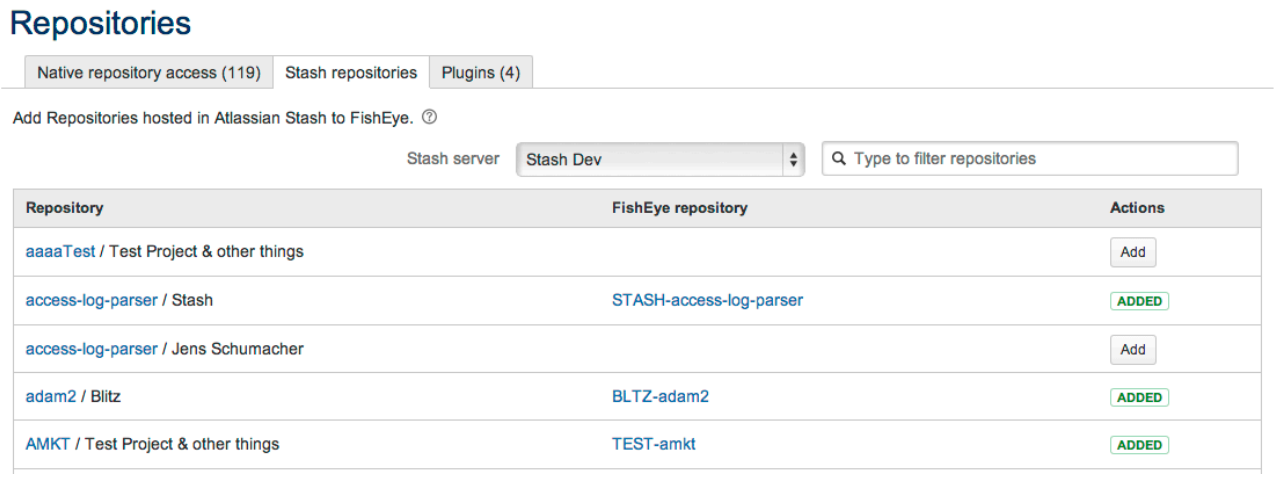
Easily add Stash repositories to FishEye

FISHEYE 3.4 +

STASH 2.11.4 +

When FishEye is integrated with Stash a FishEye administrator can easily add Stash repositories to FishEye with a single click. Once added, the repository behaves just like a native repository in FishEye, so your team gets all the benefits of FishEye indexing, browsing and searching. Furthermore, the repository becomes available to Crucible (when integrated), so you can perform in-depth code reviews for changes in the repository.

For detailed instructions for adding a Stash repository to FishEye, see Adding an external repository.



FishEye releases

FishEye 3.4

15 April 2014

- Importing Git repositories from Atlassian Stash
- Git indexing performance gains
- Administration REST APIs – `/repositories/`
- Header stalking behaviour

Read more in the FishEye 3.4 release notes and changelog.

See the FishEye upgrade guide.

FishEye 3.3

11 February 2014

- New branch/tag switcher
- Administration REST APIs
- Subversion 1.8 is now supported
- Microsoft Internet Explorer 11 is now supported

Read more in the FishEye 3.3 release notes and changelog.

See the FishEye upgrade guide.

FishEye 3.2

27 November 2013

- SCM checkout URLs
- Quick Search filters

- Internally managed Git repositories are no longer supported
- User data moved to the SQL database
- Improved protection against XSRF
- Microsoft SQL Server 2012 is now supported (support for SQL Server 2005 is deprecated)
- Microsoft Internet Explorer 10 is now supported (support for IE 8 is deprecated)
- MySQL 5.0 is deprecated
- PostgreSQL 8.2 is deprecated
- FishEye communication with JIRA versions older than 5.0 is no longer supported
- More in the release notes and changelog

See the FishEye upgrade guide.

FishEye 3.1

27 August 2013

- An improved dashboard
- QuickNav and QuickSearch improvements
- New JIRA issue dialog
- Transition JIRA issues from within FishEye
- Small improvements: native SVN 1.7, OpenJDK
- More in the release notes and changelog

See the FishEye upgrade guide.

FishEye 3.0

30 May 2013

- Optimised indexing for new SVN repositories
- Commit graph highlighter
- Redesigned UX
- Platform upgrades: Jetty 8, Infinity 3 DB
- Small improvements
- More in the release notes and changelog

See the FishEye upgrade guide.

FishEye 2.10

15 January 2013

- Repository indexing REST API
- Bitbucket and GitHub polling integration
- More in the release notes
- FishEye 2.10 upgrade guide

FishEye 2.9

14 November 2012

- Simpler JIRA integration
- More JIRA data in FishEye
- Faster JIRA source tab
- More in the release notes
- FishEye 2.9 upgrade guide

FishEye 2.8

15 August 2012

- Mentions
- Shares

- Improved performance of the activity stream
- Support for Subversion 1.7
- End of life announcements
- More in the release notes
- FishEye 2.8 upgrade guide

FishEye 2.7

7 September 2011

- Managed Git Repositories
- Smart Commits
- Web Hooks
- JIRA FishEye Plugin Improvements
- Small Improvements
- More in the release notes
- FishEye 2.7 Upgrade Guide

FishEye 2.6

6 June 2011

- Repository Commit Graph
- User Management via JIRA
- Improved Quick Search
- Redesigned HTML Emails
- Dashboard and Navigation Improvements
- Improved Support for Git Branches
- Git Commit Authors include Email Address
- Mercurial Indexing Improvements
- More in the release notes
- FishEye 2.6 Upgrade Guide

FishEye 2.5

8 February 2011

- Search Revamp
- Redesigned Activity Stream
- Mercurial and Git Authentication
- RSS Improvements
- Universal Plugin Manager
- More in the release notes
- FishEye 2.5 Upgrade Guide

FishEye older releases (click to expand)

FishEye 2.4

20 October 2010

- Branch and Tag Selector
- File History Redesigned
- Easier Application Linking
- SSL Support
- User Interface Improvements
- Performance Improvements
- More in the release notes
- FishEye 2.4 Upgrade Guide

FishEye 2.3

26 May 2010

- Mercurial SCM Alpha
- New 'Aggregate' functions in EyeQL query language
- Revamped Installation Process
- More in the release notes
- FishEye 2.3 Upgrade Guide

FishEye 2.2

18 Feb 2010

- Enhanced Side-by-Side Diff View Mode
- Improved Quick Navigation
- Annotation Context Menu
- Code Copying
- ClearCase and Git Support Now Final
- Numerous improvements and bug fixes
- More in the release notes
- FishEye 2.2 Upgrade Guide

FishEye 2.1

12 November 2009

- Wiki Markup in Commit Messages
- Streamlined JIRA Integration
- FishEye Admin API
- History Page Performance Increases
- ClearCase Support Now in Beta
- More in the release notes
- FishEye 2.1 Upgrade Guide

FishEye 2.0

30 June 2009

- Activity streams
- People statistics
- Favourites, bookmarks & saved search
- Enhanced JIRA integration
- New user interface
- Git beta
- More in the release notes
- FishEye 2.0 Upgrade Guide

FishEye 1.6

23 September 08

- FishEye search enhancements
- Multiple admin users
- Remote API improvements
- Changes to charts
- Perforce performance tweaks
- More in the release notes
- FishEye 1.6 Upgrade Guide

FishEye 1.5

14 April 2008

- Per-author lines of code statistics

- Charting improvements
- Customisable email templates
- More in the release notes
- FishEye 1.5 Upgrade Guide

FishEye 1.4

5 December 2007

- Enhancements to user management
- Crowd/SSO support
- Crucible integration
- Enhancements to JIRA plugin
- More in the release notes
- FishEye 1.4 Upgrade Guide

FishEye 1.3

1 August 2007

- Support for the Perforce version control system.
- SVN properties are now shown.
- Quicksearch now searches for changeset ids.
- New "mixed" chart on annotation pages, showing author-over-time breakdown.
- Side by Side diffs (1.3.1)
- More in the release notes
- FishEye 1.3 Upgrade Guide

FishEye 1.2.x Changelog

FishEye 1.1.x Changelog

FishEye 1.0.x Changelog

Security advisories (click to expand)

Security advisories

- FishEye and Crucible Security Advisory 2013-07-16
- FishEye and Crucible Security Advisory 2012-08-21
- FishEye and Crucible Security Advisory 2012-05-17
- FishEye and Crucible Security Advisory 2012-01-31
- FishEye and Crucible Security Advisory 2011-11-22
- FishEye and Crucible Security Advisory 2011-05-16
- FishEye and Crucible Security Advisory 2011-01-12
- FishEye Security Advisory 2010-10-20
- FishEye Security Advisory 2010-06-16
- FishEye Security Advisory 2010-05-04
- Fisheye and Crucible Security Advisory 2014-02-26

FishEye upgrade guide

This page describes how to upgrade to a new version of FishEye.

- For the latest FishEye release, see FishEye releases.
- For production environments we recommend that you test the FishEye upgrade on a QA server before deploying to production.
- The first time you run a new version of FishEye, it will automatically upgrade its data. This may involve a complete re-index of your repository.

On this page:

- Upgrading to FishEye 3.1
- Upgrading to FishEye 3.0
- Checking for known issues and troubleshooting the FishEye upgrade

Upgrade steps

This section provides general instructions for upgrading FishEye. See also the specific notes on this page for the version of FishEye you are upgrading to. We *strongly recommend* that you upgrade FishEye by following these steps:

- Back up your *entire* FishEye instance (see Backing up and restoring FishEye data), i.e.
  - If you are backing up your FishEye instance using the Admin interface, tick all of the 'Include' checkboxes (e.g. repository and application caches, plugins and their configuration data, SQL database, etc).
  - If you are backing up your FishEye instance using the command-line interface, do not use any exclusion options.
- Read the release notes and changelogs and the version-specific upgrade guide for the version you are upgrading to, as well as any versions you are skipping.
- Check the Supported platforms to ensure that your system meets the requirements for the new version.
- Check for known issues in the FishEye Knowledge Base.
- If you are using MySQL, read about the JDBC driver.
- Read about how to reduce downtime when a repository re-index is required.
- Once the new vesion of FishEye is installed, remember to update any custom configurations in the new version of FishEye, for example your SQL driver and your wrapper.config file.

Your upgrade procedure depends on whether you are using a `FISHEYE_INST` directory (i.e. "FishEye instance" directory).

- The `FISHEYE_INST` directory is the FishEye data directory (*not* the installation directory) and has a location defined by the `FISHEYE_INST` environment variable. It is used to keep the FishEye data completely separate from the FishEye/Crucible application files. We recommend that you configure FishEye/Crucible to use a `FISHEYE_INST` directory for production instances. Read more about `FISHEYE_INST` in Installing FishEye on Windows or Installing FishEye on Linux and Mac.
- The `<FishEye home directory>` is the location of the FishEye/Crucible application files.

NOTE: For all methods below, if you previously ran FishEye as a Windows service and are installing the new version of FishEye in a new location, you need to reinstall FishEye as a Windows service. Make sure to run ...\wrapper\bin\Fisheye-Uninstall-NTService.bat from the old location. Subsequently, prior to starting the new instance of FishEye, follow these instructions to set up Fisheye as a Windows service again. Make sure to copy over the changes from the old ...\wrapper\conf\wrapper.conf file to the new wrapper.conf file.

Method 1: Using a FISHEYE_INST directory

Click here to expand...

If you have FishEye/Crucible configured to use a `FISHEYE_INST` directory, then follow the instructions below. This is the recommended scenario for production installations.

1. Shut down your existing FishEye/Crucible server, using `bin\stop.bat` or `bin\stop.sh` from the `<FishEye home directory>`.
2. Make a backup of your `FISHEYE_INST` directory.
3. Download FishEye or Crucible.
4. Extract the new FishEye/Crucible version to a new directory.
5. Leave your `FISHEYE_INST` environment variable set to its existing location. Both FishEye and Crucible use this variable.
   - Please be aware that jar files in the `FISHEYE_INST/lib` directory may conflict with those required for FishEye's normal operation. Jar files in this directory should be limited to those which provide functionality not provided by FishEye (e.g. database drivers).
6. Start FishEye/Crucible from the new installation directory by running `bin/run.sh`. (Use `run.bat` on Windows.)
7. Follow any version-specific instructions found in the FishEye upgrade guide or Crucible upgrade guide .

Method 2: Without a FISHEYE_INST directory

Click here to expand...

If you do not have FishEye/Crucible configured to use a `FISHEYE_INST` directory and do not want to set one up, then follow the instructions below. The `<FishEye home directory>` is the location of the existing FishEye/Crucible installation. Note that this is the typical scenario for evaluation installations, and is not recommended for production installations.

You will need to copy some files from your old FishEye/Crucible installation to your new one.

1. Download FishEye or Crucible.
2. Extract the new FishEye/Crucible archive into a directory such as `<New FishEye home directory>`.
3. Shut down the old FishEye/Crucible server, using `bin\stop.bat` or `bin\stop.sh` from the `<FishEye home directory>`.
4. Copy `<FishEye home directory>/config.xml` to `<New FishEye home directory>`.
5. Delete the following directories from the `<New FishEye home directory>/var` directory:
   - `<New FishEye home directory>/var/cache`
   - `<New FishEye home directory>/var/data`
   - `<New FishEye home directory>/var/log`
6. Copy (or move) the following directories from `<FishEye home directory>/var` to `<New FishEye home directory>/var`:
   - `<FishEye home directory>/var/cache`
   - `<FishEye home directory>/var/data`
   - `<FishEye home directory>/var/log`

   DO NOT include the following directories when you do that:
   - `<FishEye home directory>/var/osgi-cache`
   - `<FishEye home directory>/var/plugins`
   - `<FishEye home directory>/var/tmp`
7. Delete the `<New FishEye home directory>/cache` directory.
8. Copy (or move) the `<FishEye home directory>/cache` directory to `<New FishEye home directory>/cache`.
9. Start FishEye/Crucible from the new installation by running `<New FishEye home directory>/bin/run.sh`. (Use `run.bat` on Windows.)
10. Follow any version-specific instructions found in the FishEye upgrade guide or Crucible upgrade guide.

Method 3: Without a FISHEYE_INST directory, but would like to set one up

Click here to expand...

If you do not have FishEye/Crucible configured to use a `FISHEYE_INST` directory but would like to set one up, then follow the instructions below. You may wish to do this when reconfiguring an existing installation for a production environment.

The `FISHEYE_INST` directory is the FishEye data directory, which has a location defined by the `FISHEYE_INST` environment variable, and which should be completely separate from the `<FishEye home directory>`. The `<FishEye home directory>` is the location of the existing FishEye/Crucible installation.

1. Download FishEye or Crucible.
2. Shut down the existing FishEye/Crucible server, using `bin\stop.bat` or `bin\stop.sh` from the `<FishEye home directory>`.
3. Set up the `FISHEYE_INST` environment variable, then create the `FISHEYE_INST` directory on your file system.
4. Copy `<FishEye home directory>/config.xml` to the `FISHEYE_INST` directory.
5. Copy the `<FishEye home directory>/var` directory to the `FISHEYE_INST` directory.
6. Copy the `<FishEye home directory>/cache` directory to the `FISHEYE_INST` directory.
7. If it exists, copy the `<FishEye home directory>/data` directory to the `FISHEYE_INST` directory.
8. Extract the new FishEye/Crucible archive into a directory such as `<New FishEye home directory>`.
9. Start FishEye/Crucible from the new installation by running `<New FishEye home directory>/bin/run.sh`. (Use `run.bat` on Windows.)
   - If your configuration is not automatically picked up and you cannot see your existing repositories, check your **Administration > Sys-Info** page, where you will see information about

the `<FishEye home directory>` and `FISHEYE_INST`. Check that your `FISHEYE_INST` is pointing to the right directory.

10. Follow any version-specific instructions found in the FishEye upgrade guide or Crucible upgrade guide.

Upgrading to FishEye 3.4

Please also see the Upgrade steps section above.

Please read the End of Support Announcements for FishEye. See Supported platforms.

Git manifest upgrade task

The Git manifest upgrade affects Git repositories. The rationale for the Git manifest change is described in Git manifest. There are two options that may be selected to control how the upgrade is performed. For medium to small sized Git repositories, we expect this upgrade process to be quite fast, in the order of minutes. For very large Git repositories, this could take up to a few hours.

Pre-indexing upgrade

By default, prior to indexing, each Git repository will be upgraded to add the manifest information for all changesets currently in the repository. Whilst this upgrade is in progress, the repository may be browsed normally and any existing reviews will be available for normal review workflow operations. New changesets will not be indexed, and will not be available for review until after the upgrade is complete.

Background upgrade

A background upgrade is performed during the normal repository indexing process. If there is time available within the polling interval, manifest upgrades are performed during the remaining time of the polling interval. The objective is that the next indexing poll should not be delayed unduly so new changesets continue to be indexed normally. The `fisheye.manifest.upgradebatch` system property is provided to control the minimum number of changesets that should be upgraded in each indexing poll. This is to ensure the background upgrade makes significant progress and may mean the indexing poll interval is longer than configured.

If repositories are configured to not poll, or they have a long polling period, FishEye will use the default Git polling period for the duration of the upgrade to ensure sufficient indexing occurs.

Whilst the upgrade is in progress, new changesets will be processed using the existing pre-3.4.0 approach, using Git ls-tree. Only once the upgrade is complete, will the new 3.4.0+ manifest approach be used. This means that the improved performance of the 3.4.0+ manifest upgrade will only be realized once the process is complete.

Choosing an upgrade approach

Selection of which approach to use is controlled by the FishEye `fisheye.manifest.forceupgrade` system property. The upgrade approach selected applies to the whole FishEye instance and affects all Git repositories in the instance. It is not possible to choose different upgrade approaches for different repositories.

The default setting of the property is to perform the upgrade prior to resuming normal indexing and this is the approach that we recommend. This realizes the benefits of the new manifest code as soon as possible but it does impact indexing of new changesets. To minimize the impact of such an upgrade, the upgrade could be undertaken during a low traffic period or the upgrade could be performed off-line on a separate server.

If it not feasible to have indexing of new changesets delayed at all, then the background upgrade approach can be used. The `fisheye.manifest.upgradebatch` system property can be tuned to reduce the amount of time spent upgrading to further reduce new changeset indexing impact.

It also possible to change from one approach to the other until the upgrade is complete. FishEye records the upgrade progress so that if FishEye is stopped during an upgrade, the upgrade will resume at the next opportunity. So, if you have started the upgrade using the pre-indexing approach, you can stop the FishEye server, change the system property, restart and the upgrade will continue using the background upgrade approach. Changing from background upgrade to pre-indexing upgrade is also supported.

v1 REST API resources deprecated

Note that the 'v1' REST API resources are deprecated and will be removed in a future release. See the FishEye Crucible REST API.

Known issues for FishEye 3.4

T	Key	Summary	Status
	FE-5133	Config.xml get/set/save is not synchronized	OPEN

Authenticate to retrieve your issues

1 issue

Upgrading to FishEye 3.3

Please also see the Upgrade steps section above.

As previously announced, the following platforms are no longer supported by FishEye 3.3:

- Internet Explorer 8
- MySQL 5.0
- PostgreSQL 8.2
- SQL Server 2005

Please read the End of Support Announcements for FishEye.

Supported platform upgrades

- SVN 1.8 is supported by FishEye 3.3.
- Microsoft Internet Explorer 11 is supported by FishEye 3.3.

See Supported platforms.

Known issues for FishEye 3.3

T	Key	Summary	Status
	FE-5133	Config.xml get/set/save is not synchronized	OPEN
	FE-5108	fisheye.changeset.paths.limit doesn't work as expected	OPEN
	FE-5107	Changeset comments appear on the activity stream even when nothing is starred to indicate them	OPEN
	FE-5102	Images (avatar) are not shown in email notifications	OPEN
	FE-5091	svn: Files added in revision 1 can have wrong added/removed linecounts	OPEN
	FE-5066	Username of existing user should not be displayed	OPEN

Authenticate to retrieve your issues

6 issues

Upgrading to FishEye 3.2

Please also see the Upgrade steps section above, and read the End of Support Announcements for FishEye page.

Please note the following changes in FishEye 3.2:

Internally managed Git repositories no longer supported

As previously announced, internally managed Git repositories are no longer supported by FishEye 3.2.

Please read the migration guide for information about options and procedures for migrating your internally managed Git repositories – note that we recommend that you upgrade to FishEye 3.2 before migrating any internally managed repositories.

Supported platform upgrades

- Communication with JIRA versions older than 5.0 is no longer supported.
- Microsoft SQL Server 2012 is now supported (support for SQL Server 2005 is deprecated).
- Microsoft Internet Explorer 10 is now supported (support for IE 8 is deprecated).
- MySQL 5.0 is deprecated.
- PostgreSQL 8.2 is deprecated.
- The Atlassian AUI plugin has been upgraded to AUI 5.2.
- jQuery has been upgraded to 1.8.3.
- backbonejs has been upgraded to 1.0.

See Supported platforms.

User data is moved from data0.bin to the SQL database

An upgrade task is run on startup that moves user data to the SQL database. We are doing this to mitigate the risk of data corruption or loss.

Known issues for FishEye 3.2

T	Key	Summary	Status
	FE-5133	Config.xml get/set/save is not synchronized	OPEN
	FE-5091	svn: Files added in revision 1 can have wrong added/removed linecounts	OPEN
	FE-5057	Git merge commits can show wrong diffs	OPEN
	FE-5014	RSS Feeds not working for Chrome	OPEN
	FE-4987	Mail configuration fields should be trimmed before saving changes	OPEN
	FE-4983	Failing migration from Crucible with FishEye v2.6.0 -> 3.1.5	OPEN
	FE-4930	CrowdAuth gets confused when non default crowd cookie token name is set up in crowd	OPEN
	FE-4912	Changing of avatar failed without any error message	OPEN
	FE-4886	500 error when adding repository to FishEye of conflicting name in different case	OPEN

Authenticate to retrieve your issues

9 issues

Upgrading to FishEye 3.1

Please also see the Upgrade steps section above, and read the End of Support Announcements for FishEye page.

Please note the following changes in FishEye 3.1:

Native SVN access via JavaHL requires JavaHL 1.7

You do **not** need to upgrade your subversion repositories to 1.7. SVN 1.6 is still supported.

If you are using native JavaHL to connect to your SVN repositories you may need to upgrade the SVN JavaHL client on your FishEye server. Please read Native support for SVN for more information.

If you are using SVNKit (the default) you do not need to upgrade SVN.

FishEye 3.1 Merge some per-repository Lucene indices into a global cross-repository Lucene index

FishEye 3.1 has greatly improved performance and scalability for QuickSearch and QuickNav. To achieve this, the per-repository 'METADATA' Lucene indices will be moved into a single global cross-repository Lucene index. This means FishEye is able to search across more repositories in less time because now only a single search index needs to be queried instead of the previous N. Merging these indices into the single cross-repository index can be refreshed in two ways:

1. **Recommended**: As an upgrade task that is run automatically when FishEye 3.1 is run for the first time.
2. As an offline process on a separate staging server.

During the automatic upgrade task, FishEye is fully usable and functional, although search results for files, commits and committers may be incomplete.

In our testing we have found that the automatic upgrade task took 4 hours to complete on a FishEye instance with 144 repositories of different kinds, with 58 GB of data in the FISHEYE_INST folder (excluding logs). We are confident that the automatic upgrade task is suitable for the majority of production FishEye installations. *It is worth repeating that the instance was fully functional (reviews, JIRA Integration, Activity Streams, Charts etc) apart from Quick Nav and Quick Search during this time.*

Nevertheless, where required, we provide instructions for performing the reindex as an offline process on a separate staging server.

Plugin Settings will be moved from the config.xml to the SQL database

As of FishEye 3.1.0, plugin settings which were previously stored in the <properties> element inside config.xml will be stored in the SQL database. This includes settings for any bundled plugins such as ApplicationLinks, the UniversalPluginManager etc, and any 3rd party plugins.

An upgrade task is run on startup which will first insert all the properties found in config.xml into a new table in the SQL database. Once successful, the properties are removed from config.xml.

As part of this change, the RepositoryOptions.setProperties (Map<String, String>properties) and RepositoryOptions.getProperties() methods have been removed from our API. If you are using a plugin which uses either of these methods, you will need to update the plugin to a version which uses the Spring component PluginSettingsFactory. Plugins can use this to access the migrated global and per-repository properties that were previously available via the old RepositoryOptions API.

Known issues for FishEye 3.1

T	Key	Summary	Status
	FE-5133	Config.xml get/set/save is not synchronized	OPEN
	FE-5091	svn: Files added in revision 1 can have wrong added/removed linecounts	OPEN
	FE-5057	Git merge commits can show wrong diffs	OPEN
	FE-4880	An empty git repository will never finish the 'Initial Index' phase	OPEN
	FE-4879	Problems authenticating against NTLM challenge with VisualSVN server	OPEN
	FE-4862	Commit chart rendering might affect performance	OPEN

Authenticate to retrieve your issues

6 issues

Upgrading to FishEye 3.0

Please also see the Upgrade steps section above.

Please note the following changes in FishEye 3.0:

Jetty 8

FishEye 3.0 now uses Jetty 8 as its web server and Java servelet container. This change should be completely transparent when you upgrade to FishEye 3.0. However, if you have customised either your `jetty-web.xml` file, or the `maxFormContentSize` system property, you will need to update those in the new version. See Enabling Access Logging in FishEye and this FishEye KB page for more information.

Infinity DB

FishEye 3.0 now uses the InfinityDB 3.0 database internally to provide improved performance for concurrent access to FIshEye. This change is transparent to users in all respects.

Pipelined indexing

FIshEye 3.0 introduces a new indexing approach that splits the repository indexing process into separate tasks that can be performed in a phased and concurrent way. Users will benefit from the way in which FishEye functionality, such as repository browsing, now becomes available as indexing progresses. This change is transparent to users in all other respects. See Pipelined indexing.

Improved handling of user preferences with session cookies

Upgrading may result in some users losing their preferences.

SQL Server transaction isolation configuration

We recommend a configuration change for SQL Server to use snapshot mode for the transaction isolation level – see Migrating to SQL Server. This change avoids occasional database deadlocks, and prevents performance warning messages in the FishEye logs and admin screens.

Known issues for FishEye 3.0

T	Key	Summary	Status
	FE-4657	Linecount calculator issues	OPEN
	FE-5133	Config.xml get/set/save is not synchronized	OPEN
	FE-4719	Some custom JIRA issue key patterns not linked in activity stream	OPEN
	FE-4674	Repository list empty until indexing completes and throws exception	OPEN
	FE-4506	Fisheye can break quite badly if you create a repository with a long name (longer than 100 chars) and then try to add a committer mapping	OPEN
	FE-5091	svn: Files added in revision 1 can have wrong added/removed linecounts	OPEN
	FE-5057	Git merge commits can show wrong diffs	OPEN
	FE-4880	An empty git repository will never finish the 'Initial Index' phase	OPEN
	FE-4879	Problems authenticating against NTLM challenge with VisualSVN server	OPEN
	FE-4862	Commit chart rendering might affect performance	OPEN
	FE-4841	Fisheye can get into a state where it is requesting files from SVN that do not exist (and will 404)	OPEN
	FE-4791	Could not find the attribute 'mail' in LDAP	OPEN
	FE-4779	Switching from simple to eyeQL search can create a broken eyeQL query	OPEN
	FE-4715	Git changelog out of order with git revert	OPEN
	FE-4676	Incorrect Tooltip Text	OPEN
	FE-4671	NullPointerException retrieving revisions for git repo	OPEN
	FE-4632	Commit Graph doesn't add branches when you search for a changeset not in the current set	OPEN
	FE-4345	"Polling Interval" Does Not Validate Allowed Values Properly	OPEN

Authenticate to retrieve your issues

18 issues

Checking for known issues and troubleshooting the FishEye upgrade

If something is not working correctly after you have completed the steps above to upgrade your FishEye installation, please check for known FishEye issues and try troubleshooting your upgrade as described below:

- **Check for known issues**. Sometimes we find out about a problem with the latest version of FishEye after we have released the software. In such cases we publish information about the known issues in the FishEye Knowledge Base. Please check the FishEye and Crucible Known Issues in the FishEye Knowledge Base and follow the instructions to apply any necessary patches if necessary.
- **Did you encounter a problem during the FishEye upgrade?** Please refer to the guide to troubleshooting upgrades in the FishEye Knowledge Base.
- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Git manifest

Introduction

FishEye and Crucible 3.4.0, and later versions, store information about the Git manifest. What is the manifest? The manifest for a commit is the list of the latest commit to affect each path in a repository. Maintaining this information speeds up a number of operations in FishEye indexing, which could, in some cases, be very slow in prevous FishEye/Crucible releases.The operations that are affected include new branch creation and tag application.

Git ls-tree

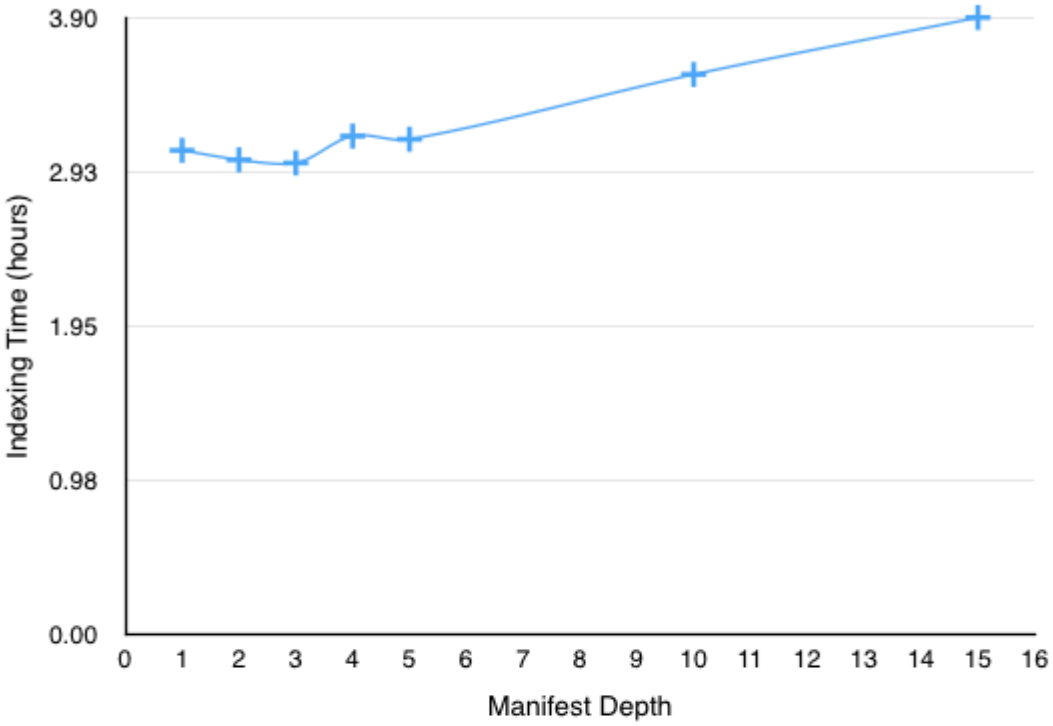
Git provides the ls-tree command to give the content manifest for a commit. For a commit, ls-tree gives the content hash for each path. It does not give any information about the commit that created that content. Prior to 3.4.0, FishEye used the ls-tree to fetch the content manifest and used this information to derive the commit manifest. In many cases there is a 1:1 relation between a content hash and the responsible commit at a given path in the repository. For those cases, the commit determination is very quick.

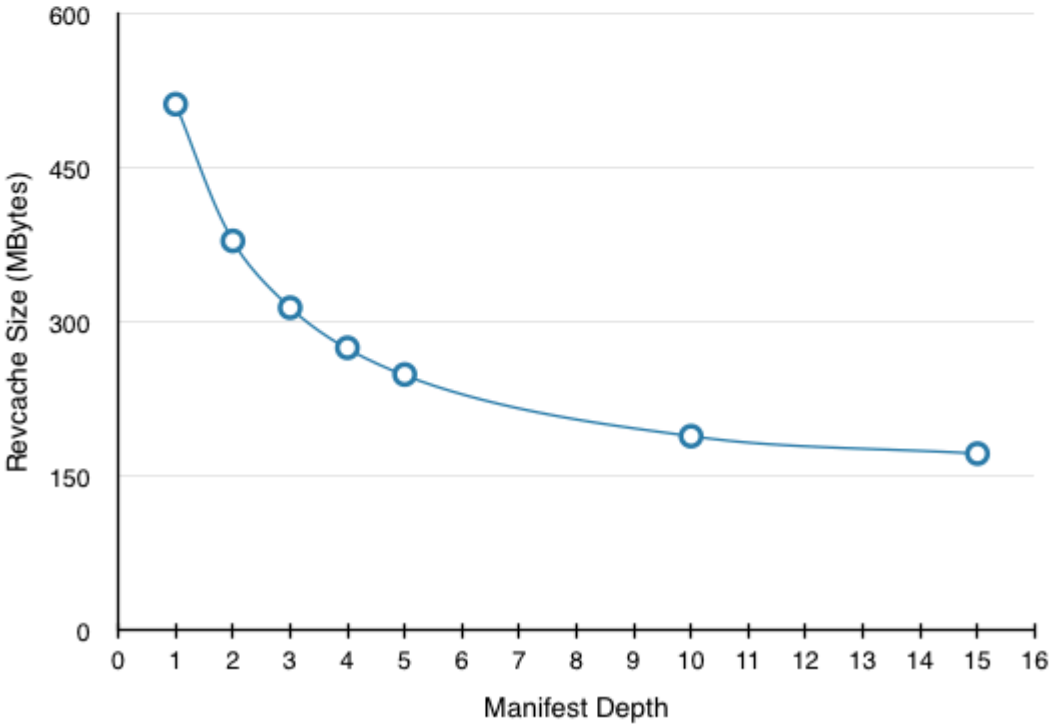
In other cases, however, the same content hash can be associated with multiple commits. In these cases FishEye would need to perform a history search to see which commit is the relevant commit. This search, performed per path, could become relatively slow. If you have seen long running ls-tree commands in previous versions of FishEye it is because these searches are occurring in the context of the ls-tree execution and not due to the time that ls-tree itself takes to execute.

Manifest storage

The solution in FishEye from version 3.4.0 is to avoid the execution of ls-tree in most situations, and to maintain the manifest for each commit in FishEye's index. Normally a single commit only changes a very small number of files in a repository. FishEye 3.4.0 uses a combination of full and delta manifests to store manifest information. Delta manifest entries only record the files that change in a commit, whereas full manifests record the full manifest. FishEye lets users control the maximum number of delta manifests that it writes before it writes a fresh full manifest. This lets users trade off the storage and performance characteristics of manifest storage using the FishEye system property *fisheye.manifest.maxdepth*. The default value for this property in FishEye is currently 3 as this gives a good balance between storage requirements and indexing performance.

An example repository showing the cache size and indexing times at various manifest depths, with FishEye 3.4, is shown below:





When FishEye 3.3 was used with this example repository the initial indexing time was 9 hours and the cache size was 155M. The speed up and space requirements will vary for each repository and depend on a number of characteristics of the repository including the number of active branches, the number of tags and the number of files in the repository

Changing manifest depth

The manifest depth can be changed by changing the FishEye system property and restarting FishEye. The new

manifest depth value only affects new manifests that are written after the FishEye restart. There is no need to reindex the repository as the manifests written with the old depth value continue to operate correctly.

Manifest upgrade

When deploying FishEye 3.4.0, and later, the manifest information needs to be created to allow the new manifest code to work correctly. There are various system properties which allow users to decide how the upgrade is performed. These are detailed in the Upgrade guide.

Migration guide for Git internally managed repositories

Internally managed Git repositories were deprecated in FishEye and Crucible 2.8, and support for these was removed for the FishEye and Crucible 3.2 releases. See End of Support Announcement for Internally Managed Repositories.

This page provides migration strategies and procedures for those who have internally managed Git repositories and wish to continue using those after upgrading to FishEye/Crucible 3.2.

What does this mean for users?

If you used managed repositories to share your code with others, you will no longer be able to push or pull changes from those repositories. Your Fisheye administrator will need to arrange access, or perhaps migrate those repositories and provide you with the new connection details so hat you can update the remotes in your local clones.

What does this mean for sysadmins?

As an administrator, you will need to decide whether you want to remove your internally managed repositories or convert them to standard repositories. When you upgrade to FishEye/Crucible 3.2, conversion will happen automatically and the repositories will be fully browsable and searchable, but any Git interaction with them through FishEye will be no longer possible. This means that users will not be able to share their code anymore, until you arrange that. There are a few options available for arranging Git access:

- Migrate the repository to different a Git hosting solution, for example Stash or Bitbucket.
- Migrate the repository to another server that can be accessed from FishEye, and by all users.
- Leave the repository in the same location.

For all these options, permissions will need to be arranged so that all the expected users can access the repositories and share their code. Detailed guides for those options are described below.

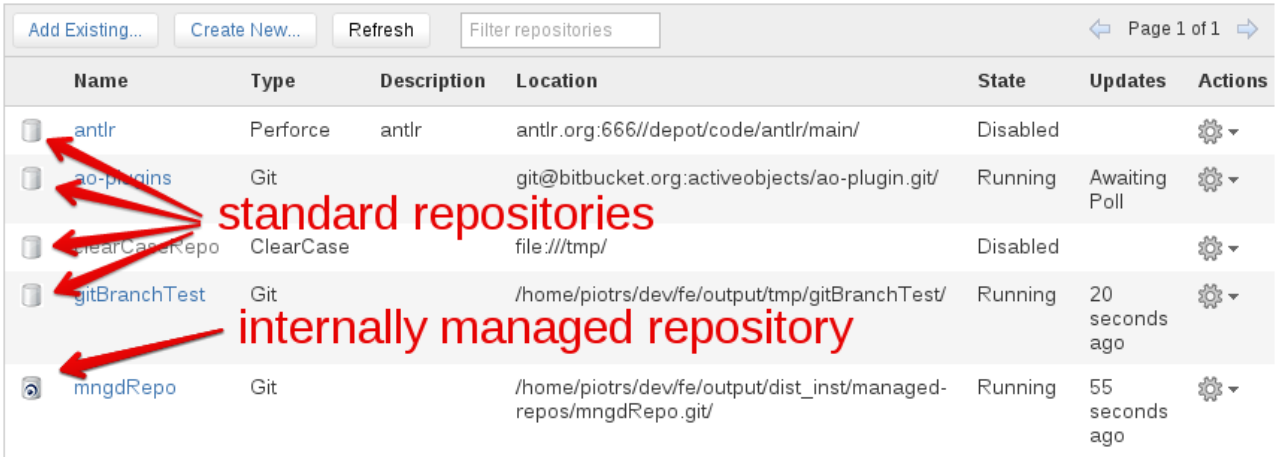
On this page:

Related pages:

- FishEye upgrade guide
- FishEye 3.2 release notes

How to check if you have managed repositories

If you using a FishEye/Crucible version prior to 3.2 you can check if you have managed repositories by going to the admin area. In the Repositories list, the icons at the left show whether the repository is internally managed or not:



Alternatively, you can use the REST service http://fecru-host:port/CONTEXT/rest-service-fe/managed-repositories-v1/git – it will list all managed repositories.

Preparing to upgrade to FishEye 3.2

Before upgrading, you should check if you have any internally managed Git repositories and decide whether you want to keep using those repositories or not. If not, you may simply delete them.

If you want to continue to use them, you may need to migrate them elsewhere and you will need to arrange access rights, so that users of those repositories will be able to share their code (that is, to push and pull to and from the repositories).

It is recommended that you upgrade your FishEye instance first, so as to have the repositories converted to standard ones, and to be able to change the repository location without the need to reindex it.

Upgrading to FishEye 3.2

Follow the standard FishEye upgrade guide and Crucible upgrade guide.

When FishEye/Crucible 3.2 is restarted and detects one or more managed repositories, it will stop the upgrade (before making any changes to your data) and will show the following warning in the log:
Click here to expand...

```
2013-11-18 16:12:10,211 WARN  -
**********************************************************************
**********************
2013-11-18 16:12:10,211 WARN  -
**********************************************************************
**********************
2013-11-18 16:12:10,211 WARN  -
**********************************************************************
**********************
2013-11-18 16:12:10,211 WARN  - *** Internally managed repositories are no
longer supported by FishEye and Crucible.
2013-11-18 16:12:10,212 WARN  - *** These were deprecated in FishEye 2.8:
2013-11-18 16:12:10,212 WARN  - *** https://confluence.atlassian.com/x/vQbSEQ
2013-11-18 16:12:10,212 WARN  - ***
2013-11-18 16:12:10,212 WARN  - ***   You can either:
2013-11-18 16:12:10,212 WARN  - ***     1. Start the old version of FishEye
and Crucible and remove all internally managed repositories,
2013-11-18 16:12:10,212 WARN  - ***        then continue with the upgrade
2013-11-18 16:12:10,213 WARN  - ***          or
2013-11-18 16:12:10,214 WARN  - ***     2. Start FishEye and Crucible with the
system property fisheye.managed.repos.convert set to true,
2013-11-18 16:12:10,214 WARN  - ***        so all the internally managed
repositories will be converted to standard repositories and
2013-11-18 16:12:10,214 WARN  - ***        any Git interaction with them
through the FishEye and Crucible will be disabled.
2013-11-18 16:12:10,214 WARN  - ***        You can set this by adding a java
system property to the FISHEYE_OPTS environment variable:
2013-11-18 16:12:10,214 WARN  - ***
-Dfisheye.managed.repos.convert=true
2013-11-18 16:12:10,214 WARN  - ***
2013-11-18 16:12:10,214 WARN  - ***   Please refer to
://confluence.atlassian.com/x/EQhcGQ for more information.
2013-11-18 16:12:10,214 WARN  - ***
2013-11-18 16:12:10,214 WARN  - *** You have the following Git internally
managed repositories in your configuration:
2013-11-18 16:12:10,214 WARN  - ***       * mngdRepo (disabled)
2013-11-18 16:12:10,215 WARN  - ***       * test-atl-streams
2013-11-18 16:12:10,215 WARN  -
**********************************************************************
**********************
2013-11-18 16:12:10,215 WARN  -
**********************************************************************
**********************
2013-11-18 16:12:10,215 WARN  -
**********************************************************************
**********************
```

If you are happy with having the repositories converted, add the *-Dfisheye.managed.repos.convert=true* optio n to the FISHEYE_OPTS environment variable, as explained in Command-Line Options.

If you run the upgrade now, with that system property defined, your repositories will be automatically converted:

Click here to expand...

```
2013-11-18 16:14:32,336 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,336 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,336 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,336 WARN  - *** Internally managed repositories are no
longer supported by FishEye and Crucible.
2013-11-18 16:14:32,336 WARN  - *** These were deprecated in FishEye 2.8:
2013-11-18 16:14:32,336 WARN  - *** https://confluence.atlassian.com/x/vQbSEQ
2013-11-18 16:14:32,336 WARN  - ***
2013-11-18 16:14:32,337 WARN  - ***   You have forced startup having Git
internally managed repositories defined.
2013-11-18 16:14:32,337 WARN  - ***   All of them will be converted to
standard repositories, but
2013-11-18 16:14:32,337 WARN  - ***   any Git interaction with them through
FishEye will be disabled.
2013-11-18 16:14:32,338 WARN  - ***
2013-11-18 16:14:32,338 WARN  - ***   Please refer to
https://confluence.atlassian.com/x/EQhcGQ for more information.
2013-11-18 16:14:32,338 WARN  - ***
2013-11-18 16:14:32,338 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,338 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,339 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,339 WARN  - ***   The following Git internally managed
repositories were converted to standard repositories:
2013-11-18 16:14:32,340 WARN  - ***       * mngdRepo (disabled)
2013-11-18 16:14:32,340 WARN  - ***       * test-atl-streams
2013-11-18 16:14:32,341 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,341 WARN  -
**********************************************************************
**********************
2013-11-18 16:14:32,341 WARN  -
**********************************************************************
**********************
```

The conversion task will:

- mark the repository as standard (rather than managed).
- set the polling period to default for managed repositories that had that disabled.

Conversion will not migrate repositories to an external Git hosting solution such as Stash. This has to be done manually and is explained below.

Migrating a managed repository from FishEye 3.2 (and later)

In FishEye/Crucible version 3.2 and later it is no longer possible to interact with managed repositories using Git. For developers to collaborate with others and push/pull their changes to a repository, the repository has to be made accessible to everybody. There are a few options available:

- Stash - Git Repository Management for Enterprise Teams

- Bitbucket - hosting site for the Git and Mercurial distributed version control systems (DVCS)
- and many others.

To migrate a repository:

1. Create a repository in the remote destination (for example, Stash).
    - For Bitbucket, choose **Create Repository** > **I have an existing project to push up**
2. Configure permissions for the repository in the remote destination so that all expected users have appropriate access.
3. Copy (or write down) the URL for the new repository (referred to here as DEST_URL).
    - Use the ssh: protocol instead of https: if you want to generate a key pair for authentication purposes. If you attempt to use https: with a key pair you will get the error: `You must not use http(s) with SSH key authentication, use ssh instead`
4. Run the following commends on the machine that hosts your FishEye instance:

```
# REPOSITORY - name of the repository
# It is assumed the repo is in the $FISHEYE_INST/managed-repos/ folder, but
this location might have been changed in Admin > Global Settings > Server >
Managed Repository Settings > Repositories Root.  If unsure, check the
Location attribute of your repository in Admin / Repositories
cd $FISHEYE_INST/managed-repos/REPOSITORY.git
git push --all DEST_URL
git push --tags DEST_URL
chmod -R a-w $FISHEYE_INST/managed-repos/REPOSITORY.git # don't allow
changes anymore; main repo is DEST_URL now.
```

5. In the FishEye admin area, click **Repositories** (under 'Repository Settings') and then the repository name and then **SCM Details**.
6. Update **Repository Location** to DEST_URL. (Be careful here, pointing to the wrong repository will result in inconsistencies in the FishEye/Crucible data – the repository will need to be fully reindexed to recover.)
7. Configure Git Authentication settings on the same 'SCM Details' page. Generate an SSH key pair and then:
    - For Stash, upload the public key as a user key (**Manage account** > **SSH keys** and click **Add key**) or as a repository access key.
    - For Bitbucket, upload the public key as a user key (**Profile** > **Manage Account** > **SSH Keys**) or as a repository Deployment Key.
8. Click **Test Connection** to validate access.
9. Click **Save**. You will be prompted to restart the repository. You may need to trigger an incremental scan manually if you have polling disabled.

Your repository will be recloned from the DEST_URL location (this may take a few minutes) but no reindex will be necessary. Once the reclone completes the repository should be ready for use in FishEye/Crucible.

You will need to notify users of the new location of the repository (for example, in Stash or Bitbucket) so that they can update remotes in their local clones.

Migrating a managed repository from FishEye 3.1 (and earlier)

This section describes how to migrate managed repositories from FishEye 3.1 to **Stash** or **Bitbucket**.

Note that we recommend that you first upgrade to FishEye 3.2 and then perform the migration, as described in the preceding section.

Click here to expand...

We will assume that you already have a new repository ready to use and that you have the latest local copy of the repository 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. Push all the branches to the remote repository

```
git push --all ssh://git@stash.mycompany.com/MYPROJECT/myrepo.git
```

4. Push all the tags to the remote repository

```
git push --tags ssh://git@stash.mycompany.com/MYPROJECT/myrepo.git
```

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.

5. 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.

6. Delete your old managed repository

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

Restoring a backup of FishEye 3.1 or earlier that contains managed repositories

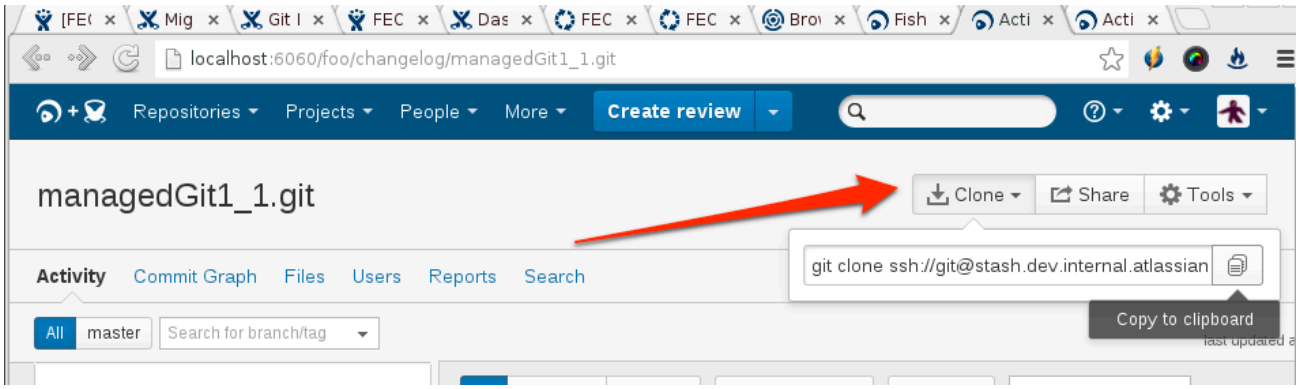
The FishEye backup/restore mechanism ignores internally managed Git repositories, so files under the `FISHEYE_INST/managed-repos/` folder had to be manually backed up and restored. Thus, if you attempt to restore a backup of FishEye 3.1 (or earlier) on FishEye 3.2 (or later), it will restore the backup keeping configuration of managed repositories in place. But on the first start of the restored instance, FishEye will check the existence of managed repositories and will issue a warning if any managed repositories are found. As explained in the section above, the system administrator may now use the *-Dfisheye.managed.repos.convert=true* system property to convert those repositories to standard ones so that they cn then be easily migrated elsewhere.

Guidance for developers who used to interact with managed repos using Git

You used to use Git to push or pull changes to a FishEye internally managed repository but now you get the following error:

```
$ git pull
fatal: repository 'http://localhost:6060/foo/git/managedGit1_1.git/' not found
```

This is because your repository has been migrated elsewhere and your local clone still points to its old location, on the FishEye instance. You need to find ithe new location – you can either ask your FishEye administrator, or simply find it in FishEye:



Once you know the repository location you can either update your local clone with the new location or purge your local clone and create a new one from the remote location. Note that the latter action will lose all your local changes. To preserve your local clone and simply update it with the new location, follow this procedure:

```
cd YOUR_LOCAL_CLONE
# check existing remotes
$ git remote -v
origin  http://localhost:6060/foo/git/managedGit1_1.git (fetch)
origin  http://localhost:6060/foo/git/managedGit1_1.git (push)
# in this case there is only one remote (origin).  It points to the old location
(my FishEye instance) so I am going to update it with the new location for the
migrated repository:
git remote set-url origin
ssh://git@stash.yourcompany.com:7999/test/managedGit1_1.git
# now I can pull from/push to that repository.
$ git pull
Already up-to-date.
```

Questions and Answers

Q: Will I lose my managed repository if I upgrade to FishEye 3.2?
A: No, FishEye/Crucible won't delete any data. The managed Git repository will stay intact in its original location.

Q: How long will it take to migrate a managed repository elsewhere?
A: The slowest step of the whole operation is cloning the SCM repository to the external location and then allowing FishEye/Crucible to reclone its local clone from the same repository location. So, it is mostly limited by the available network bandwidth. In our testing, a 61MB Git repository was migrated in approximately 90 seconds, over a fast network link to a Stash server 200ms apart.

Q: Will the upgrade process reindex all my repositories?
A: No. FishEye/Crucible will need to reclone the repository from the new location once the location changes, but no reindex will be necessary. After all, it is the same repository so all internal FishEye/Crucible caches can be easily reused.

Q: What kind of SCMs did internally managed repositories support?
A: Only Git was supported by internally managed repositories.

Q: What if I updated the repository location with a path to the wrong repository, during migration? (Imagine you have separate repos A and B, you migrated them to AStash and BStash, but then updated repo A with the BStash repository location by mistake).
A: Unfortunately such an operation will cause inconsistencies in the FishEye/Crucible data. Once this happens it can be recovered by either restoring this repository from a backup or requesting a full reindex on that repository.

FishEye 3.4 release notes

15 April 2014

Today we're excited to release FishEye 3.4. We've made huge improvements to FishEye's performance with

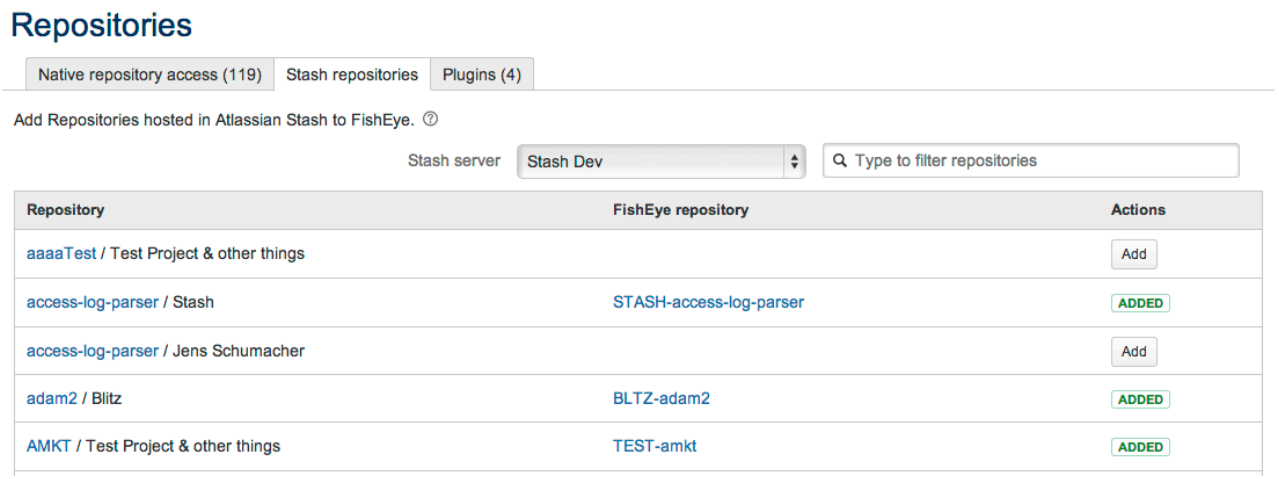
Git repositories, and made it ridiculously easy to add Git repositories from Stash. FishEye has your back covered, supporting your existing SVN repos and smoothing any future migration to Git.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide and the End of Support Announcements for FishEye.

The FishEye 3.4 changelog is at the bottom of this page.

Importing Git repositories from Atlassian Stash

When FishEye is integrated with Atlassian Stash a FishEye administrator can now easily add Stash repositories to FishEye – with just a single click. Once added, the repository behaves just like a native repository in FishEye, so your team gets all the benefits of FishEye indexing, browsing and searching. Furthermore, the repository becomes available to Crucible (when integrated), so you can perform in-depth code reviews for changes made to the Stash repository. *Read more...*



Git indexing performance gains

So that your experience with Git repositories is as good as possible, we've significantly improved the performance of the Git indexing operations in FishEye. You can read more about how this has been achieved.

The FishEye upgrade guide describes your options for how you run the upgrade task that is required.

Administration REST APIs – part 2

In FishEye 3.3, we added `/users/` and `/groups/` resources so you could programatically maintain users and groups, and manage group memberships.

Now, for FishEye 3.4, we've extended the FishEye REST APIs even further with the `/repositories/` resource to allow you to automate a range of administration tasks.

Note that the 'v1' REST API resources are deprecated and will be removed in a future release.

See the FishEye Crucible REST API.

Small improvements

Header behavior

You'll notice that the headers for changeset, source code and review listings now stalk – they stay visible when you scroll down. Just a small improvement to make life easier.

Change log

This section will contain information about the FishEye 3.4 minor releases as they become available. These releases will be free to all customers with active FishEye software maintenance.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide.

The issues listed below are just the highlights of all those that have been resolved for the FishEye 3.4.x releases.

15 April 2014 - FishEye 3.4.0

T	Key	Summary
	FE-4551	Make avatar cache TTL configurable
	FE-5129	Add LDAP timeouts and default to pooling SSL connections
	FE-5119	Watches of inactive users log errors
	FE-3461	Git repository indexing slowly
	FE-1533	Add option to disable line count
	FE-5150	REST capabilities urls returned with sitepath rather than request path
	FE-5126	OAuth application link requests might fail to be verified if AJP or Crowd SSO authentication is enabled
	FE-5118	Trying to Add Content to a Review which has a reviewer who was deleted, causes an NullPointerException.
	FE-5114	SVN Index Fails With "com.cenqua.fisheye.rep.DbException: Revkey not found for revid -1"
	FE-5095	QuickNav query not escaped when redirecting to quicksearch
	FE-5087	Mercurial repositories with no 'default' branch may be indexed in the wrong order
	FE-5076	Diff search result incorrect when a 'last modified' filter is selected
	FE-5019	Source view can fail to render with a javascript error because BaseRevisionCache.getBlameSpans does not always return all the blame spans for a given file
	FE-5004	DefaultContentManager can deadlock in database when reviewing content from a LightSCM repository
	FE-4900	Review Coverage Report has an error when hovering over committer in Individual Committer Stats panel
	FE-4768	Anonymised backups can't be restored into database
	FE-4764	FishEye present errors when using path with spaces on windows.
	FE-4698	Application Navigator uses Application URL instead of Display URL from Application Link

Authenticate to retrieve your issues

18 issues

FishEye 3.3 release notes

11 February 2014

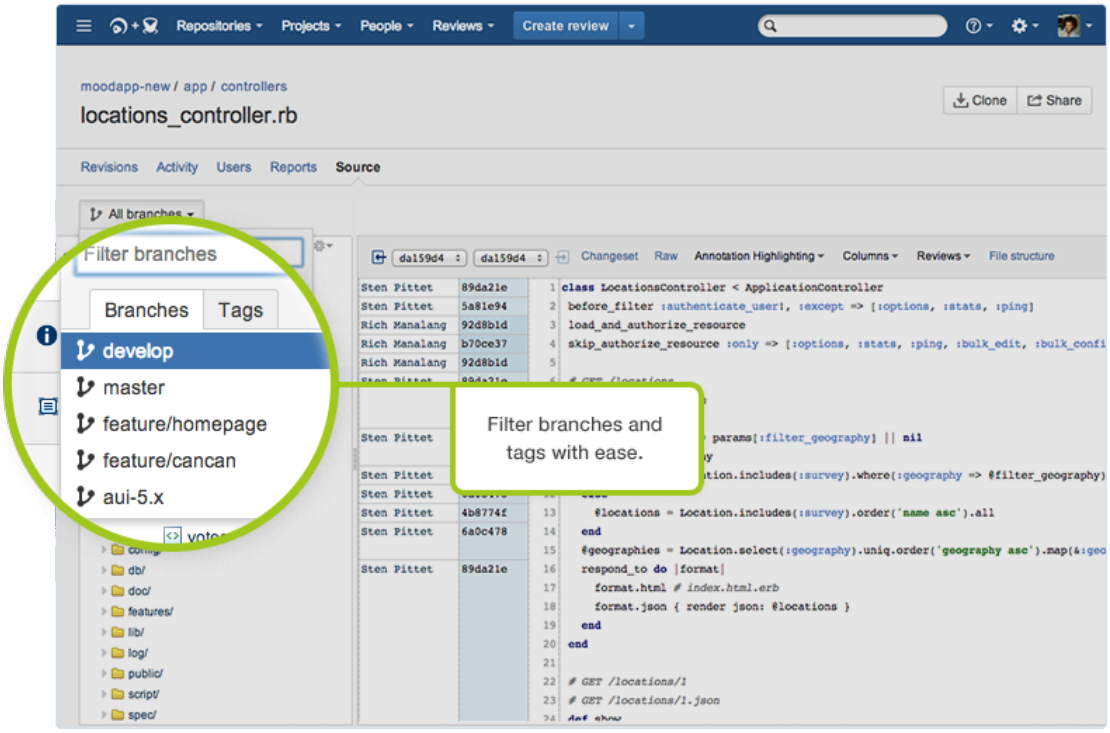
Today we're excited to release FishEye 3.3.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide and the End of Support Announcements for FishEye.

The FishEye 3.3 changelog is at the bottom of this page.

A new branch/tag switcher

As part of continuing to improve your experience of using FishEye, we've replaced the branch/tag switcher to make filtering branches and tags much easier. Just start typing to filter the braches, or tags, that are listed:



And it matches with the Atlassian Design Guidelines too!

Administration REST APIs

We're extending the FishEye REST APIs to allow you to automate a range of administration tasks. For FishEye 3.3, we've added /users/ and /groups/ resources – these allow you to programatically retrieve, update and delete both users and groups, and to add users to, or remove them from, groups.

See FishEye Crucible REST API.

Support for Subversion 1.8

FishEye now supports Subversion 1.8, so you can use FishEye to index your Subversion 1.8 repositories. See Native support for SVN.

Support for Internet Explorer 11

FishEye now supports Microsoft Internet Explorer 11.

See Supported platforms

Small improvements

Platform changes

- Support for IE 8 has now ended with the release of FishEye 3.3.

- Support for MySQL 5.0 has now ended with the release of FishEye 3.3.
- Support for PostgreSQL 8.2 has now ended with the release of FishEye 3.3.
- Support for SQL Server 2005 has now ended with the release of FishEye 3.3.

See Supported platforms.

Change log

This section will contain information about the FishEye 3.3 minor releases as they become available. These releases will be free to all customers with active FishEye software maintenance.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide.

The issues listed below are just the highlights of all those that have been resolved for the FishEye 3.3.x releases.

4 April 2014 - FishEye 3.3.3

T	Key	Summary
	FE-5126	OAuth application link requests might fail to be verified if AJP or Crowd SSO authentication is enabled
	FE-5087	Mercurial repositories with no 'default' branch may be indexed in the wrong order
	FE-5019	Source view can fail to render with a javascript error because BaseRevisionCache.getBlameSpans does not always return all the blame spans for a given file

Authenticate to retrieve your issues

3 issues

19 March 2014 - FishEye 3.3.2

T	Key	Summary
	FE-5114	SVN Index Fails With "com.cenqua.fisheye.rep.DbException: Revkey not found for revid -1"
	FE-5095	QuickNav query not escaped when redirecting to quicksearch
	FE-5004	DefaultContentManager can deadlock in database when reviewing content from a LightSCM repository
	FE-4722	Unable to index file path with double quotes
	FE-4698	Application Navigator uses Application URL instead of Display URL from Application Link

Authenticate to retrieve your issues

5 issues

21 February 2014 - FishEye 3.3.1

T	Key	Summary
	FE-5084	Unclear warning on private key file and folder permission
	FE-5085	cryptic error message when ssh binary not found on the path
	FE-5081	logged in users inherit anonymous user repo permissions no more if global anon access is off
	FE-5076	Diff search result incorrect when a 'last modified' filter is selected

Authenticate to retrieve your issues

4 issues

11 February 2014 - FishEye 3.3.0

T	Key	Summary
	FE-4709	Subversion 1.8 Support
	FE-4211	Provide REST endpoints to retrieve group membership data
	FE-5015	Increase SVN Operation Timeout from 60 minutes to 4 hours
	FE-4998	Display modified date, author and revision for directory listings in FishEye
	FE-4538	Put the New User Mapping Form Before List of User Mapping
	FE-5048	Git indexing takes a long time 'comitting changes'
	FE-5018	no progress indicator shown in quicksearch when counting other docs
	FE-5016	SVN: setting a start revision prompts a restart, not a re-index
	FE-5013	FECRU Developer Mode Plugin causes NPE when viewing a review
	FE-5000	User synchronisation fails if invalid username is found
	FE-4988	"View" and "Fisheye" menus appear behind text in IE10
	FE-4984	On Windows, Fisheye attempts to make ssh keys private but appears to be unsucessful
	FE-4974	Showing the UPM admin log can't be shown, if it contains entries made by the admin password user
	FE-4972	Git branch heads with non-ascii characters in commit message not parsed properly in some cases
	FE-4916	New pipeline index ignoring initial-import setting from svn configuration
	FE-4884	Line chart options jump back to "None" after "Apply"
	FE-4857	'E204899: Could not save file' error when trying to index svn repository
	FE-4531	Radio Buttons Reset When 'Apply' Is Pressed
	FE-4127	Reports UI - Cannot switch to Area chart from Line chart
	FE-4082	Change and Pie Charts Are Not Available

Authenticate to retrieve your issues

20 issues

FishEye 3.2 release notes

27 November 2013

Today we're excited to release FishEye 3.2, which brings a range of usability and platform improvements.

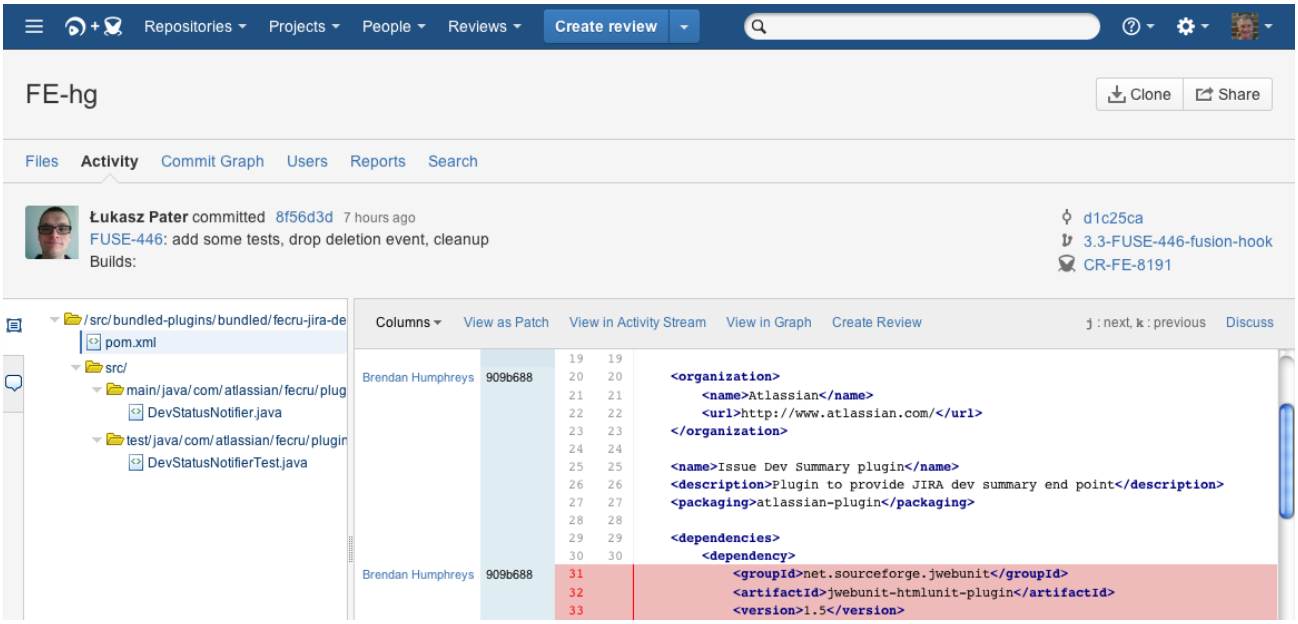
If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide and the End of Support Announcements for FishEye.

The FishEye 3.2 changelog is at the bottom of this page.

High design values with the ADG

We've been applying the Atlassian Design Guidelines to make using FishEye a better experience – the layout

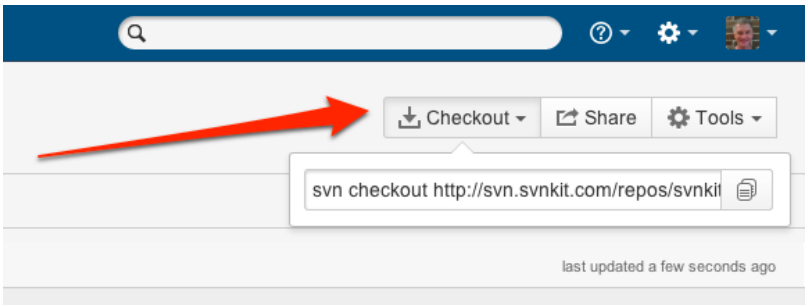
and behaviour have been improved, and if you use other Atlassian products then FishEye will feel very familiar from the moment you log in. In particular, for FishEye 3.2, the navigation menu and the headers for the Changeset and Source pages have been improved for better clarity. For example, the Changeset page now has a much cleaner layout:



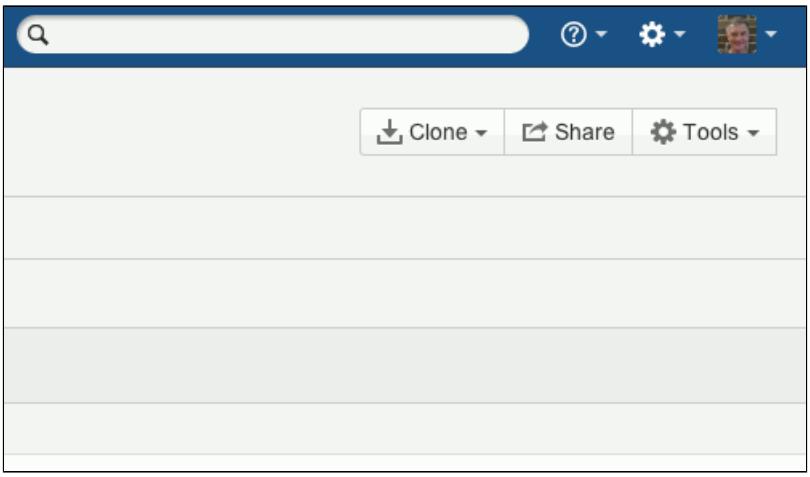
SCM checkout URLs

Sometimes when you're looking at source in FishEye you want to be able to access the code locally. We've made that much easier now by providing checkout or clone URLs at the top of the page for a repository.

For example, if you're using SVN, click **Checkout**, and copy the URL:



If you're using Git, click **Clone** and copy the URL:



If you're already using Atlassian SourceTree, simply choose **Clone in SourceTree**, to have SourceTree manage the clone operation for you.

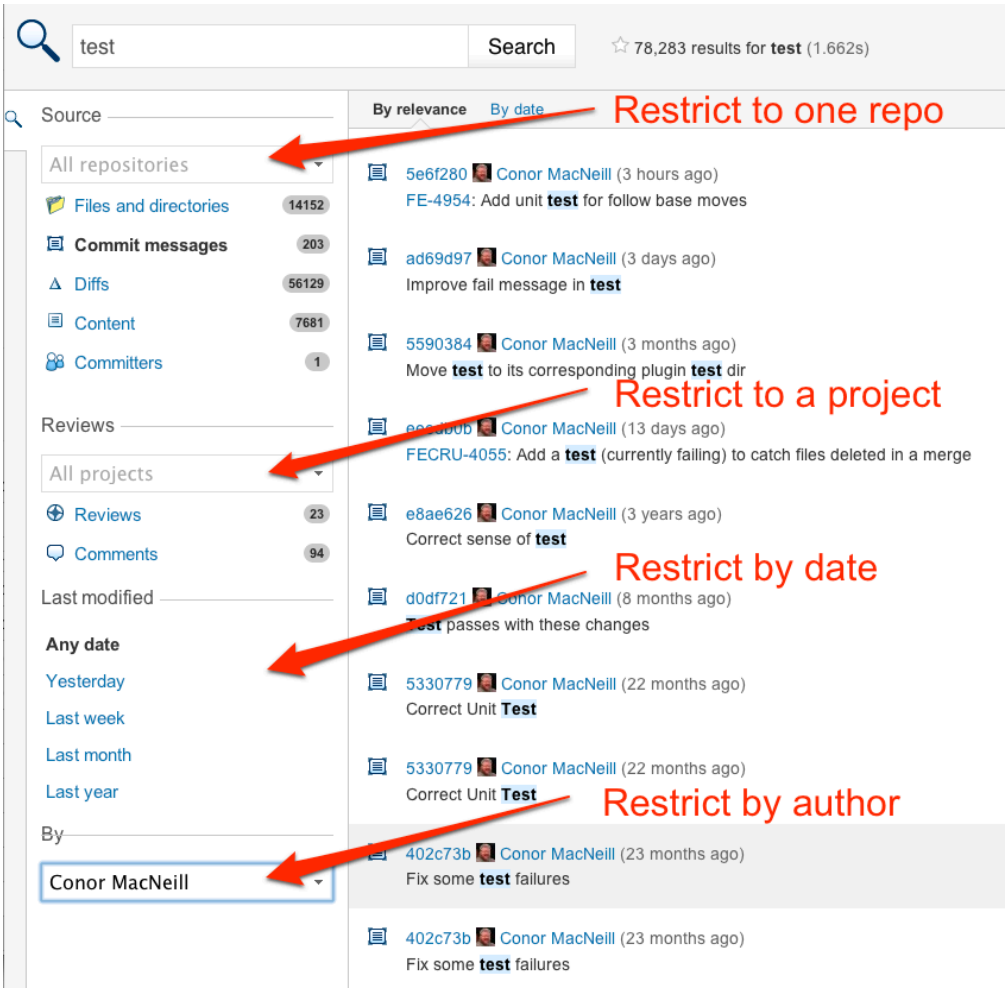
FishEye admins can control the display of the URL for the whole FishEye instance and for particular repositories. See Checkout URLs for details.

Quick Search filters

We've added filters to Quick Search, to make finding the content you want even quicker!

Now, in FishEye 3.2, you can also filter by the last modified date and by author. Sensibly, FishEye restricts some combinations of filters – for example, you can filter diff results by the last modified date, but not by author.

Of course, for a while now, you've been able to filter search results by different source types such as files and directories, commit messages, diffs, file content, and committers. And when used with Atlassian Crucible, you can filter by reviews and review comments. and restrict the search scope for reviews to a particular project.



Read more about searching in FishEye...

Internally managed Git repositories are no longer supported

As previously announced, internally managed Git repositories are no longer supported by FishEye 3.2.

Please read the migration guide for information about options and procedures for migrating your internally managed Git repositories.

FishEye communication to JIRA versions older than 5.0 is no longer supported

As previously announced , communication from FishEye to JIRA versions older than 5.0 is no longer supported by FishEye 3.2. Please note that communication from JIRA to FishEye will continue to work as it currently does.

Small improvements

Platform upgrades

- Microsoft SQL Server 2012 is now supported (support for SQL Server 2005 is deprecated).
- Microsoft Internet Explorer 10 is now supported (support for IE 8 is deprecated).
- MySQL 5.0 is deprecated.
- PostgreSQL 8.2 is deprecated.
- The Atlassian AUI plugin has been upgraded to AUI 5.2.
- jQuery has been upgraded to 1.8.3.
- backbonejs has been upgraded to 1.0.

See Supported platforms.

User data moved to the SQL database

To prevent any chance of loss, the user data store has relocated. Read the FishEye Upgrade guide for details.

Improved protection against XSRF attacks

We've added a security token to protect FishEye from XSRF attacks. See the Crucible upgrade guide for details of changes required by LightSCM plugins.

Change log

This section will contain information about the FishEye 3.2 minor releases as they become available. These releases will be free to all customers with active FishEye software maintenance.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide.

The issues listed below are just the highlights of all those that have been resolved for the FishEye 3.2.x releases.

15 January 2014 - FishEye 3.2.4

T	Key	Summary
	FE-5016	SVN: setting a start revision prompts a restart, not a re-index
	FE-5000	User synchronisation fails if invalid username is found
	FE-4988	"View" and "Fisheye" menus appear behind text in IE10
	FE-4531	Radio Buttons Reset When 'Apply' Is Pressed

Authenticate to retrieve your issues

4 issues

16 December 2013 - FishEye 3.2.3

T	Key	Summary
	FE-4996	Hidden Directory does not work when a hidden dir is a single dir

Authenticate to retrieve your issues

1 issue

11 December 2013 - FishEye 3.2.2

T	Key	Summary

	FE-4849	Some changesets never transition to 'done' when indexing a svn repository

Authenticate to retrieve your issues

1 issue

6 December 2013 - FishEye 3.2.1

T	Key	Summary
	FE-4984	On Windows, Fisheye attempts to make ssh keys private but appears to be unsucessful
	FE-4974	Showing the UPM admin log can't be shown, if it contains entries made by the admin password user

Authenticate to retrieve your issues

2 issues

27 November 2013 - FishEye 3.2.0

T	Key	Summary
	FE-3670	Add checkout commands onto Browse Repository page
	FE-4949	Allow searching content and diffs containing the @ symbol
	FE-5003	User will have login problem with RememberMe cookie when there is a racing condition
	FE-4971	A java.io.IOException was encountered Exception getting SVN blame...
	FE-4965	Privilege escalation
	FE-4962	There is no way to specify timeouts on FishEye's SMTP connection
	FE-4928	IE10 changeset view crashed
	FE-4927	Creating a new branch using SVN makes impossible to make diff between old revisions
	FE-4926	Git files with double quotes in the dir name are not processed correctly
	FE-4923	JFP: JIRA Fisheye plugin not encoding # character correctly in "Source" tab links
	FE-4892	Marketplace doesn't recognize new application versions
	FE-4890	Recent repos do not need to be case sensitive
	FE-4889	Disabling JIRA Integration Plugin breaks search, activity page scroll, and introduces navigation artifact
	FE-4888	JIRA Integration Plugin fails to remain disabled
	FE-4885	SVN repositories showing up as indexing in quicksearch even if they aren't
	FE-4855	java.lang.NullPointerException while upgrading to 3.1.1
	FE-4800	Add ViewCVS URL Mapping has an error
	FE-4794	Upgrading the Application Link to FishEye causes config.xml corruption
	FE-4792	Backup fails with 'The process cannot access the file because another process has locked a portion of the file'
	FE-4514	Align in-product text with docs RE: SMTP Server Admin

Authenticate to retrieve your issues

Showing 20 out of 23 issues

FishEye 3.1 release notes

27 August 2013

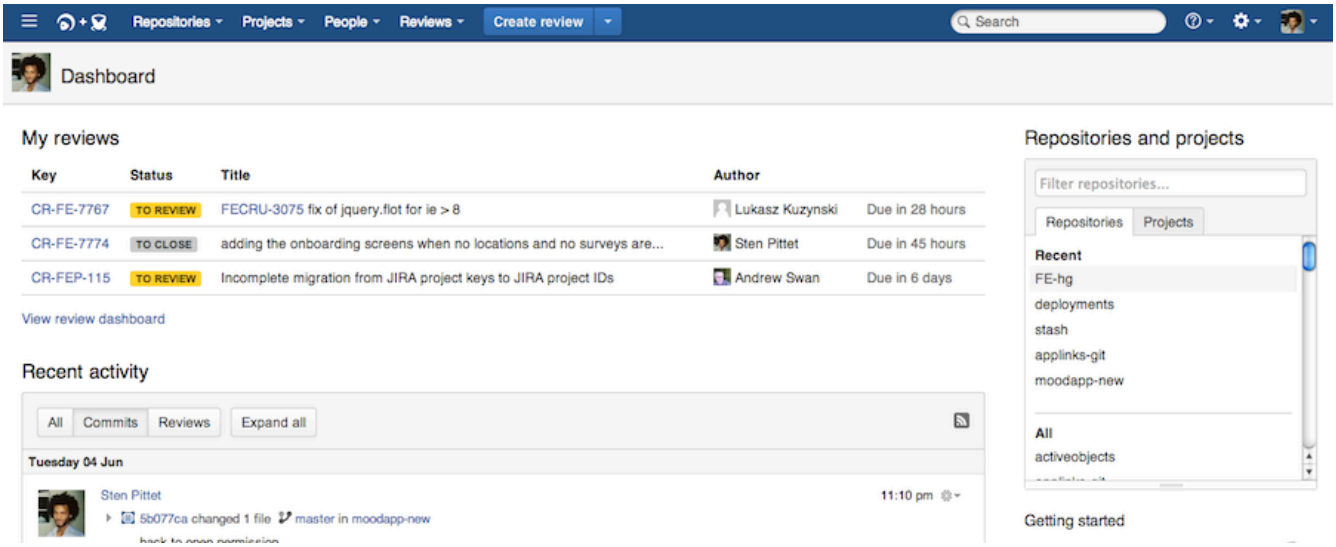
Today we're excited to release FishEye 3.1, which introduces a brand new dashboard, faster search and a new JIRA integration to ease your workflows.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide. Please also read the End of Support Announcements for FishEye.

The FishEye 3.1 changelog is at the bottom of this page.

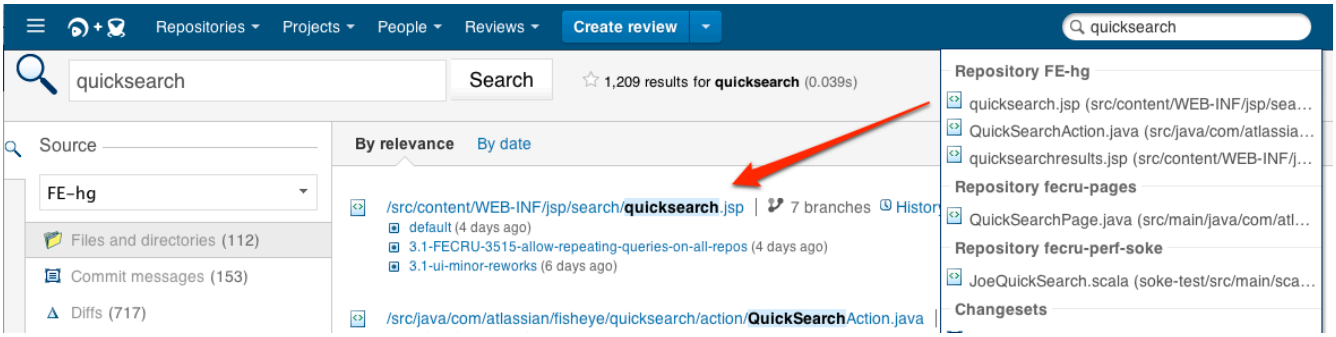
The new Dashboard

FishEye's new dashboard now includes a repository and project navigator. This will be the fastest and easiest way to jump to your recent content (try pressing Enter as soon as you launch the dashboard). It includes filters and keyboard shortcuts that facilitate navigation, especially when your organisation has hundreds of repositories.



QuickNav and QuickSearch improvements

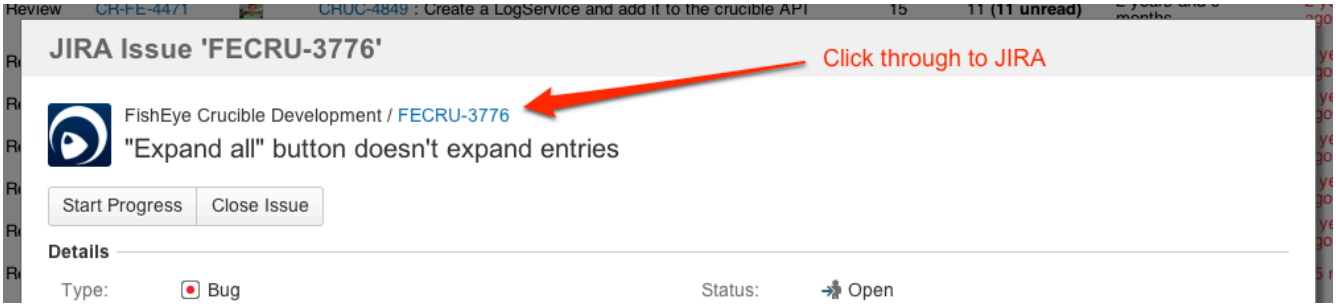
Previously in FishEye, QuickSearch was limited to searching 100 repositories concurrently. Now, in FishEye 3.1, you can search across files and directories, commit messages, and committers, for every one of your repositories. And it's 40% faster. And you can order search results by date. Furthermore, QuickNav now uses the same indexes as QuickSearch, so it's faster too, and it produces the same results as QuickSearch:



Note that upgrading to FishEye 3.1 will move some per-repository indices into the cross-repository Lucene index. See the FishEye upgrade guide for details.

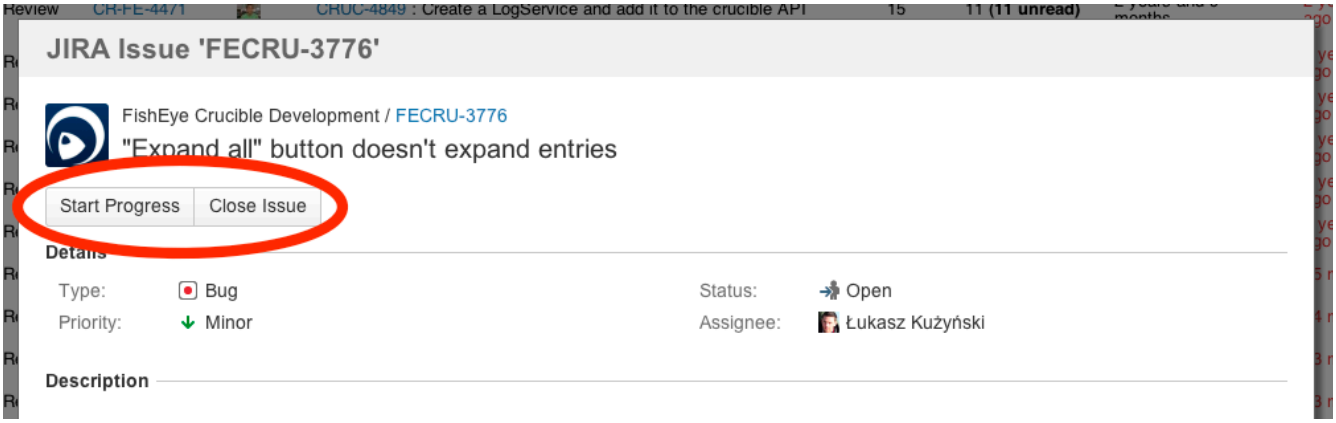
New JIRA issue dialog

Now, when you click on any linked JIRA issue key in FishEye, you'll see a dialog that displays a summary of the issue details. From that dialog, you can transition the workflow for the issue, and easily click through to edit the issue in JIRA:



Transition JIRA issues from within FishEye

Previously in FishEye, you could transition a JIRA issue linked to a review at the time that you closed the review. Now you can advance the workflow for any JIRA issue mentioned in FishEye at any time, right from within FishEye – just click the issue link to see the transitions available to you:



Read more about JIRA issue transitions in FishEye ...

Small improvements

Improved browsing performance

The display of the file source page is now 60% faster in FishEye 3.1.

Native SVN 1.7 support

FishEye 3.1 now supports the native JavaHL 1.7 SVN client. It does not support the native JavaHL 1.6 SVN client. See Native support for SVN for more details.

OpenJDK support

FishEye 3.1 now supports OpenJDK 1.7. See Supported platforms for details.

Change log

This section will contain information about the FishEye 3.1 minor releases as they become available. These

releases will be free to all customers with active FishEye software maintenance.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide.

The issues listed below are just the highlights of all those that have been resolved for the FishEye 3.1.x releases.

2 December 2013 - FishEye 3.1.6

Key	Summary
FE-4965	Privilege escalation
FE-4962	There is no way to specify timeouts on FishEye's SMTP connection
FE-4954	E160013: "File not found:" and "Attempted to open non-existent child node 'proper'" errors when viewing source on some SVN repositories

Authenticate to retrieve your issues

3 issues

30 October 2013 - FishEye 3.1.5

Key	Summary
FE-4927	Creating a new branch using SVN makes impossible to make diff between old revisions
FE-4926	Git files with double quotes in the dir name are not processed correctly
FE-4892	Marketplace doesn't recognize new application versions

Authenticate to retrieve your issues

3 issues

8 October 2013 - FishEye 3.1.4

This release also includes some minor security fixes.

Key	Summary
FE-4890	Recent repos do not need to be case sensitive
FE-4889	Disabling JIRA Integration Plugin breaks search, activity page scroll, and introduces navigation artifact
FE-4888	JIRA Integration Plugin fails to remain disabled
FE-4885	SVN repositories showing up as indexing in quicksearch even if they aren't
FE-4800	Add ViewCVS URL Mapping has an error

Authenticate to retrieve your issues

5 issues

17 September 2013 - FishEye 3.1.3 (No issues affecting FishEye)

This release fixed an issue in Crucible. See the Crucible 3.1 release notes

T	Key	Summary

Authenticate to retrieve your issues

No issues found

11 September 2013 - FishEye 3.1.2 (3 issues plus few security issues)

T	Key	Summary
	FE-4855	java.lang.NullPointerException while upgrading to 3.1.1
	FE-4840	'Tree view' resets when opening a file
	FE-4830	Quicksearch: license changes need a restart to be effective

Authenticate to retrieve your issues

3 issues

29 August 2013 - FishEye 3.1.1

T	Key	Summary
	FE-4827	Unable to create reviews after upgrade to 3.1
	FE-4792	Backup fails with 'The process cannot access the file because another process has locked a portion of the file'

Authenticate to retrieve your issues

2 issues

27 August 2013 - FishEye 3.1.0

T	Key	Summary
	FE-3840	Need to support SVN 1.7 with Native Client via JavaHL
	FE-4778	Rescanning SVN changesets causes the same changeset ids to disappear from other repositories
	FE-4769	Reload favourite star's label after changing the label
	FE-4762	Changeset comments appear in activity stream even when the user does not have permission to see that repository, or the repository is disabled
	FE-4757	3.0.2 svn upgrade task can fail when using native javaHL
	FE-4754	Large number of plugin properties harm performance
	FE-4667	FishEye SVN Indexing Fails When Filenames Have Control Characters
	FE-4376	Viewing "All Reviews" For 1000 or More Users in Oracle Throws ORA-01795
	FE-3512	A search on '*' causes an NPE
	FE-2132	QuickNav doesn't return the correct results

Authenticate to retrieve your issues

10 issues

FishEye 3.0 release notes

30 May 2013

Today we're excited to release FishEye 3.0, which introduces huge indexing performance gains – especially for Subversion repositories – and search and code visualisation upgrades to help you track changes across your project. We think development teams will be able to work faster, every day.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide.

The FishEye 3.0 changelog is at the bottom of this page.

Try it for FREE

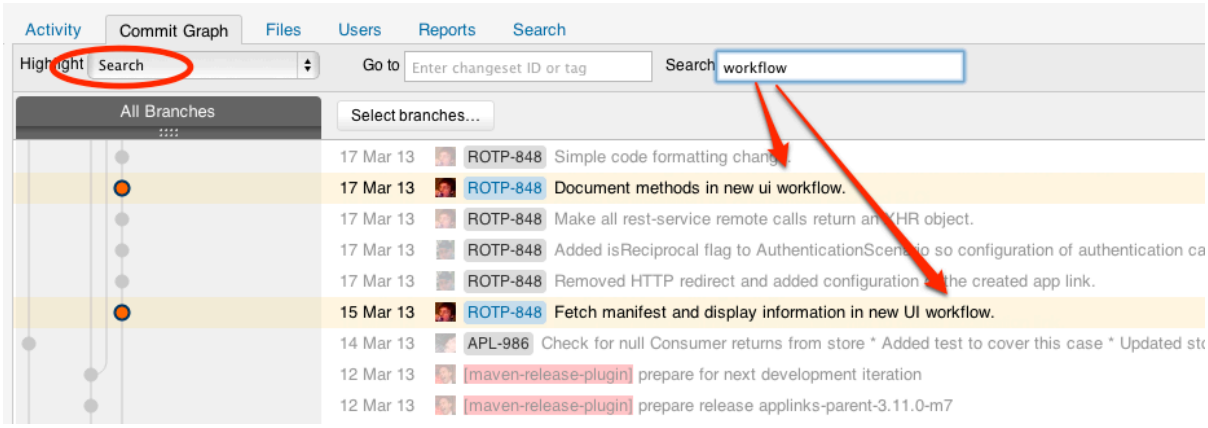
Use SVN repositories sooner

We've introduced a new approach that splits the indexing process into separate tasks that can be performed in a phased and concurrent way. This allows you to start using core functionality in FishEye, such as viewing recent changesets or commits in JIRA issues, up to 15X sooner than in FishEye 2.10. You can get on with your work, while FishEye quietly completes the fine details of indexing in the background. *Read more...*

Commit graph highlighter

We've added two new highlight types to the commit graph for a repository:

- use the **Author** highlight type to see all the changesets submitted by a particular author.
- use the **Search** highlight type to see the changesets with commit messages that contain the search term.



Read more...

Redesigned user experience

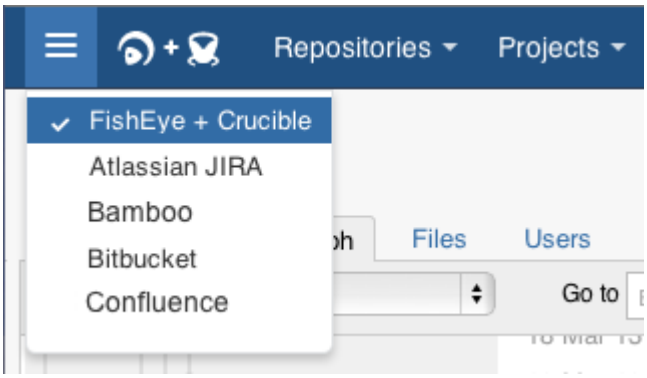
We've made a few key design changes to help you work faster with your repositories:

New global header

The new header across the top of the page makes it quicker to access recent repositories, projects, people and reviews, and has the large **Create review** button at top center, right where you can find it.

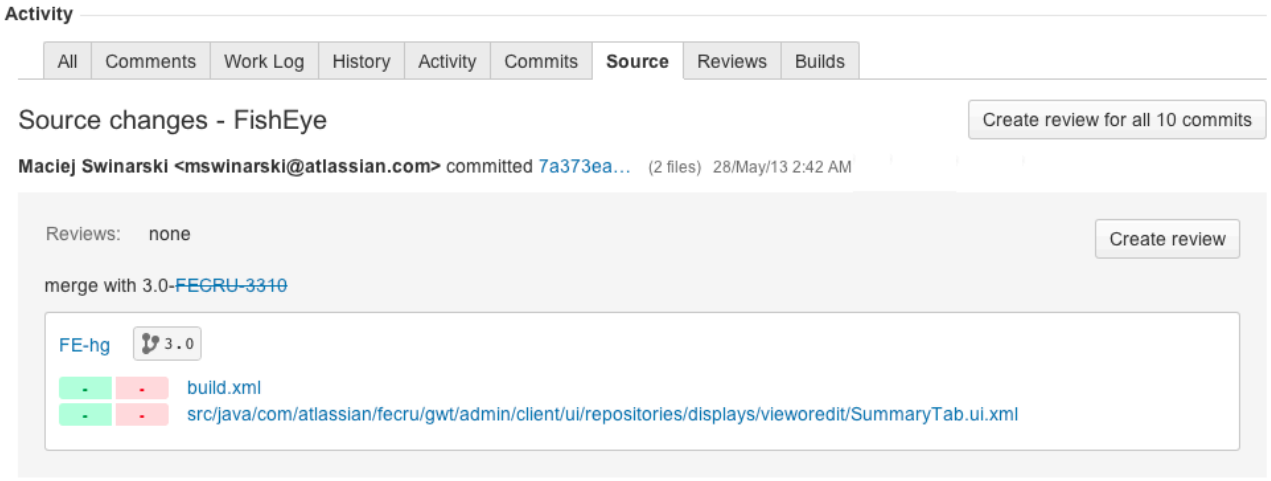
Application navigator

The new application navigator, on the left of the header, connects you directly to your other applications, such as JIRA and Bamboo. Now you can switch between FishEye and JIRA – or any other Atlassian application – all from the FishEye header. No more bookmarks in your browser; we do the job for you. Admins can easily configure which apps appear in the navigator – just click **Application navigator** in the admin area.



JIRA Source tab redesign

We wanted to simplify the experience but still give you the important information you need to make decisions around your issues.



Small improvements

Faster browsing for large teams

FishEye now uses the InfinityDB 3.0 internal database to provide improved performance for concurrent access . Your team gets a better browsing experience!

Diff hunk shortcuts

We've changed the 'j' and 'k' keyboard shortcuts to now work with diff hunks in the changeset page, rather than with files, as previously.

Improved handling of user preferences with session cookies

We've made session cookies more robust, but this may result in your existing preferences being lost at upgrade time.

Updated Jetty 8 web server

FishEye 3.0 includes an upgrade from Jetty 6 to Jetty 8 as its web server and Java servlet container.

Change log

This section will contain information about the FishEye 3.0 minor releases as they become available. These releases will be free to all customers with active FishEye software maintenance.

If you are upgrading from an earlier version of FishEye, please read the FishEye upgrade guide.

The issues listed below are just the highlights of all those that have been resolved for the FishEye 3.0.x releases.

23 July 2013 - FishEye 3.0.3

T	Key	Summary
	FE-4747	CVS changeset indexing leaks Lucene readers, file handles
	FE-4744	rescanning a perforce changeset does not update the cross repository index
	FE-4742	CrowdAuth needlessly asks Crowd for the cookie token key when validating a user login
	FE-4717	NullPointerException at Svn2ChangesetDiffSummary.addDiffPath
	FE-4697	Avoid potential recursion on same path

Authenticate to retrieve your issues

5 issues

2 July 2013 - FishEye 3.0.2

T	Key	Summary
	FE-4739	Can't use distinct in queries on tables with CLOB types in Oracle
	FE-4729	Webwork 2 code injection vulnerability
	FE-4704	"All repo" user prefs cause problems on Oracle
	FE-4687	NullPointerException in Svn2Scanner
	FE-4682	Native SVN JavaHL broken

Authenticate to retrieve your issues

5 issues

6 June 2013 - FishEye 3.0.1

T	Key	Summary
	FE-4683	NullPointerException when Repo incorrectly configured
	FE-4679	ClassCastException with native SVN client
	FE-4661	'No previous activity found' appearing twice on empty activity streams
	FE-4658	Create/Edit Project is slow when there are many users

Authenticate to retrieve your issues

4 issues

28 May 2013 - FishEye 3.0.0

T	Key	Summary
	FE-4544	Git client doc does not have any instructions

	FE-4503	Doc Update: Update "Using Dashboard"
	FE-4466	improve documentation for SVN symbolic rules
	FE-4311	Update documentation to inform customers that FishEye won't work with SVN 1.7.x and native client.
	FE-4512	provide the ability to get statistics of indexing state
	FE-4456	Change semantics of j/k shortcuts in FishEye changeset page to cycle through diffs, rather than files
	FE-3799	p4 support: Cleanup and Merge the p4 ancestry suppression flag to 2.6/7/default
	FE-4695	Pre-commit iterative reviews
	FE-4465	Add Application Navigator on BTF version
	FE-4557	Documentation Should Indicate That 'populate' Means 'copy from commit message'
	FE-4556	Indexing Hooks for Bitbucket and GitHub
	FE-4452	Highlight row in Fisheye permission screen
	FE-4450	Slow activity stream on Fisheye dashboard
	FE-4205	Possibility to change the interval of the default polling.
	FE-4083	Review smart commit should populate Crucible Objectives
	FE-3611	Improve the way avatars are loaded to avoid unnecessary calls
	FE-3045	Improve UX of Branch Selector
	FE-1182	Server/Repo restart is not required after changing the poll period value
	FE-490	Allow the poll period for a repository to have a configurable global default
	FE-425	Make file status (modify, add, delete) stand out in changelog of JIRA plugin

Authenticate to retrieve your issues

Showing 20 out of 41 issues

FishEye 2.10 release notes

15 January 2013

Atlassian is proud to present FishEye 2.10, which provides further improvements in performance.

- Visit our issue tracker to see the full list of improvements and bug fixes in FishEye and Crucible for this release.
- See the change log for FishEye 2.10.x minor releases.
- **Upgrading from a previous version of FishEye**. Upgrading FishEye should be fairly straight forward. *We strongly recommend that you back up FishEye before upgrading.* Please refer to the FishEye 2.10 Upgrade Guide for essential information about upgrading.
- **Known issues**. Please check the important technical advisories on the front page of the Knowledge Base for information about any known issues for this release.
- **JIRA 5.0 integration**. The features described below are supported by **JIRA 5.0**, or later, with the latest version of the **JIRA FishEye plugin.**

Highlights of this release:

- Repository Indexing REST API
- Bitbucket and GitHub polling integration
- Other announcements

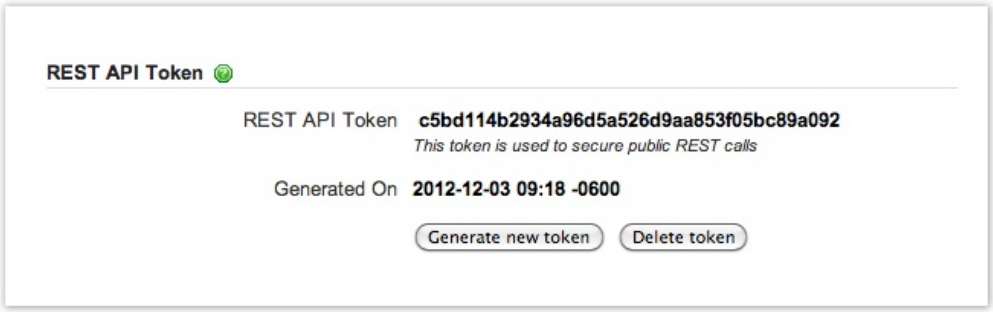
Download

Providing feedback:

Please log your votes and issues. They help us decide what needs doing, and are much appreciated!

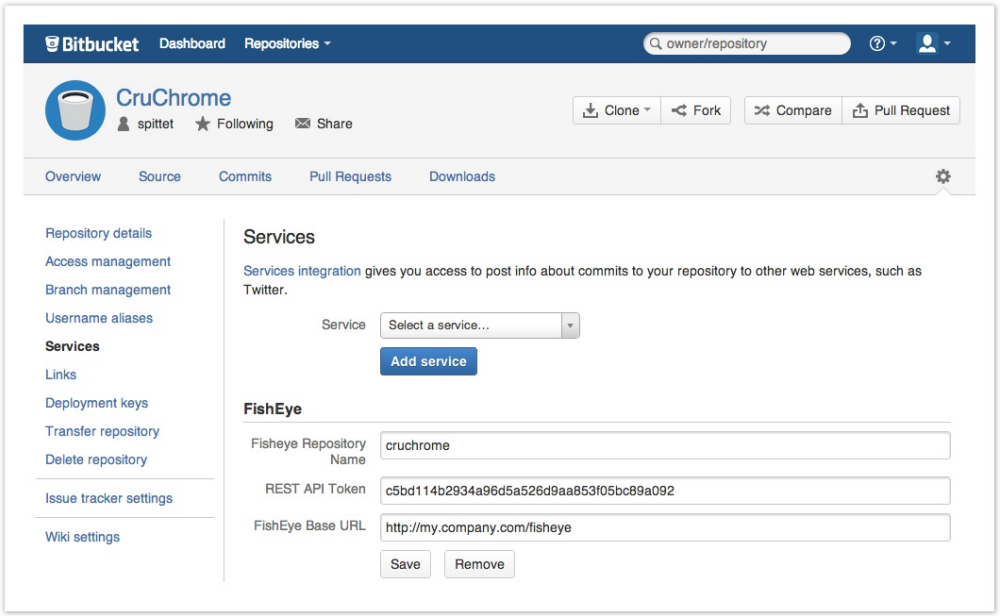
Repository Indexing REST API

With FishEye 2.10, you can now optimize your instance by using the REST API to trigger indexing when it's necessary. Instead of configuring FishEye to poll each repository every minute, you can add a post-commit or post-receive hook to your repositories which will ping FishEye when new commits need to be indexed. *Read more...*



Bitbucket and GitHub polling integration

We have implemented indexing hooks for Bitbucket and GitHub. Just provide the details of your repository and FishEye will be pinged whenever new commits need to be indexed. *Read more...*



3

Other announcements

- Improved performance of the repositories listing in the administration interface.
- Smart commits can now populate review objectives. *More...*
- The CVS poll periods can be configured. *More...*

The FishEye 2.10 team

Development

Core team

Brendan Humphreys
Conor MacNeill
Geoff Crain
Lukasz Pater
Maciej Swinarski
Piotr wity wicicki
Richard Stephens
Tom Davies
Valery Trubnikov

Team lead

Nick Pellow

Product management

Sten Pittet

Project manager

Anton Mazkovoi

Support

Ajay Sridhar
Daniel Rohan
Felipe Kraemer
Gurleen Anand
Kah Loun Foong
Malik Mangier
Patrick Hill
Renan Battaglin
Rene Verschoor
Ricardo Martins

Others

Product marketing

Giancarlo Lionetti

Technical writing

Paul Watson

Operations

James Fleming

FishEye 2.10 changelog

This page will contain information about the FishEye 2.10 minor releases as these become available. Crucible license holders should also check the Crucible 2.10 changelog. These releases will be free to all customers with active FishEye software maintenance.

Don't have FishEye 2.10 yet?

Take a look at all the features in the FishEye 2.10 release notes and see what you are missing out on!

Download

Upgrading from a previous version of FishEye

If you are upgrading, please read the FishEye 2.10 upgrade guide.

3 December 2013 - FishEye 2.10.8 Minor internal fixes

T	Key	Summary	Status
	FE-4965	Privilege escalation	CLOSED
	FE-4778	Rescanning SVN changesets causes the same changeset ids to disappear from other repositories	CLOSED

Authenticate to retrieve your issues

2 issues

17 July 2013 - FishEye 2.10.7

T	Key	Summary	Status
	FE-4739	Can't use distinct in queries on tables with CLOB types in Oracle	CLOSED

Authenticate to retrieve your issues

1 issue

05 July 2013 - Fisheye 2.10.6

T	Key	Summary	Status
	FE-4729	Webwork 2 code injection vulnerability	CLOSED

Authenticate to retrieve your issues

1 issue

06 June 2013 - Fisheye 2.10.5

T	Key	Summary	Status
	FE-4658	Create/Edit Project is slow when there are many users	CLOSED
	FE-4625	slow git output processing leads to long indexing time	CLOSED

Authenticate to retrieve your issues

2 issues

02 April 2013 - Fisheye 2.10.4

T	Key	Summary	Status
	FE-4590	Requests to dashboard and user list block when repositories are upgrading	CLOSED
	FE-4567	Email validation fails if local-part includes a '#'	CLOSED
	FE-4522	Allow FishEye notification to use the user's email address when it is set in the SMTP configuration	CLOSED
	FE-4507	Migration/Integration with MySQL 5.6 fails with "Specified key was too long" error	CLOSED

Authenticate to retrieve your issues

4 issues

19 Mar 2013 - Fisheye 2.10.3

T	Key	Summary	Status

	FE-4450	Slow activity stream on Fisheye dashboard	CLOSED
	FE-5017	XSS in the view parameter of several actions	CLOSED
	FE-4479	FishEye Not Fully Compatible With Mercurial 2.5	CLOSED

Authenticate to retrieve your issues

3 issues

25 Feb 2013 - Fisheye 2.10.2

T	Key	Summary	Status
	FE-4083	Review smart commit should populate Crucible Objectives	CLOSED
	FE-4483	NullPointerException if Crucible project key contains "-"	CLOSED
	FE-3901	Rest transactions are rolled back after they've been committed	CLOSED

Authenticate to retrieve your issues

3 issues

29 Jan 2013 - Fisheye 2.10.1

T	Key	Summary	Status
	FE-4424	Update on 'Store diff info' documentation	CLOSED

Authenticate to retrieve your issues

1 issue

15 Jan 2013 - Fisheye 2.10.0

T	Key	Summary	Status
	FE-4191	Add built-in support for use of proxy to contact external plugins site	CLOSED
	FE-4147	Extend REST API to include starting and stopping a repository and to allow for re-indexing	CLOSED
	FE-4398	Checkout errors can terminate svn repo import process	CLOSED
	FE-4361	[CVS] Allow pollPeriod to be configurable	CLOSED
	FE-4227	'Download archive' (aka tarball): name of zip is truncated to 20 characters	CLOSED
	FE-614	Extend Remote API to include methods to trigger scannow, reindex, backup etc	CLOSED
	FE-4416	repository refresh status button is broken	CLOSED
	FE-4404	DiffTextCache files not deleted	CLOSED
	FE-4399	CVS scanner activates repository on every poll, causing high native memory usage	CLOSED
	FE-4394	The repositories admin page is slow with many repositories enabled	CLOSED
	FE-4392	Svnkit fine logging causes very large debug logs	CLOSED
	FE-4391	Validation of start-rev in admin is incorrect	CLOSED
	FE-4371	Fisheye not handling empty git or hg changesets properly	CLOSED

FE-4368	Changeset Comment re-indexing incorrectly uses a threadlocal session.	CLOSED
FE-4357	Improve the way permissions are handled, as a lot of time is spent in user/repo permission checks, which should be cached	CLOSED
FE-4354	OSGI cache should be excluded from backup	CLOSED
FE-4351	Indexing stalls if parsing Perforce multi print output throws an exception	CLOSED
FE-4350	FishEye not fully compatible with Mercurial 2.3	CLOSED
FE-4349	Disable directory Listing	CLOSED
FE-4332	"The Compressed (zipped) Folder is invalid or corrupted" error in IE8 when downloading tarball from "Download Archive" link	CLOSED

Authenticate to retrieve your issues

Showing 20 out of 30 issues

FishEye 2.10 upgrade guide

Below are some important notes on upgrading to **FishEye 2.10**. For details of the new features and improvements in this release, please read the FishEye 2.10 release notes.

Upgrade notes

FishEye 2.10

- As of FishEye 2.10 the RepositoryAdminService API enforces permission checking. If you're writing a plugin that uses that API make sure to use ImpersonationService.doPrivilegedAction to execute the calls in priviledged mode.
- Note that as of FishEye 2.9 the JDBC driver for MySQL is no longer bundled with FishEye/Crucible – due to license restrictions the MySQL JDBC driver cannot be bundled with FishEye/Crucible. Please download the driver from http://dev.mysql.com/downloads/connector/j/ and install it to the FISHEYE_INST/lib directory. For more information see Migrating to MySQL.

On this page:

Related pages:

- FishEye 2.10 release notes

Upgrade procedure

Before you begin

- Test your upgrades in your test environment before rolling into production.
- Back up your *entire* FishEye instance (see Backing up and restoring FishEye data), i.e.
  - If you are backing up your FishEye instance using the Admin interface, tick all of the 'Include' checkboxes (e.g. repository and application caches, plugins and their configuration data, SQL database, etc).
  - If you are backing up your FishEye instance using the command-line interface, do not use any exclusion options.

If you are already running a version of FishEye, please follow the instructions in the general FishEye upgrade guide.

Checking for known issues and troubleshooting the FishEye upgrade

If something is not working correctly after you have completed the steps above to upgrade your FishEye installation, please check for known FishEye issues and try troubleshooting your upgrade as described below:

- **Check for known issues**. Sometimes Atlassian finds out about a problem with the latest version of FishEye after the software is released. In such cases we publish information about the known issues in the FishEye Knowledge Base. Please check for any known issues in the FishEye Knowledge Base and follow the instructions to apply any necessary patches if necessary.
- **Did you encounter a problem during the FishEye upgrade?** Please refer to the guide to troubleshooting upgrades in the FishEye Knowledge Base.
- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

FishEye 2.9 release notes

14 November 2012

Atlassian is proud to present FishEye 2.9, which provides the simplest and most powerful integration with the latest version of JIRA that FishEye has ever had.

- Visit our issue tracker to see the full list of improvements and bug fixes in FishEye and Crucible for this release.
- See the change log for FishEye 2.9.x minor releases.
- **Upgrading from a previous version of FishEye**. Upgrading FishEye should be fairly straight forward. *We strongly recommend that you back up FishEye before upgrading.* Please refer to the FishEye 2.9 Upgrade Guide for essential information about upgrading.
- **Known issues**. Please check the important technical advisories on the front page of the Knowledge Base for information about any known issues for this release.
- **JIRA 5.0 integration**. The features described below are supported by **JIRA 5.0**, or later, with the latest version of the **JIRA FishEye plugin** .

Highlights of this release:

- Simpler JIRA integration
- More JIRA data in FishEye
- Faster JIRA source tab
- Other announcements

Providing feedback:

Please log your votes and issues . They help us decide what needs doing, and are much appreciated!

Simpler JIRA integration

With FishEye 2.9, linking to JIRA is just like Plug & Play. You simply need to set up an Application Link between your JIRA server and your FishEye instance to get all the power of JIRA / FishEye integration:

- View in JIRA the list of changesets corresponding to a specific issue

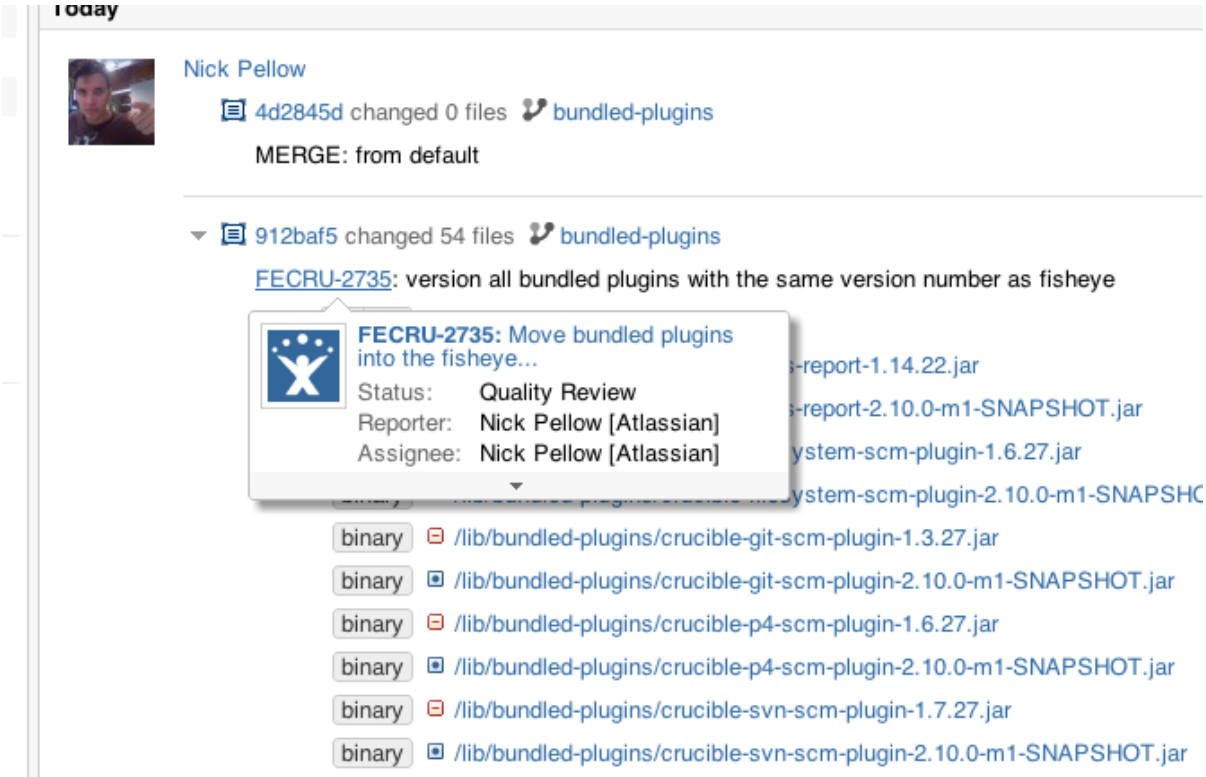
- Navigate to the related changesets from your issue
- Get the JIRA data corresponding to issues mentioned in your commit messages and your source in FishEye

You no longer need to create and maintain multiple links between projects in JIRA and repositories in FishEye. Maintaining your JIRA / FishEye integration has never been simpler! *More...*

More JIRA data in FishEye

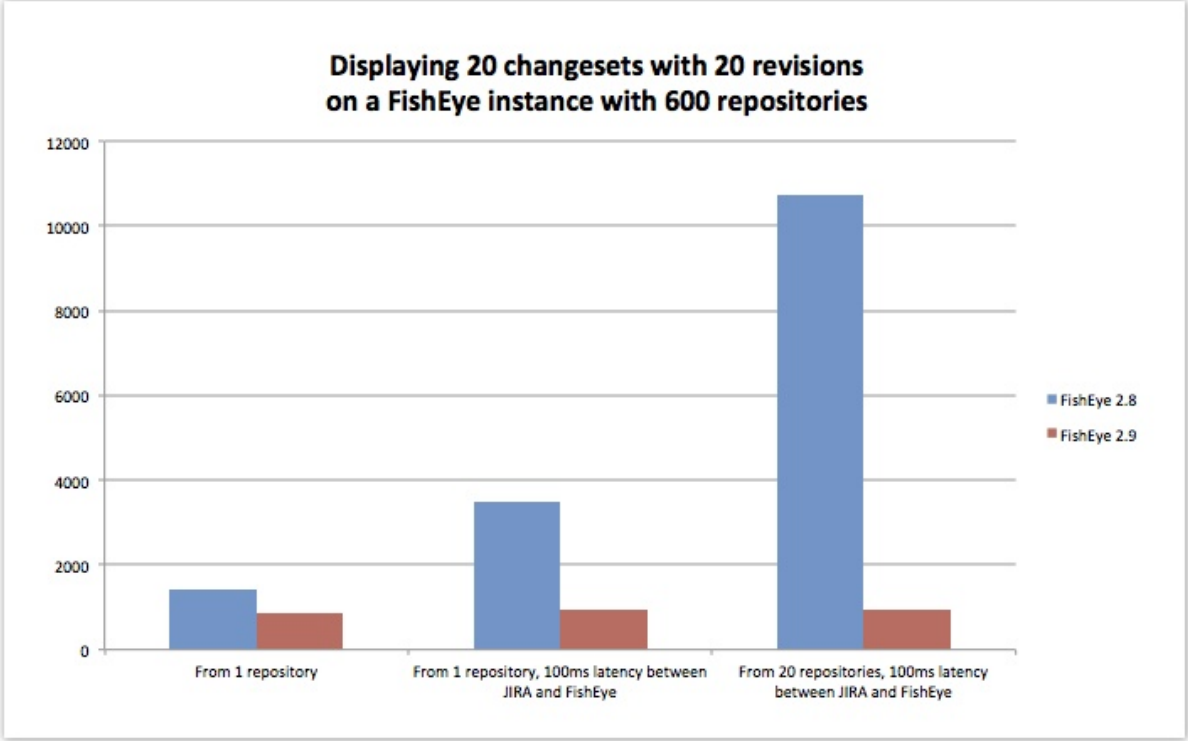
Before FishEye 2.9 you had to create links from FishEye to particular JIRA projects in order to see your issue data in FishEye. Without doing this, you just wouldn't see data for projects that were not linked to FishEye. Now, with the FishEye 2.9 release, you no longer have to create those project-specific links because all of your JIRA data is accessible in FishEye as soon as you link your instances together. Not only will you save time from not having to administer all the separate FishEye/JIRA links, but your FishEye instance is now also smarter and gets data from all the projects existing on your JIRA server.

Note that project links remain available in FishEye 2.9, but they now act as a restriction on the integration. If you set up project links from FishEye to JIRA, only issues from those particular projects will be linked in FishEye.



Faster JIRA source tab

Part of the effort of revamping the JIRA integration was to improve the performance of the issue source tab, especially for large instances with multiple repositories, each with many changesets. The chart below shows the performance gain in FishEye as a result of the effort to make it scale well as your data grows.



4

Other announcements

- **Remote API setting always on**
  We removed the Remote API setting from the Server Settings page. From this release onwards the Remote APIs will always be accessible, to make the JIRA integration straightforward.
- **JIRA FishEye Plugin option "Disable for unmapped JIRA Projects" has been removed**
  As of this release, it no longer makes sense to disable the Source and Reviews tabs when Project Links are not configured. If you still wish to disable the Source and Reviews tabs for a specific project, for example for one that has no relation to source code, you should use project-level permissions. See How do I disable the FishEye tab panel for non-code projects?
- **Changes to the diff view in the Changeset view**
  In order to facilitate file browsing, and to give a clearer interface to see all the modifications and comments on a particular file, we are now showing one file at a time in the Changeset view. You can use the file tree or the keyboard shortcuts to navigate between files.

The FishEye 2.9 team

Development

Core team

Brendan Humphreys
Conor MacNeill
Geoff Crain
Lukasz Pater
Maciej Swinarski
Piotr wity wicicki
Richard Stephens
Tom Davies
Valery Trubnikov

Team lead

Nick Pellow

Product management

Sten Pittet

Project manager

Anton Mazkovoi

Support

Ajay Sridhar
Daniel Rohan
Douglas Fabretti
Felipe Kraemer
Gurleen Anand
Kah Loun Foong
Leonardo Macedo
Malik Mangier
Patrick Hill
Renan Battaglin
Rene Verschoor
Ricardo Martins

Others

Product marketing

Giancarlo Lionetti

Quality assurance

George Filippoff

Technical writing

Paul Watson

Operations

James Fleming

FishEye 2.9 changelog

This page will contain information about the FishEye 2.9 minor releases as these become available. Crucible license holders should also check the Crucible 2.9 changelog. These releases will be free to all customers with a ctive FishEye software maintenance.

Don't have FishEye 2.9 yet?

Take a look at all the features in the FishEye 2.9 release notes and see what you are missing out on!

Download

Upgrading from a previous version of FishEye

If you are upgrading, please read the FishEye 2.9 upgrade guide.

On this page:

From 2.9.1 to 2.9.2

11 December 2012

This is a bug fix release.

T	Key	Summary
	FE-4392	Svnkit fine logging causes very large debug logs
	FE-4391	Validation of start-rev in admin is incorrect
	FE-4368	Changeset Comment re-indexing incorrectly uses a threadlocal session.
	FE-4357	Improve the way permissions are handled, as a lot of time is spent in user/repo permission checks, which should be cached

Authenticate to retrieve your issues

4 issues

From 2.9.0 to 2.9.1

19 November 2012

This is a bug fix release.

T	Key	Summary
	FE-4364	REST calls to /rest-service/repositories-v1/ and /rest-service-fe/repositories-v1/ fail when there are ClearCase repositories configured

Authenticate to retrieve your issues

1 issue

FishEye 2.9.0

14 November 2012

T	Key	Summary
	FE-4288	Profile viewing source files in FECRU with autolink enabled
	FE-281	Administer Email Watches
	FE-4194	Enable automatic repository->Jira project entity link creation
	FE-4151	Every FeCru repo sends a REST call to Crowd with a getUserInfo function when the source tab is clicked in JIRA
	FE-4150	Implement a REST endpoint that can return the results for all the repos at once. This will be used by the JIRA FishEye Plugin.
	FE-2907	Branch selector should trim whitespace
	FE-2639	Reduce memory usage of mysql backup process

	FE-2476	Avoid spurious package scanner warnings in the debug log
	FE-1071	Restructure FishEye so that svn operations do not occur on the main thread
	FE-4356	"fisheyectl.sh restore" removes some database connection attributes
	FE-4340	SVNKit upgrade 1.7.4v1 to 1.7.6
	FE-4321	Replacements sourced outside repository scope not processed
	FE-4261	500 error received from FishEye when retrieving changesets for Source tab in JIRA
	FE-4203	Non-ASCII end of line characters in commit messages break git scanning (JVM 1.7 fixes)
	FE-4109	Commit graph does not link to JIRA issue if the key is surrounded by [ ]
	FE-3892	P4 JobIds are not being returned with the REST call to rest-service-fe/changeset-v1/listChangesets
	FE-3659	Slow response times in the JIRA FishEye Plugin caused by EyeQL suffering from performance issues
	FE-1965	when showing a deleted image in a changeset diff, the old version / new version is kinda broken

Authenticate to retrieve your issues

18 issues

FishEye 2.9 upgrade guide

Below are some important notes on upgrading to **FishEye 2.9**. For details of the new features and improvements in this release, please read the FishEye 2.9 release notes.

Upgrade notes

FishEye 2.9

- **The JDBC driver for MySQL is no longer bundled with FishEye/Crucible – Due to license restrictions the MySQL JDBC driver cannot be bundled with FishEye/Crucible. Please download the driver from http://dev.mysql.com/downloads/connector/j/ and install it to the FISHEYE_INST/lib directory. For more information see Migrating to MySQL.**
- **The Connection pool manager has changed** — FishEye and Crucible now use the BoneCP connection pool manager instead of the third-party c3p0 library previously used in FishEye/Crucible 2.8 and earlier versions. The BoneCP connection pool manager has been preconfigured in FishEye and Crucible to work out-of-the-box for most customers. Read Configuring the database connection pool for a description of the default settings and for instructions on how to override them.
- **The 'Suggest Reviews' function has been removed from Crucible** — From this release onwards, a Crucible user will no longer be prompted to add to an existing review when creating a review from a changeset. We have removed this functionality based on user feedback that it was confusing and not being used. This allows users who want to add changesets to an existing review to simply navigate to that review and use the 'Add Content' option. Users can also add commits to an existing review via commit messages. Removing 'Suggest Reviews' also improves performance when creating reviews, as Crucible does not have to search for and suggest reviews.
- **Remote API is now permanently enabled** — From this release of FishEye and Crucible onwards, the 'Remote API' setting in 'Server Settings' will not be configurable. The remote API is now always turned on, to make integration with JIRA and other services more straightforward.

On this page:

Related pages:

- FishEye 2.9 release notes

Upgrade procedure

Before you begin

- Test your upgrades in your test environment before rolling into production.
- Back up your *entire* FishEye instance (see Backing up and restoring FishEye data), i.e.
  - If you are backing up your FishEye instance using the Admin interface, tick all of the 'Include' checkboxes (e.g. repository and application caches, plugins and their configuration data, SQL database, etc).
  - If you are backing up your FishEye instance using the command-line interface, do not use any exclusion options.

If you are already running a version of FishEye, please follow the instructions in the general FishEye upgrade guide.

Checking for known issues and troubleshooting the FishEye upgrade

If something is not working correctly after you have completed the steps above to upgrade your FishEye installation, please check for known FishEye issues and try troubleshooting your upgrade as described below:

- **Check for known issues**. Sometimes Atlassian finds out about a problem with the latest version of FishEye after the software is released. In such cases we publish information about the known issues in the FishEye Knowledge Base. Please check the FishEye 2.8 known issues in the FishEye Knowledge Base and follow the instructions to apply any necessary patches if necessary.
- **Did you encounter a problem during the FishEye upgrade?** Please refer to the guide to troubleshooting upgrades in the FishEye Knowledge Base.
- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

FishEye 2.8 release notes

15th August 2012

Atlassian is proud to present FishEye 2.8, which provides a variety of social features as well as performance improvements.

See the change log for FishEye 2.8.x minor releases.

Highlights of this release:

- Mentions
- Shares
- Improved performance of the activity stream
- Support for Subversion 1.7
- End of life announcements

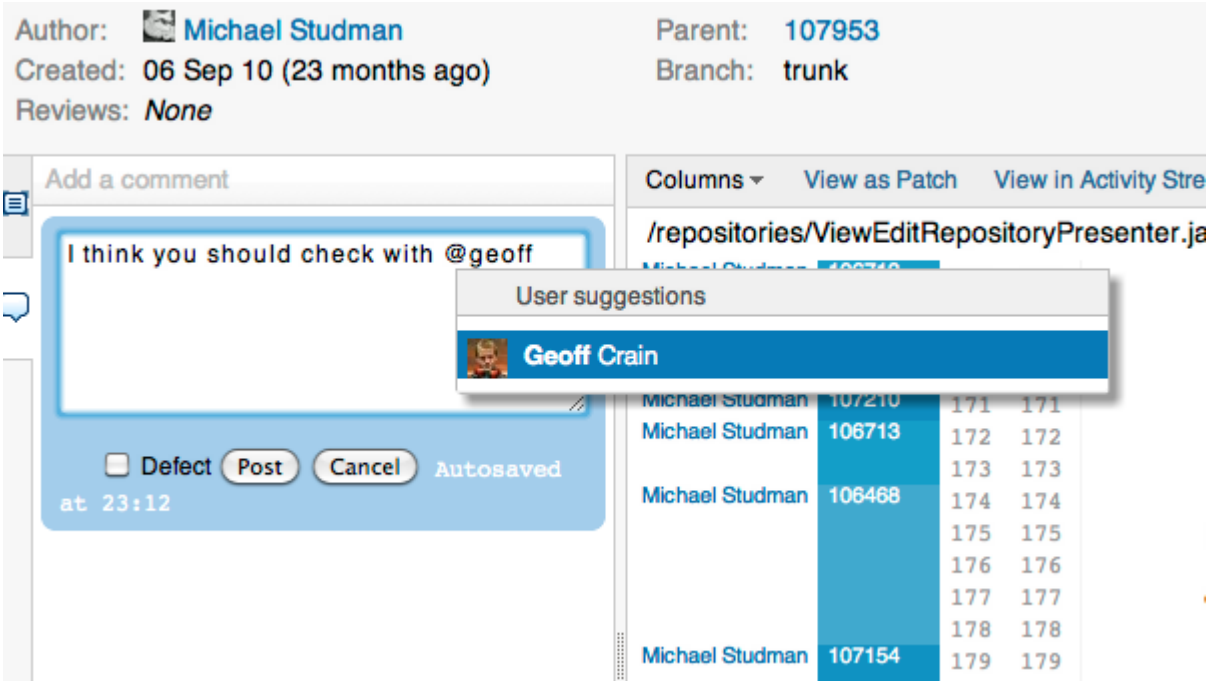
Download

Providing feedback:

Please log your votes and issues. They help us decide what needs doing, and are much appreciated!

Mentions

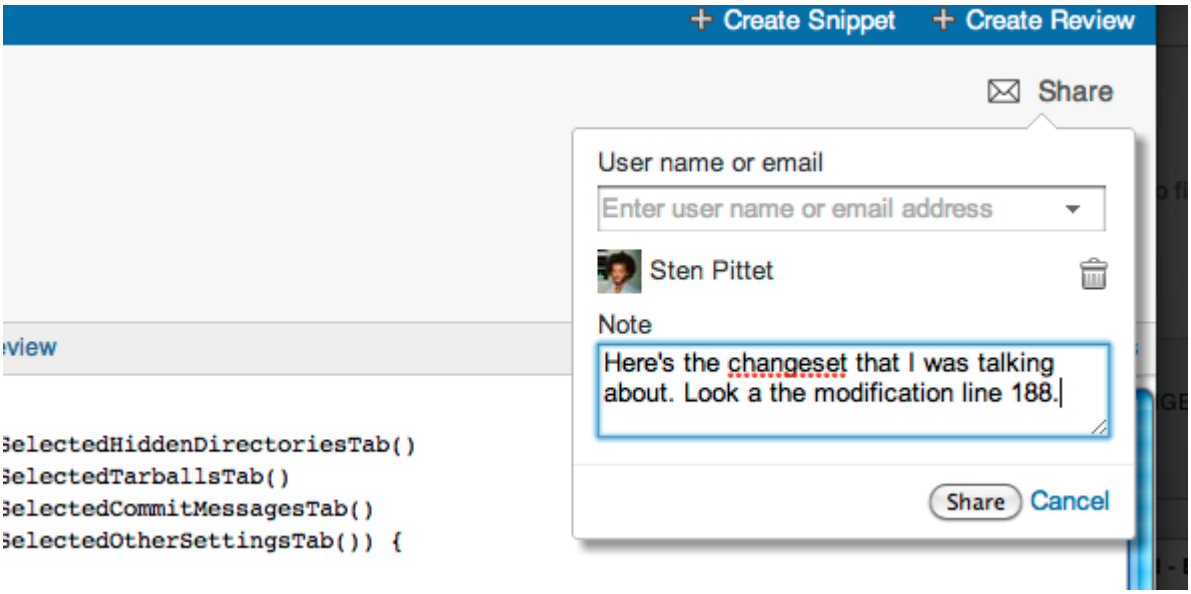
In FishEye 2.8 you can notify other users in a changeset discussion, review comment or snippet comment by using mentions. To mention someone, simply type @ and then the person's name (not their username) and choose from the suggestions that FishEye offers. FishEye sends a notification to that person, so they know that you have mentioned them.



2

Shares

Share a changeset, a review, or a source repository with other members of your team, quickly and easily from where you are working. Just click the **Share** button at the top, enter their name, username or email address and add a cheery comment. FishEye sends an email.



3

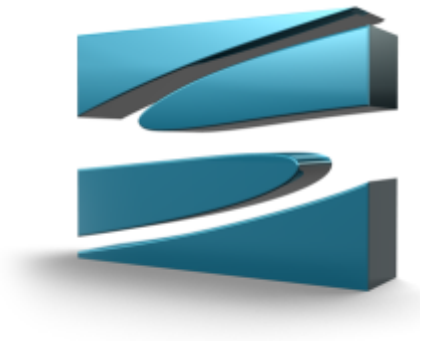
Improved performance of the activity stream

The activity streams throughout FishEye (such as the commits and review activity, and on the dashboard) have an improved user experience due to faster speed and infinite scroll (that replaces paging).

4

Support for Subversion 1.7

FishEye now supports Subversion 1.7 (although not with a native SVN client).



5

End of life announcements

- **ClearCase**
  As previously announced, IBM ClearCase is no longer supported in Fisheye 2.8.
- **Internally managed repositories**
  On August 13th 2013 we are ending support for internally managed repositories. Read more about this.
- **JIRA activity stream**
  We've removed JIRA information from the activity stream in order to simplify the user experience and improve the performance of FishEye. Please note that all other JIRA integration features still remain in FishEye.

The FishEye 2.8 team

Development

Core team

Geoff Crain
Tom Davies
Brendan Humphreys
Conor MacNeill
Richard Stephens

Team lead

Nick Pellow

Product management

Sten Pittet

Project manager

Anton Mazkovoi

Support

Ajay Sridhar
Armen Khachatryan
Daniel Rohan
Douglas Fabretti
Felipe Kraemer
Gurleen Anand
Renan Battaglin
Rene Verschoor
Zed Yap

Patrick Hill

Others

Product marketing

Giancarlo Lionetti
Jeff Park

Quality assurance

George Filippoff
Mark Hrynczak

Technical writing

Paul Watson

Operations

James Fleming

FishEye 2.8 changelog

This page will contain information about the FishEye 2.8 minor releases as these become available. Crucible license holders should also check the Crucible 2.8 changelog. These releases will be free to all customers with a ctive FishEye software maintenance.

Don't have FishEye 2.8 yet?

Take a look at all the features in the FishEye 2.8 release notes and see what you are missing out on!

Download

Upgrading from a previous version of FishEye

If you are upgrading, please read the FishEye 2.8 upgrade guide.

On this page:

From 2.8.1 to 2.8.2

5 October 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-4291	Indexing threadpool size does not shrink	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Sep 19, 2012	Oct 05, 2012
	FE-4286	java.lang.NoClassDefFoundError: Could not initialize class com.sun.jna.Native	Brendan Humphreys [Atlassian]	Malik Mangier [Atlassian]		CLOSED	Fixed	Sep 17, 2012	Oct 04, 2012
	FE-4266	Email Notifications fired from plugins fail with: java.lang.ClassCastException: com.cenqua.crucible.notification.Notification cannot be cast to java.lang.Comparable	Nick Pellow [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	Sep 05, 2012	Oct 03, 2012
	FE-4234	Fix Link for "Maximum Indexable File Size" Help	Brendan Humphreys [Atlassian]	Renan Battaglin [Atlassian]		CLOSED	Fixed	Aug 17, 2012	Oct 04, 2012
	FE-4220	Group associations are not preserved for synced users with mixed-case usernames	Tom Davies [Atlassian]	Daniel Rohan [Atlassian]		CLOSED	Fixed	Aug 08, 2012	Sep 25, 2012
	FE-3122	FIsheye is indexing changeset ids of GIT submodules when it shouldn't be causing	Brendan Humphreys [Atlassian]	None		CLOSED	Fixed	Nov 03, 2010	Nov 06, 2012

Authenticate to retrieve your issues

6 issues

From 2.8.0 to 2.8.1

29 August 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated

	FE-4253	Update documentation related to "People" tab	Paul Watson [Atlassian]	Gurleen Anand [Atlassian]		CLOSED	Fixed	Aug 27, 2012	Mar 22, 2013
	FE-4092	Revisit wording of the Hardware Supported to store FeCru data.	Paul Watson [Atlassian]	Renan Battaglin [Atlassian]		CLOSED	Fixed	May 11, 2012	May 08, 2013
	FE-4499	Mention File Size Limit Settings Apply Only to Subversion Repositories	Paul Watson [Atlassian]	Ricardo Martins [Atlassian]		CLOSED	Fixed	Feb 13, 2013	Mar 22, 2013
	FE-4238	Update database encoding warning message to be more generic	Unassigned	Gurleen Anand [Atlassian]		CLOSED	Fixed	Aug 20, 2012	Aug 28, 2012
	FE-4252	Email Review option won't work	Nick Pellow [Atlassian]	Leonardo De Macedo [Atlassian]		CLOSED	Fixed	Aug 24, 2012	Oct 01, 2012

Authenticate to retrieve your issues

5 issues

FishEye 2.8.0

15 August 2012

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	C
	FE-4094	Create 2.8.0-m1 release containing svn 1.7 support	Conor MacNeill [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	M 2
	FE-4179	Blitz-test pages changed in admin UI due to radio button issues/changes	Unassigned	Richard Stephens [Atlassian]		CLOSED	Fixed	J 2
	FE-4218	Always have resource monitor start with the FECRU server to log initial data	Unassigned	Daniel Rohan [Atlassian]		CLOSED	Fixed	A 2
	FE-4087	IllegalArgumentException: No local entity with key 'REPONAME' and type 'class com.atlassian.applinks.application.fecru.FishEyeRepositoryEntityTypeImpl' exists	Geoff Crain [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	M 2
	FE-4017	Add authenticated user in http response headers for logging ability in reverse proxy	Brendan Humphreys [Atlassian]	Issa		CLOSED	Fixed	M 2
	FE-4128	Remove deprecated TrustedAppsService	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	J 2
	FE-4095	Upgrade crowd client to 2.3.x. Currently fisheye ships with 2.2.5.	Richard Stephens [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	M 2
	FE-3819	Subversion 1.7 Support	Conor MacNeill [Atlassian]	Ajay Sridhar [Atlassian]		CLOSED	Fixed	M 2
	FE-4090	Poor Dashboard page performance for logged in users	Unassigned	Nick Pellow [Atlassian]		CLOSED	Fixed	M 2
	FE-4224	Subversion Non-Versioned Properties re-scan not working	Unassigned	Kevin Wood		CLOSED	Fixed	A 2
	FE-4222	FishEye privilege escalation vulnerability	Vitaly Osipov [Atlassian]	Paul Watson [Atlassian]		CLOSED	Fixed	A 2
	FE-4198	if non-standard crowd properties are set in config.xml, don't overwrite them if config is changed through UI	Richard Stephens [Atlassian]	Richard Stephens [Atlassian]		CLOSED	Fixed	J 2
	FE-4183	Fix Activity Stream paging (possibly with Ajaxified, infinite scroll or similar)	Brendan Humphreys [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	J 2
	FE-4182	Hg scanning chokes on invalid branch name	Brendan Humphreys [Atlassian]	Brendan Humphreys [Atlassian]		CLOSED	Fixed	J 2

FE-4176	Encoding of asian characters + accented characters in email subject doesn't seem to work properly	Unassigned	Richard Stephens [Atlassian]		CLOSED	Fixed	J 2
FE-4173	Polling Interval help text on screen needs to be updated	Roman Tekhov [Atlassian]	Daniel Rohan [Atlassian]		CLOSED	Fixed	J 2
FE-4156	Watch Emails throwing exception in certain circumstances	Unassigned	Nick Pellow [Atlassian]		CLOSED	Fixed	J 2
FE-4139	It takes 5 minutes to index a new branch in git	Conor MacNeill [Atlassian]	Adam Ahmed [Atlassian]		CLOSED	Fixed	J 2
FE-4129	Fix the Subtask integration for Crucible	Unassigned	Douglas Fabretti [Atlassian]		CLOSED	Fixed	J 2
FE-4125	Implicit committer mappings are potentially expensively computed multiple times on every request	Brendan Humphreys [Atlassian]	Brendan Humphreys [Atlassian]		CLOSED	Fixed	J 2

Authenticate to retrieve your issues

Showing 20 out of 28 issues

FishEye 2.8 upgrade guide

Below are some important notes on upgrading to **FishEye 2.8**. For details of the new features and improvements in this release, please read the FishEye 2.8 release notes.

Upgrade notes

FishEye 2.8

Please be sure to read the End of Life Announcements for FishEye/Crucible 2.8 . Most notably: Support for Clearcase repository types has been dropped.

On this page:

Related pages:

- FishEye 2.8 release notes

Upgrade procedure

Before you begin

- Test your upgrades in your test environment before rolling into production.
- Back up your *entire* FishEye instance (see Backing up and restoring FishEye data), i.e.
  - If you are backing up your FishEye instance using the Admin interface, tick all of the 'Include' checkboxes (e.g. repository and application caches, plugins and their configuration data, SQL database, etc).
  - If you are backing up your FishEye instance using the command-line interface, do not use any exclusion options.

If you are already running a version of FishEye, please follow the instructions in the general FishEye upgrade guide.

Checking for known issues and troubleshooting the FishEye upgrade

If something is not working correctly after you have completed the steps above to upgrade your FishEye installation, please check for known FishEye issues and try troubleshooting your upgrade as described below:

- **Check for known issues**. Sometimes Atlassian finds out about a problem with the latest version of FishEye after the software is released. In such cases we publish information about the known issues in the FishEye Knowledge Base. Please check for FishEye 2.8 known issues in our JIRA issue tracker and follow the instructions to apply any necessary patches if necessary.
- **Did you encounter a problem during the FishEye upgrade?** Please refer to the guide to troubleshooting upgrades in the FishEye Knowledge Base.
- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Upgrading from FishEye 2.7 with existing ClearCase Repositories

ClearCase Repositories

FishEye 2.8 no longer supports ClearCase repositories. Please refer to the End of Support Announcement for IBM ClearCase.

If you have any ClearCase repositories present in your FishEye instance, FishEye 2.8 will log a warning and will not start. The intention behind this rather draconian behaviour is to give you the opportunity to backup and update your instance before FishEye performs the upgrade operations to convert your indexes and database into a 2.8 compatible format.

You can force FIshEye to start with clearcase repositories disabled by setting the `fisheye.clearcase.disable` system property to `true`. Please see the documentation on setting up fisheye environment variables for more information.

For example, you can force startup by setting your `FISHEYE_OPTS` environment variable to

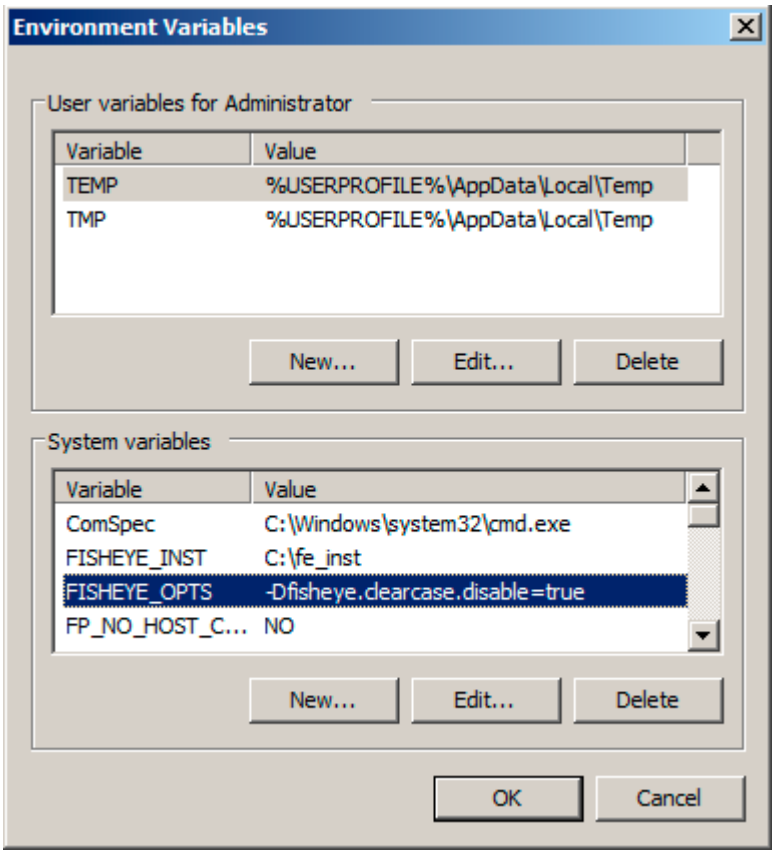
`-Dfisheye.clearcase.disable=true`

ie.:

```
$ export FISHEYE_OPTS="${FISHEYE_OPTS} -Dfisheye.clearcase.disable=true"
```

On Windows Server 2008 or Windows 7, you can configure this as follows:

1. Open the Start menu, right-click on "Computer" and select Properties
2. Click on "Advanced System Settings" on the left of the dialog
3. Click the "Environment Variables" button at the bottom of the Advanced tab
4. Add a FISHEYE_OPTS environment variable to the "System variables" section



FishEye 2.8 EAP Release Notes

FishEye & Crucible 2.8.0-m1

The 2.8.0-m1 EAP release is being provided to give early access to Subversion 1.7 support. The build contains the same features as FishEye/Crucible 2.7.14 with the updated version of svnkit necessary to work with Subversion 1.7.

We have chosen to provide this as an EAP release because we felt it was not appropriate to ship an update to a library so fundamental to FishEye's operation in a 2.7 point release.

Although this is an EAP release, the only difference between this build and the 2.7.14 release is the updated svnkit libraries. We believe, therefore, that this version can safely be deployed to production environments. As always, please take suitable backups before any such deployments. Please see the FishEye 2.7 Upgrade Guide for more information.

FishEye 2.7 Release Notes

7 September 2011
With great pleasure, Atlassian presents **FishEye 2.7** featuring Git repository management, Smart Commits and Web Hooks.

Highlights of this Release:

- Managed Git Repositories
- Smart Commits
- Web Hooks
- JIRA FishEye Plugin Improvements
- Small Improvements

DOWNLOAD latest version

- Thank you for all your issues and votes. Keep logging issues to help us keep improving!
- Read the release notices for important

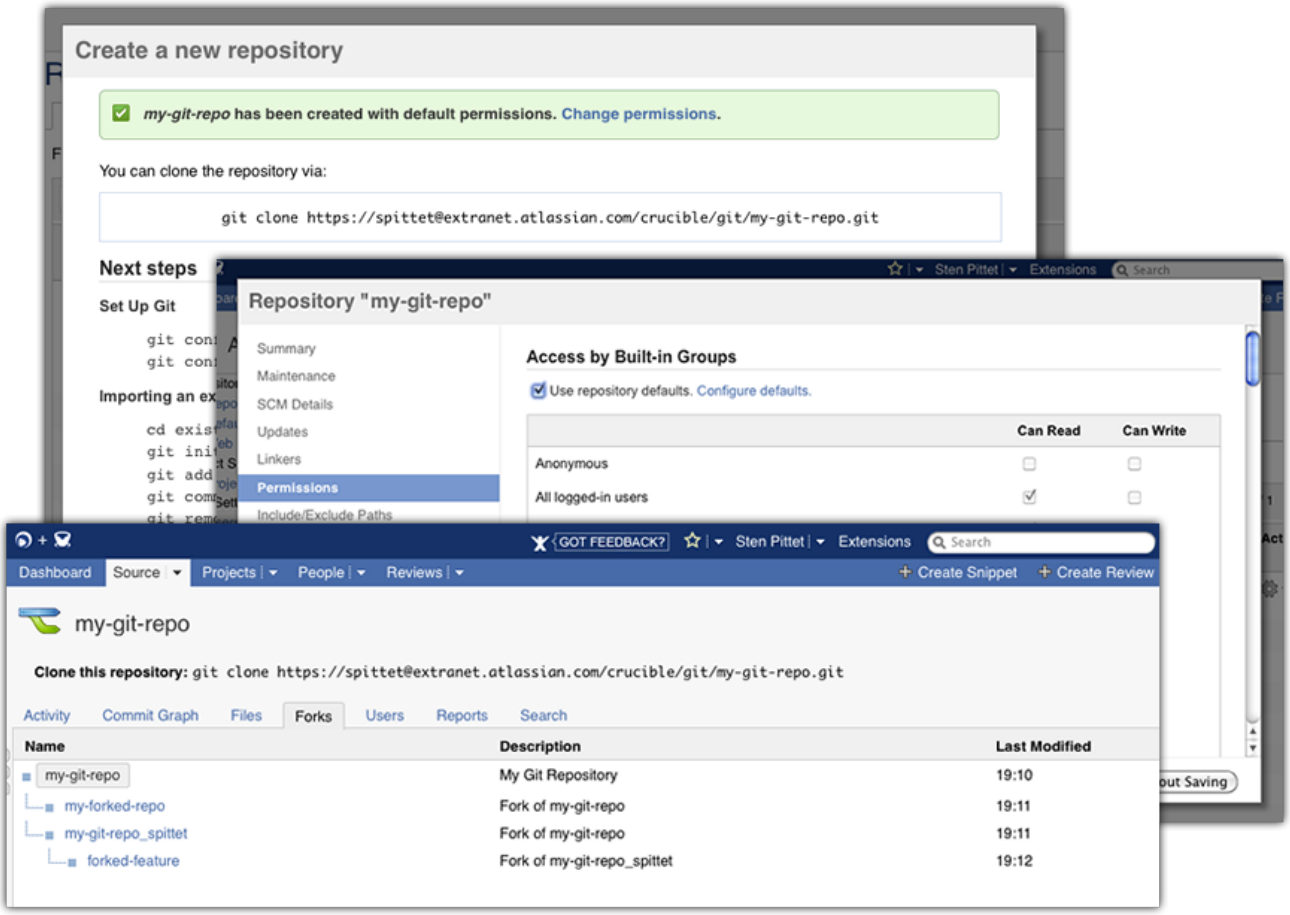
information about this release.

Highlights of FishEye 2.7

Managed Git Repositories

Administrators can **create and manage Git repositories** directly from FishEye on their own servers. FishEye Git repository management gives you the flexibility to do the following:

- Create forks and clones of existing projects.
- Manage permissions easily with push and pull access per repository.
- Visualize the hierarchy of repository forks.
- Explore and visualize changes with the commit graph



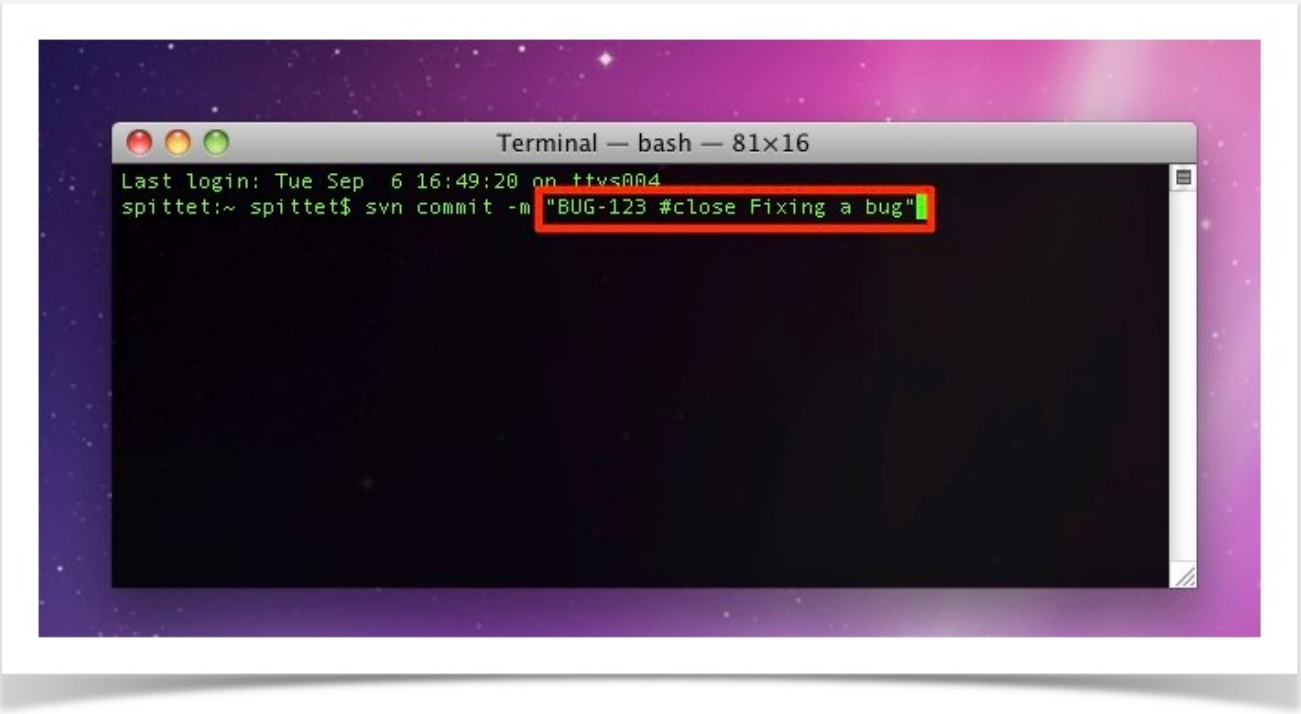
More...

Smart Commits

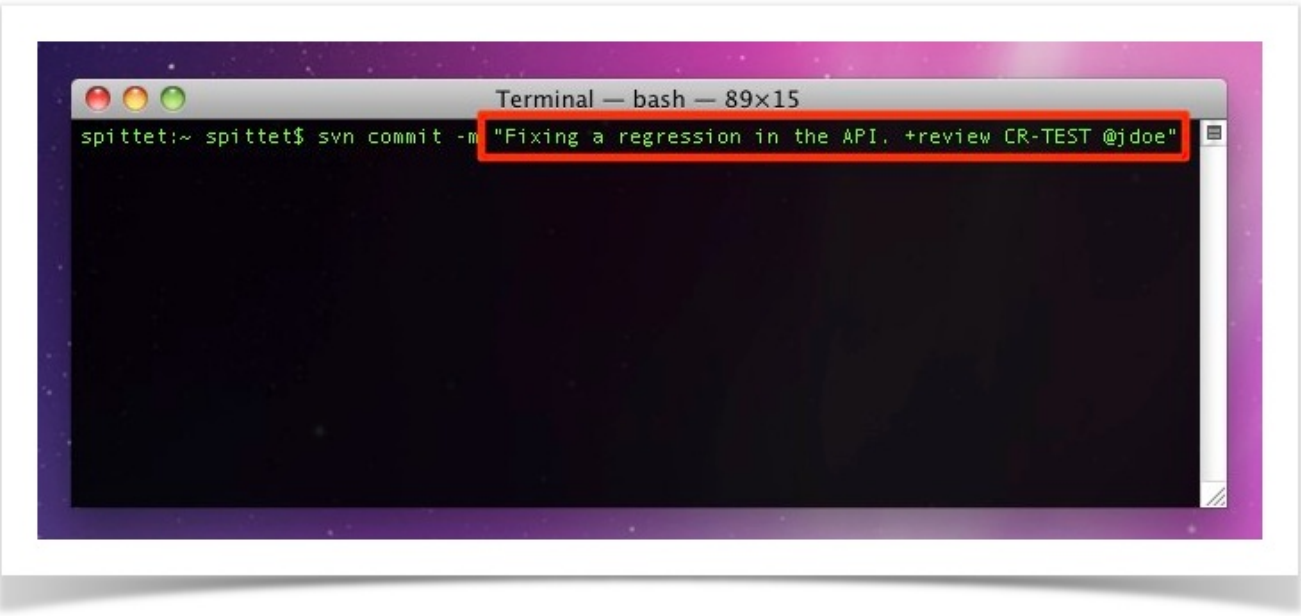
Smart Commits allow repository committers to perform actions like transitioning JIRA issues or creating Crucible code reviews by embedding specific commands into their commit messages.

FishEye 2.7 comes bundled with support for creating and updating Crucible reviews, as well as transitioning, commenting, and logging work to JIRA issues.

Placed into a commit message, the example below would close the JIRA issue `BUG-123`.



The following example would create and start a review for the current changeset in the project `CR-TEST`, as well as add the user `jdoe` as a reviewer:



For more details on using Smart Commits, please see the User's guide to Smart Commits.

JIRA issue transitioning requires at least version 3.4.5 of the JIRA FishEye plugin. It can be downloaded via the Atlassian plugin manager from within JIRA or manually downloaded from the Atlassian plugin exchange.

Smart Commits are extensible via plugins, allowing you to simply implement your own handlers to integrate with services used in your workplace. Please see the Smart Commit Tutorial for developer resources on creating a Smart Commit plugin.

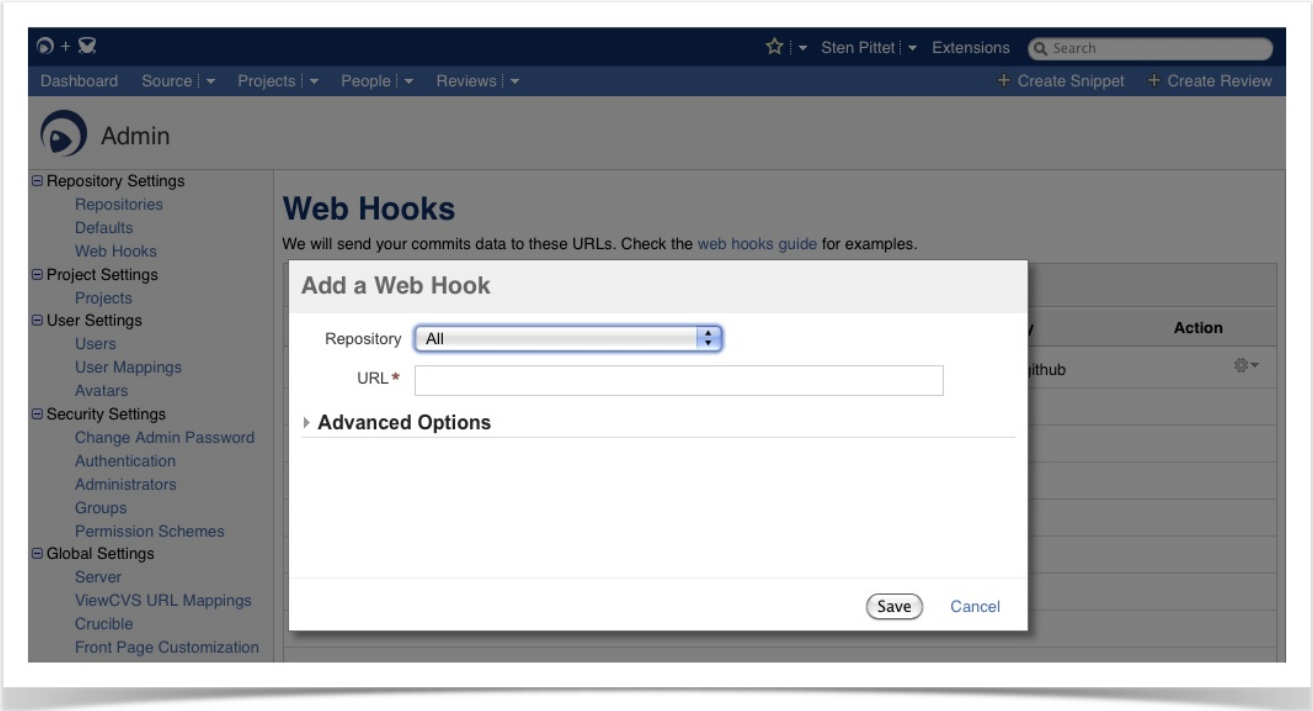
By default, smart commits are disabled if the FishEye instance has any Mercurial or Git repositories. Please see the Smart Commits administrator's guide for more details.

More...

Web Hooks

FishEye has added a Web Hooks capability. A Web Hook is a form of event-notification via HTTP POST.

In the case of FishEye Web Hooks, developers can add URLs which will receive commit(s) data through POST requests. This can be used to easily create useful gadgets in various programming languages that will give push notification of commit changes.



More...

JIRA FishEye Plugin Improvements

The FishEye team are also proud to update the JIRA FishEye plugin to version 3.4.5. Upgrades to the plugin are available immediately via the Atlassian plugin manager from within JIRA or the Atlassian plugin exchange.

This update provides significant performance improvements, most notably asynchronous loading of the "Source" and "Reviews" tab in JIRA which allows the JIRA view issue page to load without waiting for the FishEye server to respond with data.

More...

Small Improvements

FishEye 2.7 also comes bundled with numerous other bug fixes and improvements, including:

- Syntax highlighting for Java 7, Groovy, Velocity and Scala.
- FishEye can now run on Java 7
- Improved user interface for the administration screens
- HEAD label in revisions page only appears for currently selected branch (or default branch if *All* is selected).
- Improved plugin points for developers

Visit our issue tracker to see the full list of improvements and bug fixes in FishEye and Crucible for this release.

Release Notices

- **Upgrading from a previous version of FishEye.** Upgrading FishEye should be fairly straight forward. *We strongly recommend that you back up FishEye before upgrading.* Please refer to the FishEye 2.7 Upgrade Guide for further essential information about your upgrade.
- **Known Issues.** Please check the important technical advisories on the front page of the Knowledge Base for information about any known issues for this release.

FishEye 2.7 Changelog

This page contains information about the FishEye 2.7 minor releases. Crucible license holders should also check the Crucible 2.7 Changelog.
See the FishEye 2.7 Release Notes for details of what's new in 2.7.0.

⚠ Please read the FishEye 2.7 Upgrade Guide before upgrading to any of the minor releases below.

On this page:

From 2.7.13 to 2.7.14

12 June 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-4119	Missing changesets cause Hg indexing to fail	Unassigned	Tom Davies [Atlassian]		CLOSED	Fixed	May 30, 2012	May 31, 2012
	FE-4111	If Anon Access is OFF, smart commits that interact with JIRA will not work	Nick Pellow [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	May 25, 2012	Aug 01, 2012
	FE-4106	Fisheye 2.7.13 startup error when FISHEYE_INST have spaces	Nick Pellow [Atlassian]	Leo Leung		CLOSED	Fixed	May 24, 2012	Jun 08, 2012
	FE-4105	100+ requests of type "updateJiraTimeAjax!default" timing out when modifying Time Spent	Geoff Crain [Atlassian]	Daniel Rohan [Atlassian]		CLOSED	Fixed	May 24, 2012	Jun 08, 2012

FE-4103	NPE when indexing git repo with missing ancestor	Tom Davies [Atlassian]	Tom Davies [Atlassian]	CLOSED	Fixed	May 22, 2012	Feb 14, 2014	

Authenticate to retrieve your issues

5 issues

From 2.7.12 to 2.7.13

21 May 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Upd
	FE-3961	Emulate pre-2.7 behaviour: when an user is created by the user synchronisation, restore any deleted user with the same username	Brendan Humphreys [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Feb 16, 2012	Aug 201
	FE-4019	web-server/max-threads increases from 20 to 150	Nick Pellow [Atlassian]	Nick Pellow [Atlassian]		CLOSED	Fixed	Mar 14, 2012	Nov 201
	FE-4018	EyeQL Search Results not displaying author fields	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Mar 27, 2012	Apr 201
	FE-4017	Add authenticated user in http response headers for logging ability in reverse proxy	Brendan Humphreys [Atlassian]	Issa		CLOSED	Fixed	Mar 26, 2012	Apr 201
	FE-3989	Allow disabling of precise contentHash to csid mapping	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Mar 09, 2012	May 201
	FE-3922	Add custom syntax highlighting for shell scripts	Brendan Humphreys [Atlassian]	Eddie Webb		CLOSED	Fixed	Jan 27, 2012	Nov 201
	FE-3495	com.atlassian.crowd.exception.OperationFailedException if SSO is enabled with JIRA user management	Brendan Humphreys [Atlassian]	Gurleen Anand [Atlassian]		CLOSED	Fixed	Jun 14, 2011	Jul 3 201
	FE-4084	hidden text behind the left navigation panel when using Firefox 12	Unassigned	Leonardo De Macedo [Atlassian]		CLOSED	Fixed	May 01, 2012	Jun 201
	FE-4070	Fisheye does not allow username returned by AuthToken to be different from username entered by user	Richard Stephens [Atlassian]	Richard Stephens [Atlassian]		CLOSED	Fixed	Apr 19, 2012	Apr 201
	FE-4060	CC Indexing can fetch version 0 and index it even if before the start date.	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Apr 17, 2012	Apr 201
	FE-4044	Lack of usernames element in security config can stop AJPAuth working	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Apr 11, 2012	Apr 201
	FE-4043	Handle NPE caused by changeset without a position value in Commit Graph	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Apr 11, 2012	Apr 201
	FE-4036	P4 GetLatestRevision will fail for paths with no revisions - leads to excessive logging.	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Apr 05, 2012	Apr 201
	FE-4034	Copy from a file which existed before the start-revision is ignored.	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Apr 04, 2012	Apr 201
	FE-4029	Store Perforce Branch Details	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Apr 03, 2012	Apr 201
	FE-4025	Modified port not reflected in startup message of uninitialized FishEye	Brendan Humphreys [Atlassian]	Rene Verschoor [Atlassian]		CLOSED	Fixed	Mar 31, 2012	Apr 201
	FE-4024	Clearcase FileHistory can throw an exception if no HEAD revisions	Conor MacNeill	Conor MacNeill		CLOSED	Fixed	Mar 30, 2012	Apr 201

			[Atlassian]	[Atlassian]					
	FE-4020	Deleted git branches show in branch selector	Conor MacNeill [Atlassian]	Matthew Watson [Atlassian]		CLOSED	Fixed	Mar 28, 2012	Apr 201
	FE-4012	Avatars in user dropdown are too big	Richard Stephens [Atlassian]	Seb Ruiz [Atlassian]		CLOSED	Fixed	Mar 23, 2012	Apr 201
	FE-4003	JIRA Application Password Field Not Refreshing with Apply Button	Brendan Humphreys [Atlassian]	Daniel Rohan [Atlassian]		CLOSED	Fixed	Mar 15, 2012	Apr 201

Authenticate to retrieve your issues

Showing 20 out of 27 issues

From 2.7.11 to 2.7.12

16 April 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-4016	FishEye XML Vulnerability	Vitaly Osipov [Atlassian]	Paul Watson [Atlassian]		CLOSED	Fixed	Mar 26, 2012	Jan 20, 2014
	FE-4001	Perforce branches are unnecessarily, repeatedly, scanned	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Mar 15, 2012	Apr 03, 2012
	FE-4000	Corruption of CrossRepository Index Properties	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Mar 14, 2012	Mar 22, 2012
	FE-3996	Cleartool since date format need to be English	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Mar 12, 2012	Mar 22, 2012
	FE-3976	ClassCastException on startup on windows	Richard Stephens [Atlassian]	Wannes Geysen		CLOSED	Fixed	Feb 27, 2012	Apr 11, 2012
	FE-3970	Commit Calendar incorrect for projects including several paths in the same repository	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Feb 23, 2012	Feb 29, 2012
	FE-3969	The 'Parameters' on the 'Database Configuration' page are ignored.	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Feb 22, 2012	Oct 03, 2012

Authenticate to retrieve your issues

7 issues

From 2.7.10 to 2.7.11

27 February 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status
	FE-3941	Add a limit for the number of paths to index to a changeset	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED
	FE-3937	Add properties to stop branch and tag point ancestry detection for SVN	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED
	FE-3931	Null Pointer Exception com.cenqua.fisheye.rep.impl.CommonFileRevision.setAuthor(CommonFileRevision.java:149)	Brendan Humphreys [Atlassian]	Cardinal Armand du Piessis de Richelieu First		CLOSED

				Minister of Louis XIII			
	FE-3958	Content Indexing: Filter the paths to index before processing the list twice.	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED	
	FE-3957	Lucene re-index is only triggered for cross-repo until after first commit	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED	
	FE-3956	Content Indexing: Add an option to only index trunk, not root and trunk	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED	
	FE-3955	ReIndex: Only delete all the content docs if there are >0 docs in the index	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED	
	FE-3953	Per user LoC figures are meaningless when the 'All' button is selected	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	
	FE-3952	Breadcrumb links on the Users tab are incorrect when a branch is selected	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	
	FE-3939	PHP syntax highlighting incorrectly matches keywords at the end of function names	Brendan Humphreys [Atlassian]	Chris Butler		CLOSED	
	FE-3934	Changesets with large number of revisions can exhaust heap	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	
	FE-3932	FishEye appears hung on upgrade, till upgrade finishes.	Conor MacNeill [Atlassian]	Matthew Watson [Atlassian]		CLOSED	
	FE-3921	Fisheye treats files as binary if they used to have a svn:mime-type property, even after the property has been removed	Michael Heemskerk [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	
	FE-3908	Investigate lib dir containing random JS files and directories. and scmutils.jar is missing.	Unassigned	Nick Pellow [Atlassian]		CLOSED	
	FE-3905	Upgrade from 2.5->2.7 fires events for all commits and replays all smart commits	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED	
	FE-3885	Smart Commits: Using the #time command with a complex time format (i.e. 2d 4h) only saves the 2d part	Nick Pellow [Atlassian]	Felipe Cuozzo [Atlassian]		CLOSED	
	FE-3859	Unable to add users when using IE8	Seb Ruiz [Atlassian]	Felipe Kraemer [Atlassian]		CLOSED	
	FE-3851	Old broken Support Zip link still present	Brendan Humphreys [Atlassian]	Rene Verschoor [Atlassian]		CLOSED	
	FE-3608	Issue keys, wrapped in square brackets [JRA-9] are not linked correctly in quick search results	Brendan Humphreys [Atlassian]	Nick Pellow [Atlassian]		CLOSED	

Authenticate to retrieve your issues

19 issues

From 2.7.9 to 2.7.10

20 January 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3903	Unable to see FishEye activity stream in JIRA	Unassigned	Matthew		CLOSED	Fixed	Jan 19,	Jan 19,

		Activity Stream gadget on dashboard when have Repository with no description		Watson [Atlassian]				2012	2012
	FE-3902	Issue Transition controls don't appear when closing a review	Unassigned	None		CLOSED	Fixed	Jan 19, 2012	Jan 19, 2012
	FE-3900	fecru-review-issue-transition-plugin fails under fisheye 2.7.9 due to incompatible class change error	Unassigned	Nick Pellow [Atlassian]		CLOSED	Fixed	Jan 19, 2012	Jan 19, 2012

Authenticate to retrieve your issues

3 issues

From 2.7.8 to 2.7.9

19 January 2012

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3896	FishEye wrongly removes admin privileges for a Crowd group when Crowd is not up when FishEye is starting up	Nick Pellow [Atlassian]	Alex Wei [Atlassian]		CLOSED	Fixed	Jan 05, 2012	Jan 18, 2012
	FE-3891	Webwork 2 vulnerability	Vitaly Osipov [Atlassian]	Paul Watson [Atlassian]		CLOSED	Fixed	Jan 17, 2012	Dec 23, 2013
	FE-3888	FishEye does not correctly handle moved tags	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Jan 16, 2012	Jan 18, 2012
	FE-3884	DefaultUserManager.getUsersInGroup incorrectly acquires a write lock instead of a read lock	Brendan Humphreys [Atlassian]	Brendan Humphreys [Atlassian]		CLOSED	Fixed	Jan 13, 2012	Jan 13, 2012
	FE-3880	P4 client doesn't handle job names with a leading -	Brendan Humphreys [Atlassian]	Brendan Humphreys [Atlassian]		CLOSED	Fixed	Jan 09, 2012	Jan 10, 2012
	FE-3878	When processing perforce file revisions, only branch specs that have been added or modified since the last restart are considered	Anna Buttfield [Atlassian]	Anna Buttfield [Atlassian]		CLOSED	Fixed	Jan 08, 2012	Jan 09, 2012
	FE-3875	SVN property changes with content lines containing certain strings break diff parsing	Anna Buttfield [Atlassian]	Anna Buttfield [Atlassian]		CLOSED	Fixed	Jan 04, 2012	Jan 18, 2012
	FE-3862	JS errors in ie8	Anna Buttfield [Atlassian]	Anna Buttfield [Atlassian]		CLOSED	Fixed	Dec 14, 2011	Jan 18, 2012
	FE-3861	lock mismanagement in LuceneConnection	Brendan Humphreys [Atlassian]	Brendan Humphreys [Atlassian]		CLOSED	Fixed	Dec 13, 2011	Dec 13, 2011
	FE-3817	Clicking on repository in Fisheye Administration generates error dialog in IE8	Anna Buttfield [Atlassian]	Felipe Kraemer [Atlassian]		CLOSED	Fixed	Nov 04, 2011	Jan 17, 2012
	FE-3790	Trying to look at the source of an empty file results in java.lang.ArrayIndexOutOfBoundsException - -1	Brendan Humphreys [Atlassian]	Rene Verschoor [Atlassian]		CLOSED	Fixed	Oct 20, 2011	Dec 23, 2011

Authenticate to retrieve your issues

11 issues

From 2.7.7 to 2.7.8

30 November 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3846	crowd admin groups get removed on startup	Geoff Crain [Atlassian]	Geoff Crain [Atlassian]		CLOSED	Fixed	Nov 28, 2011	Nov 29, 2011
	FE-3842	"Internal Server Error" on jira issue hover	Geoff Crain [Atlassian]	Geoff Crain [Atlassian]		CLOSED	Fixed	Nov 24, 2011	Nov 24, 2011
	FE-3839	autocomplete dropdowns give inconsistent and sometimes wrong results (results missing)	Geoff Crain [Atlassian]	Geoff Crain [Atlassian]		CLOSED	Fixed	Nov 21, 2011	Nov 22, 2011
	FE-3775	Allowed Review Participants field doesn't work when some users are present	Pierre-Etienne Poirot [Atlassian]	Adam Ahmed [Atlassian]		CLOSED	Fixed	Oct 11, 2011	Nov 29, 2011
	FE-3745	Changing Block Size triggers a reindex + restart, while only a restart is needed	Pierre-Etienne Poirot [Atlassian]	Rene Verschoor [Atlassian]		CLOSED	Fixed	Sep 26, 2011	Nov 25, 2011
	FE-3040	FishEye Access Logging is not writing the username	Pierre-Etienne Poirot [Atlassian]	Felipe Kraemer [Atlassian]		CLOSED	Fixed	Apr 27, 2011	Nov 29, 2011
	FE-2473	Cannot find link for downloading latest and greatest RAW revision of a file	Pierre-Etienne Poirot [Atlassian]	David Hergert		CLOSED	Fixed	Feb 19, 2010	Nov 30, 2011

Authenticate to retrieve your issues

7 issues

From 2.7.6 to 2.7.7

21 November 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3836	In SVN, if a tag is created and the identified parent changeset id is not indexed by FishEye the tag won't show up in the branches and tags dropdown or on the commit graph	Anna Buttfield [Atlassian]	Anna Buttfield [Atlassian]		CLOSED	Fixed	Nov 17, 2011	Nov 18, 2011
	FE-3833	Perforce jobids are not indexed	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Nov 15, 2011	Nov 17, 2011
	FE-3832	Links in RSS feed do not use absolute URLs	Seb Ruiz [Atlassian]	Oleg Semyonov		CLOSED	Fixed	Nov 15, 2011	Nov 17, 2011
	FE-3826	Jetty log messages are not reported anywhere	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Nov 11, 2011	Nov 17, 2011
	FE-3822	mime type mapping for file names rather than suffixes removes first character of candidate file name before matching with key filename	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Nov 09, 2011	Nov 17, 2011
	FE-3174	When navigating forward in time in activity stream, items are shown earliest to latest down the page	Michael Studman [Atlassian]	None		CLOSED	Fixed	Sep 22, 2010	Feb 01, 2012
	FE-3155	Browse/diff menu inactives look active and vice-versa	Jonathan Poh [Atlassian]	None		CLOSED	Fixed	Dec 13, 2010	Nov 17, 2011

Authenticate to retrieve your issues

7 issues

From 2.7.5 to 2.7.6

8 November 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3816	Shutdown the thread pool once the background upgrade tasks have been completed	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Nov 04, 2011	Nov 04, 2011
	FE-3791	Change authentication providers to return CommunicationException when appropriate	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Oct 21, 2011	Nov 02, 2011
	FE-3772	Add option to enable client authentication SSL	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Oct 09, 2011	Oct 18, 2012
	FE-3611	Improve the way avatars are loaded to avoid unnecessary calls	Anna Buttfield [Atlassian]	Renan Battaglin [Atlassian]		CLOSED	Fixed	Jul 29, 2011	Jan 23, 2013
	FE-3823	If the total length of the branch names selected for the commit graph is too long, the spinner never disappears	Unassigned	Tom Davies [Atlassian]		CLOSED	Fixed	Nov 09, 2011	Nov 10, 2011
	FE-3812	When synchronizing users with LDAP, do not deactivate users if the error is a communication problem	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Nov 02, 2011	Nov 02, 2011
	FE-3809	Upgrade of instances using Oracle fail when upgrade_75.sql is run	Anna Buttfield [Atlassian]	Renan Battaglin [Atlassian]		CLOSED	Fixed	Oct 31, 2011	Nov 02, 2011
	FE-3794	Sourcing a Tag from a Tag Can cause FishEye not to display tag contents	Conor MacNeill [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Oct 24, 2011	Nov 03, 2011
	FE-3787	When synchronizing users with Crowd, do not deactivate users if the error is a communication problem	Pierre-Etienne Poirot [Atlassian]	Adam Ahmed [Atlassian]		CLOSED	Fixed	Oct 20, 2011	Aug 20, 2012
	FE-3784	Modified tags fail in the branch selector	Adam Ahmed [Atlassian]	Adam Ahmed [Atlassian]		CLOSED	Fixed	Oct 19, 2011	Nov 03, 2011
	FE-3783	Activity Stream UI sometimes slow in git - it makes calls to git remote show.	Adam Ahmed [Atlassian]	Adam Ahmed [Atlassian]		CLOSED	Fixed	Oct 17, 2011	Nov 04, 2011
	FE-3767	WARN messages 'Repository index does not match the repository configuration' incorrectly logged after upgrading to 2.7	Conor MacNeill [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	Fixed	Oct 06, 2011	Nov 02, 2011
	FE-3645	Old non-http/s SimpleLinker configuration passes upgrade but results in error when trying to open repo	Pierre-Etienne Poirot [Atlassian]	Rene Verschoor [Atlassian]		CLOSED	Fixed	Aug 15, 2011	Nov 04, 2011
	FE-3620	Displayname not being picked up correctly in some cases with JIRA user management	Pierre-Etienne Poirot [Atlassian]	Gurleen Anand [Atlassian]		CLOSED	Fixed	Aug 04, 2011	Jan 17, 2012

Authenticate to retrieve your issues

14 issues

From 2.7.4 to 2.7.5

21 October 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3799	p4 support: Cleanup and Merge the p4 ancestry suppression flag to 2.6/7/default	Conor MacNeill [Atlassian]	Matthew Watson [Atlassian]		CLOSED	Fixed	Sep 27, 2011	Jan 23, 2013
	FE-3768	FishEye logs lots of ApplicationPermissionException when synchronising Crowd users that have no permission on FishEye / Crucible	Michael Heemskerk [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	Fixed	Oct 06, 2011	Oct 12, 2011
	FE-3788	Git does not report an error when creating a managed repository over an existing one	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Oct 19, 2011	Oct 20, 2011

	FE-3777	Fix documentation of the managed repositories' REST API	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Oct 12, 2011	Oct 12, 2011
	FE-3774	Search results page fails to render for EyeQL queries > 4000 characters on Oracle.	Michael Heemskerk [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	Fixed	Oct 11, 2011	Oct 12, 2011
	FE-3755	SVN: Paths under tag directories are displayed as deleted / empty directories (greyed out)	Michael Heemskerk [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	Fixed	Sep 30, 2011	Oct 12, 2011
	FE-3752	Repository path, includes, excludes and hidden directories should be copied to newly created forks	Tom Davies [Atlassian]	Tim Pettersen [Atlassian]		CLOSED	Fixed	Sep 29, 2011	Oct 05, 2011
	FE-3751	'Store diffs' setting is not respected for initial imports of SVN repositories	Michael Heemskerk [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	Fixed	Sep 29, 2011	Oct 09, 2011
	FE-3737	Cannot customize front page with global anonymous access turned off	Tom Davies [Atlassian]	Gurleen Anand [Atlassian]		CLOSED	Fixed	Sep 22, 2011	Oct 07, 2011
	FE-3727	Review coverage report screen messed up when using Firefox 6.0.2 and Chrome 12.0.7	Michael Studman [Atlassian]	Camila Ayres [Atlassian]		CLOSED	Fixed	Sep 16, 2011	Oct 26, 2011
	FE-3721	Error when changing content of a review	Tom Davies [Atlassian]	Federico Silva Armas [Atlassian]		CLOSED	Fixed	Sep 15, 2011	Jan 23, 2013
	FE-3698	Web Hooks payload received with "application/xml" as the content type	Anna Buttfield [Atlassian]	Sten Pittet [Atlassian]		CLOSED	Fixed	Sep 07, 2011	Oct 24, 2011
	FE-3692	When pushing to a Git managed repo while another client is cloning, the client which is cloned freezes	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Sep 06, 2011	Oct 09, 2011
	FE-3678	Provide a more friendly message to the users	Anna Buttfield [Atlassian]	Zed Yap [Atlassian]		CLOSED	Fixed	Sep 05, 2011	Oct 13, 2011
	FE-2625	code metrics error for branches directories	Pierre-Etienne Poirot [Atlassian]	Tracey Lum [Atlassian]		CLOSED	Fixed	Jun 03, 2010	Oct 11, 2011

Authenticate to retrieve your issues

15 issues

From 2.7.3 to 2.7.4

20 September 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3748	NullPointerException while loading user preferences from database, causing login failures	Michael Heemskerk [Atlassian]	Michael Heemskerk [Atlassian]		CLOSED	Fixed	Sep 27, 2011	Sep 28, 2011
	FE-3747	If you edit and then save a managed repository in Internet Explorer, the text "fork of null" will appear and the forks view will no longer be available.	Tim Pettersen [Atlassian]	Tim Pettersen [Atlassian]		CLOSED	Fixed	Sep 27, 2011	Sep 27, 2011
	FE-3742	After a user has been deleted and then added again, subsequent deletions fail	Tom Davies [Atlassian]	Tom Davies [Atlassian]		CLOSED	Fixed	Sep 26, 2011	Sep 27, 2011
	FE-3741	Manual Auth Re-Sync button doesn't do anything	Tom Davies [Atlassian]	Phil Davies		CLOSED	Fixed	Sep 23, 2011	Sep 27, 2011
	FE-3738	Log files not included in support zip	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]		CLOSED	Fixed	Sep 23, 2011	Jan 17, 2012
	FE-3735	Commit events fired on Git repos when branches are added/removed	Seb Ruiz [Atlassian]	Matt Ryall [Atlassian]		CLOSED	Fixed	Sep 22, 2011	Sep 28, 2011
	FE-3734	Error upon exceeding license limit with Crowd user	Tom Davies	Gurleen		CLOSED	Fixed	Sep 22,	Sep 27,

		management	[Atlassian]	Anand [Atlassian]				2011	2011
	FE-3716	Hg authentication fails with special character in password	Michael Studman [Atlassian]	Gurleen Anand [Atlassian]		CLOSED	Fixed	Sep 14, 2011	Jan 17, 2012
	FE-3168	Restricting Fisheye content in Projects Has No Effect	Michael Studman [Atlassian]	None		CLOSED	Fixed	Dec 06, 2010	Feb 01, 2013

Authenticate to retrieve your issues

9 issues

From 2.7.2 to 2.7.3

20 September 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3717	Augment the git-http-backend from 1min to 1hour	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]		CLOSED	Fixed	Sep 15, 2011	Sep 19, 2011
	FE-3635	Backup fails if write.lock file still exists in lucene	Michael Studman [Atlassian]	Renan Battaglin [Atlassian]		CLOSED	Fixed	Aug 09, 2011	Jan 17, 2012

Authenticate to retrieve your issues

2 issues

From 2.7.1 to 2.7.2

19 September 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3616	Change com.cenqua.fisheye.logging.DailyRollingFileAppender to keep the extension .log	Conor MacNeill [Atlassian]	Felipe Cuozzo [Atlassian]		CLOSED	Fixed	Aug 02, 2011	Jan 17, 2012
	FE-3718	Eclipse Git integration (egit) fails to clone from managed Git repository	Unassigned	Tim Pettersen [Atlassian]		CLOSED	Fixed	Sep 15, 2011	Sep 19, 2011
	FE-3711	Some Directory links in Quick Search have double encoded URLs	Jason Hinch [Atlassian]	Jason Hinch [Atlassian]		CLOSED	Fixed	Sep 12, 2011	Jan 17, 2012
	FE-3696	Web Hooks should be deleted when deleting a repository	Tom Davies [Atlassian]	Seb Ruiz [Atlassian]		CLOSED	Fixed	Sep 07, 2011	Sep 14, 2011
	FE-3693	Pagination numbers at bottom of Repositories screen overlap with system info footer	Jonathan Poh [Atlassian]	Rene Verschoor [Atlassian]		CLOSED	Fixed	Sep 06, 2011	Sep 15, 2011
	FE-3647	Google collections and Guava both bundled, package exported without version	Conor MacNeill [Atlassian]	Richard Wallace [Atlassian]		CLOSED	Fixed	Aug 15, 2011	Sep 14, 2011

Authenticate to retrieve your issues

6 issues

From 2.7.0 to 2.7.1

9 September 2011

This is a bug fix release. The complete list of issues is below.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated
	FE-3701	CommitHookModuleDescriptor#enable can fail if ActiveObjectsConfiguration for Commit Hooks Plugin is not available yet	Jason Hinch [Atlassian]	Jason Hinch [Atlassian]		CLOSED	Fixed	Sep 08, 2011	Sep 12, 2011
	FE-3700	Can't reset user password	Pierre-Etienne Poirot [Atlassian]	Conor MacNeill [Atlassian]		CLOSED	Fixed	Sep 07, 2011	Sep 09, 2011
	FE-3695	Deleting a web hook dialog has a broken layout	Jason Hinch [Atlassian]	Seb Ruiz [Atlassian]		CLOSED	Fixed	Sep 07, 2011	Sep 09, 2011
	FE-3694	Testing a Web Hook doesn't send mock changeset displayId	Seb Ruiz [Atlassian]	Seb Ruiz [Atlassian]		CLOSED	Fixed	Sep 07, 2011	Sep 07, 2011
	FE-3018	Unnecessary implicit antglobbing in Query3	Brendan Humphreys [Atlassian]	Brendan Humphreys [Atlassian]		CLOSED	Fixed	Apr 11, 2011	Oct 05, 2012

Authenticate to retrieve your issues

5 issues

FishEye 2.7 Upgrade Guide

Below are some important notes on upgrading to **FishEye 2.7**. For details of the new features and improvements in this release, please read the FishEye 2.7 Release Notes.

On this page:

Upgrade Notes

FishEye 2.7

- When FishEye 2.7 is upgraded, it will run a one-time upgrade task that will create a new index of file revisions, and minor database indexing tasks. These should have no functional impact and should complete in a few minutes.
- FishEye 2.7 adds ActiveObjects support, which allows plugins to store configuration data in the database. Any already scheduled backups will not have plugin data included during backups. Please modify your scheduled backup configuration appropriately to include ActiveObjects backups.
- If you are using Custom authentication, FishEye 2.7's managed repositories have changed the responsibilities of the `com.cenqua.fisheye.user.plugin.FishEyeAuthenticator` class. For external repositories the `hasPermissionToAccess()` method is still invoked to check whether a user may access a particular repository. For internal or managed repositories, `hasPermissionToAccess()` is used to determine whether a user has both read and write access to the repository.

Upgrade Procedure

Before you begin

- Test your upgrades in your test environment before rolling into production.
- Back up your *entire* FishEye instance (see Backing up and restoring FishEye data), i.e.
  - If you are backing up your FishEye instance using the Admin interface, tick all of the 'Include' checkboxes (e.g. repository and application caches, plugins and their configuration data, SQL database, etc).
  - If you are backing up your FishEye instance using the command-line interface, do not use

any exclusion options.

If you are already running a version of FishEye, please follow the instructions in the general FishEye Upgrade Guide .

Checking for Known Issues and Troubleshooting the FishEye Upgrade

If something is not working correctly after you have completed the steps above to upgrade your FishEye installation, please check for known FishEye issues and try troubleshooting your upgrade as described below:

- **Check for known issues**. Sometimes Atlassian finds out about a problem with the latest version of FishEye after the software is released. In such cases we publish information about the known issues in the FishEye Knowledge Base. Please check the FishEye 2.7 Known Issues in the FishEye Knowledge Base and follow the instructions to apply any necessary patches if necessary.
- **Did you encounter a problem during the FishEye upgrade?** Please refer to the guide to troubleshooting upgrades in the FishEye Knowledge Base.
- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

RELATED TOPICS

FishEye 2.7 Release Notes

FishEye FAQ

FishEye FAQ

Answers to frequently asked questions about configuring and using FishEye.

- Top Evaluator Questions
  - How do I fix problems with indexing my repository?
  - How do I setup JIRA integration?
  - How do I setup LDAP or external user management?
  - How do I speed up slow CVS updates?
  - How do I start FishEye as a Windows service?
  - How do I view changesets and diffs?
  - How is FishEye licensed?
  - What kind of search capabilities does FishEye have?
  - What programming languages are supported?
- CVS FAQ
  - How does FishEye calculate CVS changesets?
  - How is changeset ancestry implemented for CVS?
- Example EyeQL Queries
  - How do find changes made to a branch after a given tag?
  - How do I filter results?
  - How do I find changes between two versions, showing separate histories?
  - How do I find changes made between two version numbers?
  - How do I find commits without comments?
  - How do I find files on a branch, excluding deleted files?
  - How do I find files removed from a given branch?
  - How do I find revisions made by one author between versions?
  - How do I select the most recent revisions in a given branch?
  - How do I show all changesets which do not have reviews?
- FishEye Developer FAQ

Do you have a question, or need help with FishEye? Please create a support request.

CVS FAQ

FishEye CVS FAQ

How does FishEye calculate CVS changesets?

FishEye's goal is to allow changesets to be seen as a consistent stream of atomic commits. Revisions are collated into the same changeset provided that:

- They have the same commit comment.
- They are by the same author.
- They are on the same branch.
- The changeset does not span more than 10 minutes.
- The same file does not appear in a changeset more than once.

How is changeset ancestry implemented for CVS?

About Changeset Ancestry in FishEye

When FishEye indexes a CVS respository, it synthesizes a changeset identifier to group file-level changes into a single consistent changeset. The grouping is described in this FAQ: How does FishEye calculate CVS changesets?.

Changeset ancestry was added in FishEye 2.6. Changeset ancestry refers to the linking of a changeset to a preceding/parent changeset(s). This allows you to view the development progress of your repository using the Commit Graph (see Viewing the Commit Graph for a Repository).

Changeset Ancestry for CVS

For CVS respositories, changeset ancestry is implemented, as follows:

- For all but the first change on a branch, FishEye chooses the most recent change on that branch as the parent changeset.
- For the first change on a branch, FishEye examines the branchpoints of all files in the branch and chooses the latest changeset that affected any such files as the parent changeset.

This approach ensures that a branch, whose first change is to a file which is very old, is not considered to have been branched at the time that file was last changed. It is considered to be branched at the last change to the repository instead.

Example EyeQL Queries

EyeQL
• How do find changes made to a branch after a given tag? • How do I filter results? • How do I find changes between two versions, showing separate histories? • How do I find changes made between two version numbers? • How do I find commits without comments? • How do I find files on a branch, excluding deleted files? • How do I find files removed from a given branch? • How do I find revisions made by one author between versions? • How do I select the most recent revisions in a given branch? • How do I show all changesets which do not have reviews?

For more information on using EyeQL, see the Reference guide.

How do find changes made to a branch after a given tag?

Find changes made to Ant 1.5.x after 1.5 FINAL:

```
select revisions where on branch ANT_15_BRANCH and after tag ANT_MAIN_15FINAL group
by changeset
```

How do I filter results?

This query, finds files removed on the Ant 1.5 branch, but just returns the person and time the files were deleted:

```
select revisions where modified on branch ANT_15_BRANCH and is dead return path,
author, date
```

How do I find changes between two versions, showing separate histories?

As above, but show the history of each file separately:

```
select revisions where between tags (ANT_MAIN_15FINAL, ANT_151_FINAL] group by file
```

How do I find changes made between two version numbers?

Find changes made between Ant 1.5 and 1.5.1:

```
select revisions where between tags (ANT_MAIN_15FINAL, ANT_151_FINAL] group by
changeset
```

How do I find commits without comments?

Using the Advanced Search and EyeQL you can find commits that do not have comments using the following query:

```
select revisions from dir / where  comment = "" group by changeset
```

How do I find files on a branch, excluding deleted files?

Find files on branch and exclude delete files:

```
select revisions where modified on branch ANT_15_BRANCH and not is deleted group by
changeset
```

How do I find files removed from a given branch?

Find files removed on the Ant 1.5 branch:

```
select revisions where modified on branch ANT_15_BRANCH and is dead group by
changeset
```

How do I find revisions made by one author between versions?

Find changes made by conor to Ant 1.5.x since 1.5.0:

```
select revisions where between tags (ANT_MAIN_15FINAL, ANT_154] and author = conor
group by changeset
```

How do I select the most recent revisions in a given branch?

Find Java files that are tagged ANT_151_FINAL and are head on the ANT_15_BRANCH: (i.e. files that haven't changed in 1.5.x since 1.5.1)

```
select revisions from dir /src/main where is head and tagged ANT_151_FINAL and on
branch ANT_15_BRANCH and path like *.java group by changeset
```

How do I show all changesets which do not have reviews?

The following query will return any changesets that have not been reviewed.

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

FishEye Developer FAQ

This page contains answers to frequently asked questions posed by FishEye developers. For detailed information about developing in FishEye, see the Fisheye Developer documentation.

Feel free to comment, make submissions, or pose your own question on FishEye Development here.

- **Q**: *I'm getting the error "API access is disabled" as a response from* `http://fisheye/api/rest/rep`

`ositories` on my installation. How do I enable the API as a Fisheye administrator?
Click here to expand...

A: A toggle to enable the API under "Server Settings" in the web admin interface existed in versions prior to 2.9 (see Configuring the FishEye web server for more details).

See the FishEye 2.9 Release Notes

- **Q**: *Is there any way to return unique results from an EyeQL query?*
  Click here to expand...

  A: It is not currently possible to return unique results.
  An improvement request exists: FE-1136. Your vote and comments on that issue are appreciated.

General FAQ

FishEye General FAQ
• About database encoding • About the Lines of Code Metric • Cannot View Lines of Code Information in FishEye • Finding your Server ID • How Do I Archive a Branch within Perforce • How do I Avoid Long Reindex Times When I Upgrade? — If reindexing your repository takes longer than you can allow, you can use a temporary copy of your repository and FishEye instance to reduce downtime during the reindexing process. • Mercurial Known Issues • Ordering of Branches Important When Visualising Git Changesets • Permanent authentication for Git repositories over HTTP(S) • Perforce Changesets and Branches • What SCM systems are supported by FishEye? • Automating Administrative Actions in Fisheye

About database encoding

It is possible to have files in your repository whose names differ only in case, e.g. `Foo.java` and `foo.java`. Hence, your database will need to use rules for comparing string values which recognise that upper and lower case letters are different, that is, the database should use 'case sensitive collation'.

If your database was originally configured to use case-insensitive and/or non-UTF8 collation, FishEye will display the following message at the bottom of your screen:

> *"Your database is not using a case sensitive UTF8 encoding for character fields."*

The following sections provide instructions for changing your database collation for each database type supported by FishEye and Crucible.

On this page:

Related pages:

- Migrating to an external database

MySQL

⚠ **Please take a backup of your database before changing its collation.**

To change your collation to `utf8_bin` you need to change your database's default collation, but as this only affects newly created tables you will also need to change the collation on the table for which case sensitivity is

critical.

Change your database's collation

Use the ALTER DATABASE command, as follows:

```
alter database character set utf8 collate utf8_bin;
```

Change collation for the CRU_STORED_PATH table

Use the ALTER TABLE command, as follows:

```
alter table cru_stored_path convert to character set utf8 collate
utf8_bin;
```

Oracle

Oracle collation encoding must be configured when installing the database server. It cannot be configured on a per database level. When installing Oracle, you should select the `AL32UTF8` encoding.

PostgreSQL

⚠ **Please take a backup of your database before changing its collation.**

If you have created your PostgreSQL database with the incorrect encoding, you will need to dump your database, drop it, create a new database with the correct encoding and reload your data.

You can do this using the standard database migration procedure – instead of migrating from HSQLDB to PostgreSQL, you migrate from a PostgreSQL database with the incorrect encoding to one created with the correct encoding.

SQL Server

⚠ **Please take a backup of your database before changing its collation.**

Unfortunately, changing the database collation for an existing SQL Server database (even using the ALTER DATABASE ... COLLATE... statement) does not change the collation for existing objects stored in the database. See http://blogs.msdn.com/b/qingsongyao/archive/2011/04/04/do-not-alter-database-collation-in-your-server.aspx for an explanation of this.

The recommended route for changing the collation for SQL Server is to migrate to a new database that has the correct collation configuration. You can do this using the standard database migration procedure – instead of migrating from HSQLDB to SQL Server, you migrate from a SQL Server database with the incorrect collation to one created with the correct collation.

The correct collation to use when you create the new SQL Server database is Latin1_General_CS_AS.

About the Lines of Code Metric

This page contains information about the Lines of Code metric and how it is processed and represented by FishEye.

On this page:

Definition

Lines of Code or LOC (also known as Source Lines of Code - SLOC) is a quantitative measurement in computer programming for files that contains code from a computer programming language, in text form. The number of lines indicates the size of a given file and gives some indication of the work involved.

LOC is literally the count of the number of lines of text in a file or directory. In FishEye, blank lines and comment lines are counted toward the total lines of code.

LOC for a file/directory is the total number of lines in the relevant files, while LOC for an author is the number of lines **blamed on that author**. Neither of these should ever be less than zero. However, the **change** in LOC over a period of time can be negative if there was a net reduction in the LOC over the period.

Disadvantages

While it can be useful, LOC has some well documented disadvantages. Keep these disadvantages and limitations in mind when using LOC in your work environment.

In addition, the nature of branching in SCM applications means that calculating a LOC value for a whole project is not possible. A naive summation of the LOC of all the branches will give a meaningless number that jumps every time a branch is copied to create a new branch. Thus, in FishEye we usually look at the LOC of the trunk, unless we can infer from the context that another branch is more appropriate.

LOC in FishEye

FishEye calculates the LOC for the trunk only. For SVN repositories it can calculate LOC for a branch if it is "tricked" to see the branch as part of the trunk. FishEye also calculates the LOC for each user, unless that facility is turned off in the repository (see Store Diff Info). The LOC count will include all files except those identified by the SCM as binary.

FishEye presents LOC data as charts of the change in LOC over time, and as informational statistics in various places:

- **Chart pages**
  The best way to explore the evolution of LOC in your project is the LOC chart report where you can easily filter the LOC by branch, author, file extension and date range. Here you can investigate what caused a particular spike in the LOC charts, or find the user whom has the most lines of code blamed on them and how this has changed over time.
- **Repository-specific activity pages**
  These show trunk LOC statistics for the repository, limited to the directory being viewed and its subdirectories. The LOC charts show the LOC for the directory, using trunk LOC unless the directory can be identified as a branch.
- **User pages**
  Here, the statistics pane in the sidebar shows the trunk LOC blamed on the user for the all repositories that have user-specific LOC enabled. The chart shows the trunk LOC from all the repositories that the user has contributed to.
- **The global User List page**
  This shows the trunk LOC for all users from the repositories that have user-specific LOC enabled. Repository-specific user lists (in repositories that have user-specific LOC enabled) show the trunk LOC for the users and committers, limited to the directory being viewed and its subdirectories.
- **Project pages**
  This shows a chart of the LOC for all associated repository paths, and statistics include the trunk LOC for those paths.

User-Specific LOC

The evolution of user-specific LOC over subsequent commits can appear at first glance to be counter-intuitive. It is important to keep in mind that the LOC for a given user is the number of lines in the repository that were last changed by them (as calculated by Fisheye).

A couple of simple examples:

- Alice adds a files with 30 lines to the SCM. Her LOC for this file is now 30. She then edits the file, deletes 10 lines and adds 20 (+20 -10). Her LOC is now 40, as is the LOC of the file.
- Alice adds a files with 30 lines to the SCM. Her LOC for this file is now 30. Now Bob edits the file, deletes 10 lines and adds 20 (+20 -10). Alice now has LOC of 20, because Bob deleted 10 lines that were blamed on her, and Bob has LOC of 20, from the 20 lines he added. The total LOC is still 40.

A user can have LOC on a branch that they have never committed on, *if something that has been blamed on them is copied*. For example, a developer may have never committed to a particular branch, but FishEye may still report a lot of LOC for them in that area.

One current limitation of FishEye's user-specific LOC calculation is the handling of merging. For example, if a file has been changed on both trunk and branch, and the changes made on the branch are merged to trunk, the changes made on branch will generally be blamed on the person who did the merge; **not** the person who made the change.

Cannot View Lines of Code Information in FishEye

Symptoms

The LOC (Lines of Code) information in FishEye cannot be seen, for example in charts or when viewing the statistics for a user.

See About the Lines of Code Metric for more information about the usage of the LOC (Lines of Code) metric in FishEye.

Cause

There are four possible causes:

- LOC data will not be shown for users if the Store diff info setting is disabled. If a page is being viewed in FishEye that relates to a particular user or committer, and the Store Diff Info setting is disabled, no LOC information for the user will be visible.
- LOC data is currently not supported for Mercurial repositories.
- LOC data is currently not supported for Git repositories.
- The SVN repository is indexing branches only.

Resolution

Cannot view LOC information for specific users or committers:

- Enable the Store Diff Info setting for the repository in `Administration > Repository Settings > Repositories`, click on the repository name, and then "SCM Details". A full re-index needs to be performed on the repository after enabling this setting for FishEye to collect the diff information for all revisions in the repository. Please note that the Store Diff Info setting is always enabled for CVS repositories.

Cannot view LOC information for Mercurial repositories:

- There is an outstanding feature request for LOC support in Mercurial.

Cannot view LOC information for Git repositories:

- There is an outstanding feature request for LOC support in Git.

The SVN repository is indexing branches only:

- FishEye can calculate LOC for a branch if it is "tricked" to see the branch as part of the trunk.

Finding your Server ID

Your Server ID can be found in the FishEye administration area.

To find your Server ID:

1. Navigate to FishEye's administration area.
2. Click **System Info** (under 'System Settings').

The Server ID for your FishEye server is displayed in the 'License' section.

The Server ID should match the one set for your license. You can check this at http://my.atlassian.com.

How Do I Archive a Branch within Perforce

In SVN, a branch exists as a separate directory. However in Perforce, files are given a label to identify them as belonging to the branch. Thus it may not be possible to download the branch as a tarball via FishEye.

You may be able to download the branch as a tarball, depending on your structure:

⚠ If it is not a single folder, then it is not possible to download the tarball in your perforce repository.

1. In FishEye, navigate to your perforce repository.
2. In the Constraint section on the left, select the branch. This will return the directories that belong to that branch.
3. If it is one single folder, download the tarball of it. Under `constraint and sub directories`, there is a panel tarball giving options on how to download the directory.

How do I Avoid Long Reindex Times When I Upgrade?

Mitigating lengthy reindex times

If reindexing your repository takes longer than you can allow, you can use a temporary copy of your repository and FishEye instance to reduce downtime during the reindexing process.

⚠ Most upgrades (even major ones) do not require a reindex. If a reindex is required, this will always be explicitly mentioned in the Upgrade Guide for that release.

On this page:

Reindexing with a temporary copy of your FishEye instance

This section describes how to perform a full reindex of a particular repository. Note that, depending on the repository size, the reindex could take up to several days.

To reindex a temporary copy of your FishEye instance:

1. Make a copy of your FishEye instance to another server. See 'How to make a temporary copy of your FishEye instance' below for instructions.
2. Upgrade the temporary FishEye, then start it up, connected to your repository. It will automatically begin the scanning process.
   ℹ If you are concerned about the repository being overloaded by the scanning process, you can make a copy of that as well. See 'How to make a temporary copy of your repository' below for instructions.
   If you do that, you must edit the `config.xml` of your temporary FishEye instance to point to your temporary repository.
3. The copied instance will run its course without affecting your production instance.
   a. Shutdown both your servers completely.
   b. Make a backup of your `FISHEYE_INST` directory.
   c. Replace the `FISHEYE_INST/var/cache` directory on live FishEye with the `FISHEYE_INST/var/cache` from your test server.
   d. Download the latest FishEye/Crucible from Atlassian downloads.
   e. Follow the instructions in the Upgrade Guide to upgrade to the new version.
4. The scan of the temporary FishEye instance (and repository, if you copied that also) is complete. You're now free to delete the temporary copy(s).

How to make a temporary copy of your FishEye instance

To make a copy of your FishEye instance, follow the instructions for Migrating FishEye Between Servers.

How to make a temporary copy of your repository

To make a copy of your repository use rsync (for CVS repositories in the Linux environment) or svnsync documentation (for Subversion only).

How to reindex a single repository on a test server

If you need to reindex your repository on your production system but don't want to burden your production server, carry out the following steps:

1. Install another instance of FishEye on a test server (the same FishEye version as the one you are using).
2. Add a repository to FishEye with the exact same name and details as that referenced by the production server.
3. Let it finish indexing. Go to **Administration** > **View Repository List** > **Stop** (shown next to the name of your repository) and disable on both production and test.
4. Copy over the `FISHEYE_INST/var/cache/REPO` directory on the production FishEye with the `FISHEYE_INST/var/cache/REPO` directory from the test server.

5. Trigger a review revision data re-index: **Administration > Repository > Maintenance > Review-Revision Data Index.**

For this procedure, neither server needs to be shut down.

Upgrading your cross-repository index using a temporary staging server

This section describes how to upgrade the cross-repository index for selected repositories. Note that, depending on the repository size, the reindex will typically finish in a few hours, but should never take longer than a few days.

In this procedure it is assumed that you have a production server (referred to as *PROD* in these instructions) that is running a FishEye version earlier than 3.1, and a separate staging server (*STAGING*) that will be used to perform the cross-repository index upgrade offline.

1. Make a live backup of the *PROD* server with the following options:
    a. Repository and application caches
    b. SQL database

    You can do this either from the FishEye Admin area (go to **Administration** > **System Settings** > **Backup**), or from a command line, for example:

    ```
    $ ./bin/fisheyectl.sh backup -f ~/Documents/backup.zip --no-uploads
    --no-templates --no-plugins --cache --no-ao
    ```

2. Install FishEye 3.1, or a later version, on the *STAGING* server.
3. Restore the backup of *PROD* to the *STAGING* server.
4. Start FishEye on the *STAGING* server. Note that:
    - The cross-repository index upgrade will start automatically on the *STAGING* server. If you want to perform the cross-repository index upgrade for selected repositories only, it is safe to remove unwanted repositories from the *STAGING* server now, either by going to **Administration** > **Repositories**, or by using REST endpoints (see below).
    - The *STAGING* server doesn't need to have access to configured SCM's as the cross-repository upgrade task does not interact with them.
    - You may want to disable polling on the *STAGING* server. You can either go to **Administration** > **Repository Settings** > **Defaults** > **Updater** to disable polling for all repositories (although this will *not* affect particular repositories that have been configured to ignore default settings), or go to **Administration** > **Repository Settings** > **Repositories** > **Repository X** > **Updates** to disable polling for just *Repository X*. Disabling polling is not required, but will avoid logging errors to the FishEye log file if the SCMs are not accessible from the *STAGING* server.

5. Wait for the cross-repository index upgrade to finish on the *STAGING* server. Check by going to **Administration** > **Repositories**.
6. Stop the *STAGING* server.
7. Make a full backup of the *PROD* server and then stop it.
8. Install the same version of FishEye on the *PROD* server as used on the *STAGING* server (as in step 2 above).
9. Delete the following FishEye indexes on the *PROD* server and replace them with the equivalent caches from the *STAGING* server. You can choose your preferred option to copy files between machines using ssh/scp/rsync, possibly combined with tar/zip tools. The example below shows how the scp command could be used:

```
ssh PROD
cd FISHEYE_INST  # replace FISHEYE_INST with the location of your FISHEYE_INST
folder
rm -rf cache/globalfe
rm -rf var/cache/repoX  # repeat for each repository repoX that was upgraded
on STAGING server
scp -r STAGING:STAGING_FISHEYE_INST/cache/globalfe cache/
scp -r STAGING:STAGING_FISHEYE_INST/var/cache/repoX var/cache/ # repeat for
each repository repoX that was upgraded on STAGING server
```

10. Start the *PROD* server.
11. All the changesets that were added to SCMs after backing up the *PROD* server in step 1 will now be indexed on the *PROD* server.

The only drawback with this procedure is that changeset comments added for changesets in *PROD* after step 1 will not get indexed, so they will not appear in the activity stream. There is no easy way to reindex them, apart from fully reindexing each affected repository, which is what this procedure is intended to avoid. A new REST endpoint could be implemented to address this (see **FECRU-3764** - Authenticate to see issue details).

Note: the following REST endpoint could be used to force a cross-repository index upgrade for a selected repository: /rest-service-fecru/admin/repositories-v1/repoX/reindex-search. There should be no need to use this, but it may be useful if something goes wrong.

Mercurial Known Issues

- CRUC-3474: If a file is removed and then another file is copied or moved over the same file within one commit, the ancestor revision is miscalculated and can result in errors in "diff to previous".
- CRUC-3470: Permission changes (and prop changes in repos converted from svn) may result in revisions that have no ancestors - subsequent changes will consider it's parent revision to be their parent revision.
- CRUC-3468: Scanning repositories converted from svn (especially using hgsubversion) can result in commits that take a long time to scan (due to the changes produced by merges from other branches).

Ordering of Branches Important When Visualising Git Changesets

FishEye 2.6 introduced the repository commit graph. The commit graph allows you to visualise changesets in their branches by showing them in configurable branch "swimlanes". One of the ways in which you can configure the commit graph is by reordering the swimlanes. Reordering swimlanes is useful for non-Git repositories, if you 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 displayed in, when a commit belongs to multiple branches.

Git Branches and Changesets in FishEye

Before considering how Git repositories are visualised in the commit graph, it is important to understand how FishEye relates Git changesets to branches.

In FishEye 2.6 and later, FishEye considers the ancestry of a Git changeset when determining which branch it is a part of. Branches can effectively be considered as pointers to changesets. Hence, merging and branching can change the branches that a changeset is considered part of.

For example, if a branch 'fisheye-2.6' is merged back to the 'master' branch, then all changesets that were seen as part of the 'fisheye-2.6' branch only will also be considered to be part of the 'master' (e.g. the changeset will be seen as part of 'master' and 'fisheye-2.6' in the activity stream).

Viewing Git Changesets and Branches in the Commit Graph

The previous section describes how a changeset can be associated with multiple branches, due to its ancestry. Instead of showing the changeset in every branch swimlane on the commit graph, FishEye represents these changesets as described below.

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

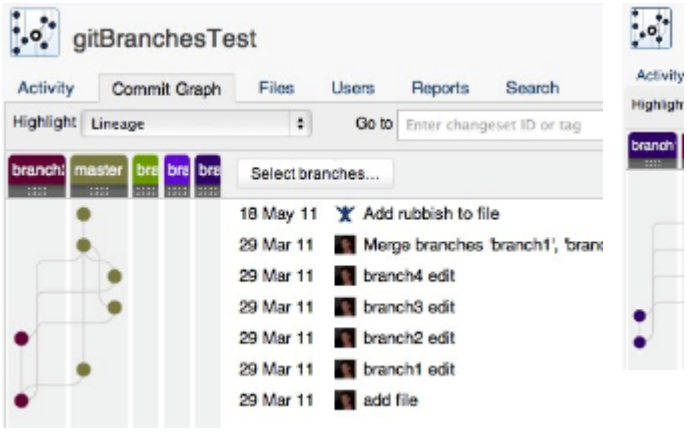
- For each swimlane, FishEye checks if the commit is in that branch. If the commit is in the branch, a dot is shown representing the commit.
- If the commit is not in the branch, the dot for the commit is moved to the next column on the right.

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.

This allows you to visually isolate changesets in the desired branches by reordering the swimlanes. For example, if you want to see the lineage of a branch, 'fisheye-2.6', but not 'fisheye-2.5' after both branches have previously been merged to 'master', you could arrange your swimlanes to 'fisheye-2.6', 'master', 'fisheye-2.5' from left to right. You will be able to see the 'fisheye-2.6' changesets and where the merge back to 'master' was made. The 'fisheye-2.5' changesets will just be seen as part of the 'master' branch.

Screenshots below: Example of how ordering swimlanes affects the branches that changesets are displayed on (click to view full-size images)

1



2

Permanent authentication for Git repositories over HTTP(S)

Currently, FishEye only supports HTTP or HTTPS for pushing and pulling from Git repositories. Git has no method of caching the user's credentials, so you need to re-enter them each time you perform a clone, push or pull.

Fortunately, there is a mechanism that allows you to specify which credentials to use for which server: the `.netrc` file.

Warning!
Git uses a utility called cURL under the covers, which respects the use of the .netrc file. Be aware that other applications that use curl to make requests to servers defined in your `.netrc` file will also now be authenticated using these credentials. Also, this method of authentication is potentially unsuitable if you are accessing your FishEye server via a proxy, as all curl requests that target a path on that proxy server will be authenticated using your `.netrc` credentials.

Warning!
cURL will not match the machine name in your .netrc if it has a username in it, so make sure you edit

your `.git/config` file in the root of your clone of the repository and remove the user and '@' part from any clone url's (url fields) that look like `https://user@machine.domain.com/...` so instead they look like `http://machine.domain.com/...`

Linux or OSX

1. Create a file called `.netrc` in your home directory (`~/.netrc`). Unfortunately, the syntax requires you to store your passwords in plain text - so make sure you modify the file permissions to make it readable only to you.
2. Add credentials to the file for the server or servers you want to store credentials for, using the format below. You may use either IP addresses or hostnames, and you **do not** need to specify a port number, even if you're running FishEye on a non-standard port.

```
machine fisheye1.mycompany.com
login myusername
password mypassword
machine fisheye2.mycompany.com
login myotherusername
password myotherpassword
```

3. And that's it! Subsequent `git clone`, `git pull` and `git push` requests will be authenticated using the credentials specified in this file.

Windows

1. Create a text file called _`netrc` in your home directory (e.g. `c:\users\kannonboy\_netrc`). Curl has problems resolving your home directory if it contains spaces in its path (e.g. `c:\Documents and Settings\kannonboy`). However, you can update your `%HOME%` environment variable to point to any old directory, so create your _`netrc` in a directory with no spaces in it (for example `c:\curl-auth\`) th en set your `%HOME%` environment variable to point to the newly created directory.
2. Add credentials to the file for the server or servers you want to store credentials for, using the format from the **Linux or OSX** section above.

Related Links

Unable to render {children}. Page not found: Repository Management FAQ.

Perforce Changesets and Branches

Why does FishEye say this changeset is on more than one branch? Why does that changeset have multiple parents?

Perforce allows a single changeset to include files on multiple branches, so FishEye marks those changesets as being on all of the branches involved.

When a changeset is on multiple branches FishEye may consider it to have multiple parents from the different branches.

FishEye does not track merges in Perforce, so merges are not shown in the graph.

Changeset branches and parents are only annotations at the changeset level --- the individual file revisions will each only have a single branch and at most one parent.

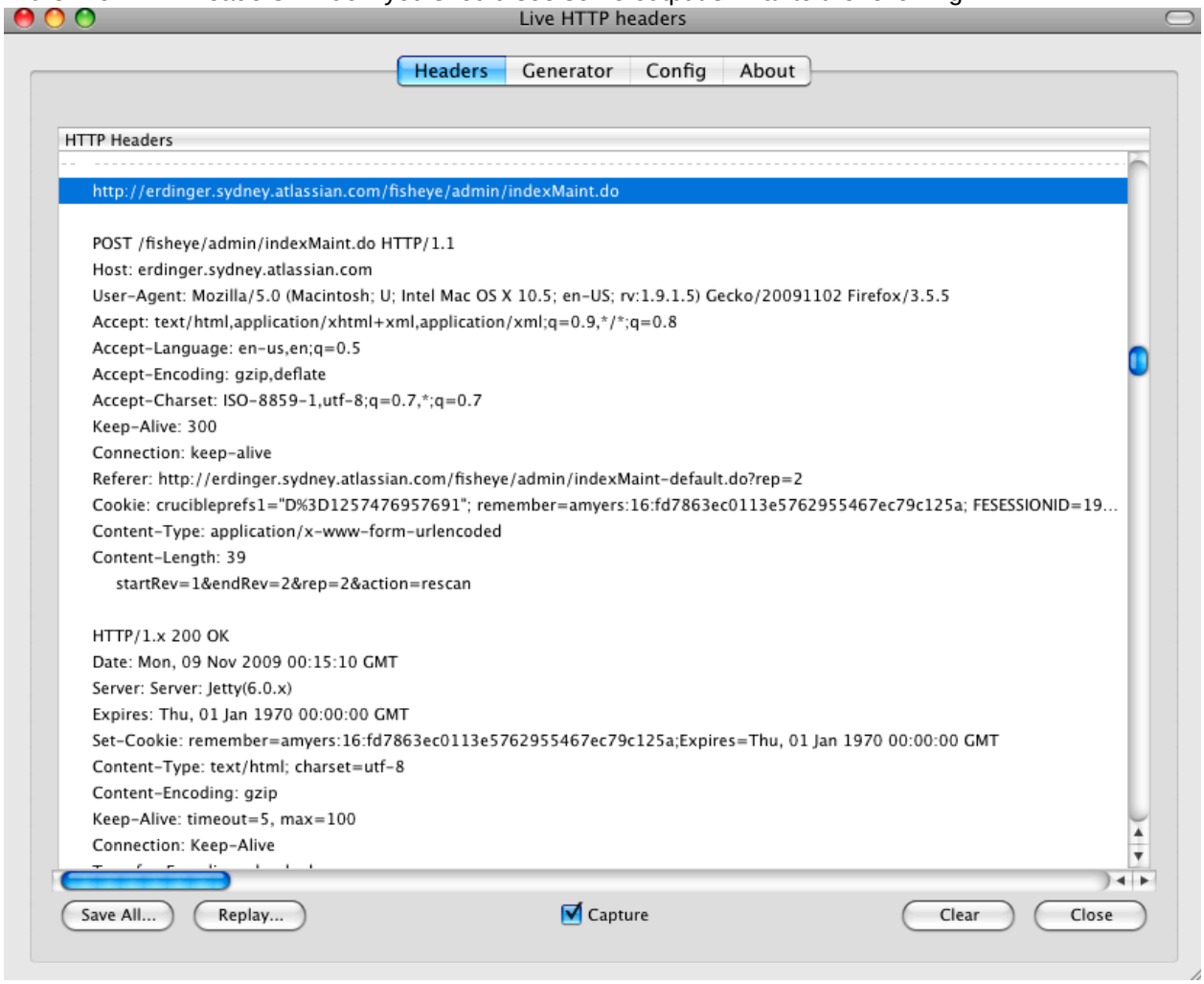
What SCM systems are supported by FishEye?

To see the list of SCM systems that is supported by FishEye, see Supported platforms.

Automating Administrative Actions in Fisheye

With some command line scripting and a tool like wget, and Live HTTP Headers for firefox you can automate actions. In this example, Fisheye will automatically rescan revision properties of an SVN when the commit message is updated to reference a new JIRA issue.

1. Enable live HTTP headers in firefox, then perform the action you want to perform automatically via the Fisheye Adminstration UI.
2. In the live HTTP headers window you should see some output similar to the following:



3. The important parts are the URL I've highlighted above (http://erdinger.sydney.atlassian.com/fisheye/admin/indexMaint.do) and any GET/POST parameters (`startRev=0&endRev=58&rep=2&action=rescan`)
4. Now we can construct a script with wget to automate this:

```
wget --keep-session-cookies --save-cookies cookie.txt
http://erdinger.sydney.atlassian.com/fisheye/admin/login.do
--post-data="origUrl=&adminPassword=admin"
wget --load-cookies cookie.txt
--post-data="startRev=0&endRev=58&rep=2&action=rescan"
http://erdinger.sydney.atlassian.com/fisheye/admin/indexMaint.do
```

With that you could generate a `post-revprop-change` hook in svn that will update the repositories automatically.

Subversion FAQ

FishEye Subversion FAQ

Configuring Start Revision based on date

For Subversion repositories Fisheye has the ability to configure a **Start Revision** parameter to allow you to only index content from a given point in your repository.

Quite often users will find it helpful to index from a revision on a given date. For example, you may want to only index SVN data in the past year. To determine the revision based on date, you can use the following command:

```
svn log -r {YYYY-MM-DD}:HEAD <SVN_URL> -l 1
```

The output of this command will the revision number closest to the date that you provide.

Errors 'SEVERE assert' or 'Checksum mismatch'

When using SVNKit, you may see errors in the FishEye log such as 'SEVERE: assert #B' or 'Checksum mismatch'.

SVNKit may have problems with older version Subversion servers - versions 1.1.x and prior. If this is the case you should either use the native JavaHL layer or upgrade your Subversion server to a more recent version.

FishEye fails to connect to the Subversion repository after a short time of successful operation.

If you use the `svn://` protocol to access a Subversion repository, you may notice that FishEye fails to connect to the repository after a short time of successful operation.

On Unix systems, the `svn://` protocol is usually handled by `inetd` or `xinetd`. These daemons apply, by default, a connection per second limit to incoming connections. Any connections above this rate are rejected by the server.

Two options for fixing this problem:

- Ask your system administrator increase the connection rate allowed for the svn connection by updating the `xinetd` configuration, or
- Specify a connection per second limit in your FishEye repository definition, to prevent FishEye from exceeding the `xinetd` limits.

How can FishEye help with merging of branches in Subversion?

FishEye gives you a logical view of your branched files so you can see activity on a single file across multiple branches/trunk.

In merge management, the main advantages of FishEye come from its search functionality. If you record the revisions merged when you check in a merge result, you can find this information in FishEye easily for the next merge operation.

As an example, let's say you have a branch `dev` created at revision `1300` from `trunk`. Development has proceeded on both `trunk` and `dev`. At some point you wish to add the latest `trunk` changes into the `dev` branch. Let's say that is at revision 1400. When you check in the results of this merge, you would use some standard format checkin comment such as:

```
merge from trunk to dev 1300:1400
```

When you come to do the next merge, say at revision 1500, you can use FishEye search to find this checkin comment and know what the starting point for the merge should be. You can then check this in as:

```
merge from trunk to dev 1400:1500
```

Merges back to `trunk` from the `dev` branch are managed in the same way.

Subversion Changeset Parents and Branches

Why do some changesets have more than one branch? Why do these changesets have more than one parent?

In Subversion, a single changeset can have files and directories that are on different branches, as defined by the SVN tag and branch structure. In this situation, the changeset is considered to be on all of the branches of its constituent file revisions. If a changeset is on more than one branch it can have a parent changeset of each of its branches, giving the changeset multiple parents.

FishEye does not track SVN merges, so merges are not indicated on the graph.

When I create a complex branch, how does FishEye determine which is its parent changeset? When I create a complex tag, how does FishEye decide which changeset to tag?

In Subversion, a *simple* branch or tag is created by copying a source directory, e.g. copying "`/trunk`" to "`/branches/branch1`" or "`/tags/tag1`". The tag or branch is considered *complex* if a part of the copied directory is replaced with another version, e.g."`/trunk`" is copied to "`/tags/tag1`", and then "`branches/branch1/dir1`" is copied to "`/tags/tag1/dir1`".

For the purpose of the commit graph, FishEye looks at where the root directory was copied from, to determine where the branch or tag originated. In the example above, the label "`tag1`" would be applied to the latest changeset on trunk when the tag was created, even though part of the tag was copied from branch1. This only affects the annotation of the changeset, not the file revisions that are tagged — the tagged file revisions are still those on `trunk` or `branch1` as appropriate.

SVN Authentication Issues

If multiple repositories have been defined in FishEye for the same SVN Server and those repositories use different credentials, FishEye may not use the correct credentials.

FishEye does not control directly when authentication information is used to access Subversion repositories. It delegates this operation to the JavaHL layer in use. JavaHL will then ask FishEye to supply credentials when required, using a callback. The default JavaHL layer shipped with FishEye, SVNKit, can cache credentials at the server level rather than at the repository level.

When experiencing this problem, FishEye can be configured to use the native JavaHL implementation, which will correctly apply the appropriate credentials.

The simplest solution is to have the same credentials for accessing the Subversion Server.

Alternatively, SVNKit can be tricked into thinking that different servers are being used. For each connection to a repository a hostname in the *hosts* file can be defined.
All these entries then point to the same IP address of the SVN Server, but to SVNKit they look like different servers, thus bypassing the problem.

Example *hosts* entries (replace the IP address with the address of the SVN Server):

```
123.45.6.78  account1
123.45.6.78  account2
```

Replace these new server names in the SVN URLs:

```
http://account1/svn/project-a/
http://account2/svn/project-b/
```

What are Subversion root and tag branches?

FishEye identifies branches and tags in your Subversion repository by applying your specified SVN tag and branch structure.

The "root:" branch

Any files or directories that fall outside the tag and branch structure are identified as being on the special branch, "`root:`". Some directories will almost always fall outside this structure. In general, root directories of branches are considered to be on the "`root:`" branch. This means that any changeset in which a branch is created is considered to be on branch, "`root:`". Additionally, any files or directories that fall outside the defined structure will be assigned branch, "`root:`". If you're seeing a lot of files and changesets on "`root:`", you may need to update your branch and tag structure in FishEye and rescan your repository, or exclude parts of your repository that don't follow your defined structure.

"tag:" branches

When FishEye detects that a tag has been created, it looks at the files that were tagged and adds the tag as an annotation to those file revisions. No file revisions are created at this point.

If a tag is modified after it has been created and committed, FishEye promotes the tag to a branch to preserve the history of the modification. For example, a user may create the tag "`build1`" by copying "`trunk`" to "`tags/build1`". If they then modify contents of `tags/build1`, a new branch "`tag:build1`" will be created for the modification.

Why do I need to describe the branch and tag structure for Subversion repositories?

In Subversion, branches and tags are defined by convention, based on their path within a repository, and not directly defined by the repository. A few different layout alternatives are commonly used. It is also possible that you are using your own custom layout. As a result you need to describe to FishEye which paths in your repository are used as branches and tags.

It is very important that you correctly define in FishEye the layout you are using. If you do not, FishEye will not know which paths represent tags and branches. This will prevent FishEye from relating different versions of the same logical file across separate paths within your repository. It will also mean that FishEye's cache will be much larger as each tagged path will be indexed separately. This will result in an increase in the initial scan time and may reduce runtime performance.

If you are having trouble using Subversion tags, see How tags work in Subversion.

Why don't all my tags show up in FishEye?

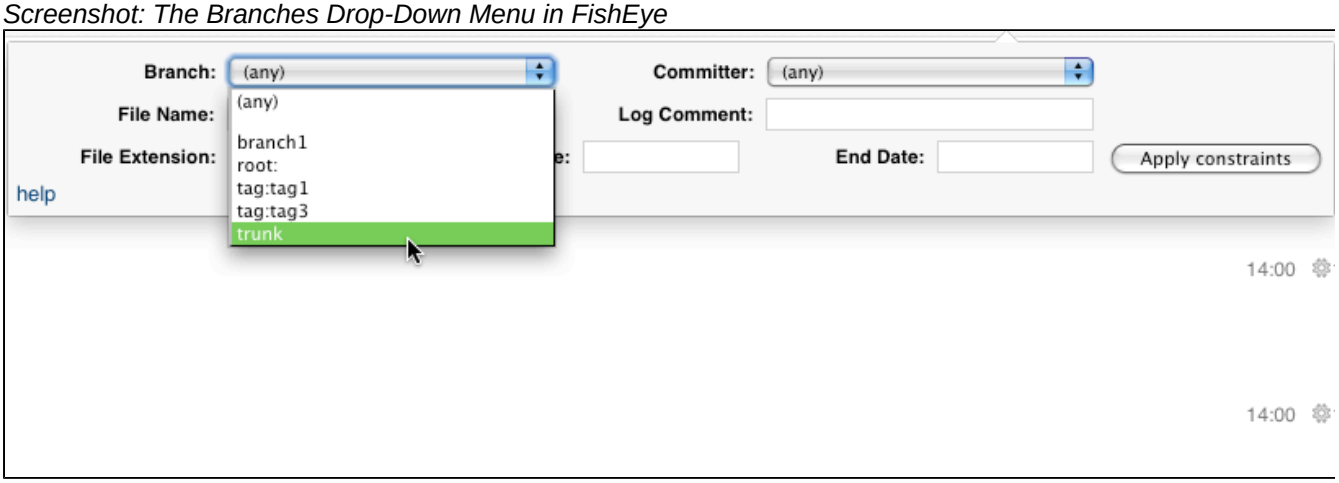
This page gives a detailed technical explanation of why certain issues affect Subversion users.

On this page:

Introduction

When accessing Subversion via FishEye, you may see references to tags in the branches drop-down menu. In the example below, we can see **tag1** and **tag3** in the drop-down menu but not **tag2**:

Screenshot: The Branches Drop-Down Menu in FishEye



In actual fact, the branches drop-down menu shows only branch names. It does not show tags, but in some instances FishEye will synthesise a branch name to record certain operations. To understand how this occurs, you will need some background knowledge on Subversion tagging (introduced in the following segments of this page).

How Subversion Processes Tags and Branches

In Subversion, tags are only a convention and are typically the result of a copy operation from the trunk to a tag area in the tags directory. When FishEye processes this copy operation, it recognises that the destination is a tag directory and tags the source file on trunk with the name of the tag.

i.e. FishEye is interpreting the Subversion copy to a tag directory as a tagging operation on the trunk files.

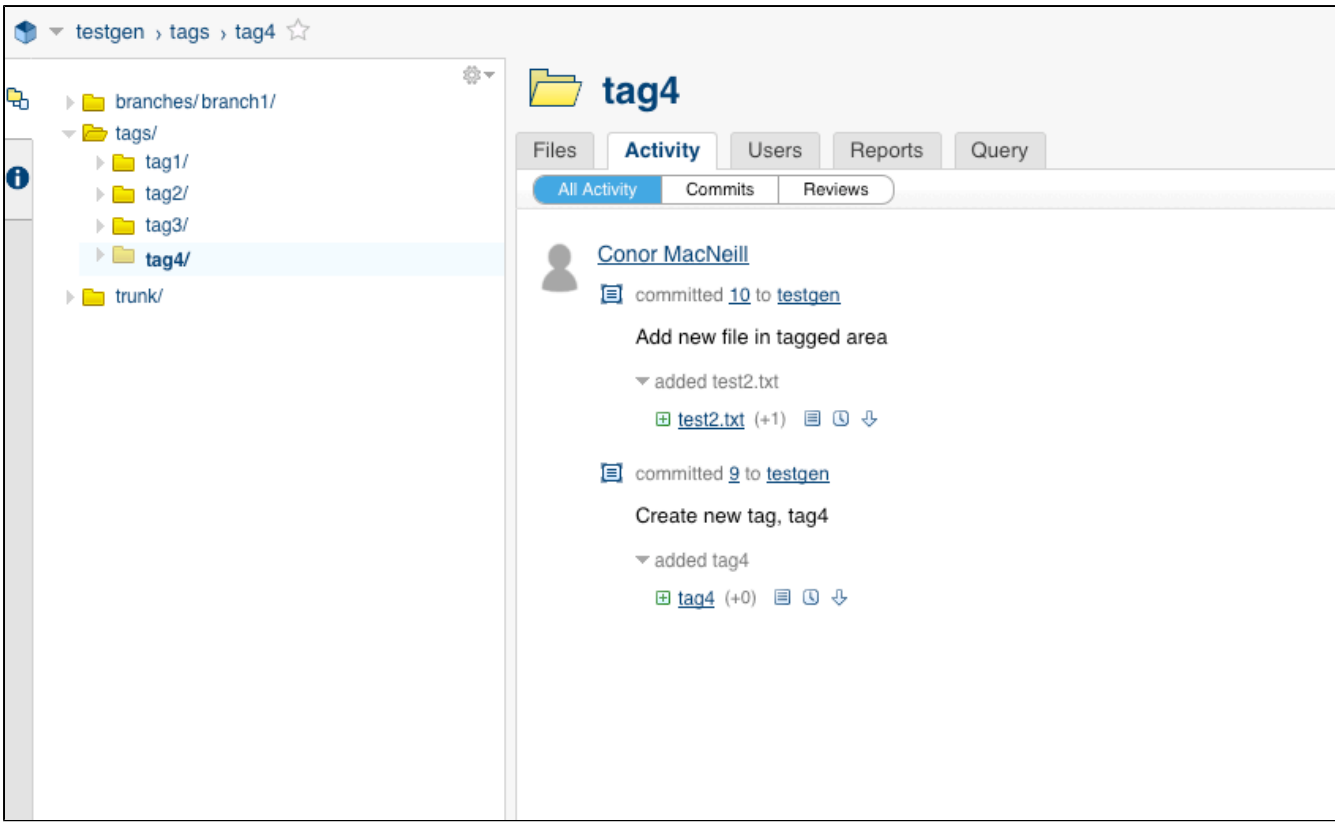
For regular changes in your Subversion repository, FishEye records each change against a branch where the change took place. If, however, after tagging, you make a change to a file in the tagged area, you are making a change outside trunk or a recognized branch. FishEye records such changes by creating an artificial branch name and associating that branch name with the change. The branch name is derived from the tag name by prepending "tag:" (in other words, the characters "tag:" appear as the first part of the name). The same thing will occur if you create a new file in the tagged area which does not come from an existing branch or trunk.

This is the reason you see some of your tags in the branch drop down. It means that for those tags, you have made a modification after the tagging operation.

An Example from a Live Subversion Repository

For example, consider **tag4** in this screenshot:

Screenshot: Subversion Tag Changes in FishEye



There are two changes here. The first creates the tag and the second adds a new file in the tagged area. This will result in the creation of an artificial branch, called "**tag:tag4**" within FishEye.

Avoid Modifications in the Tag Area

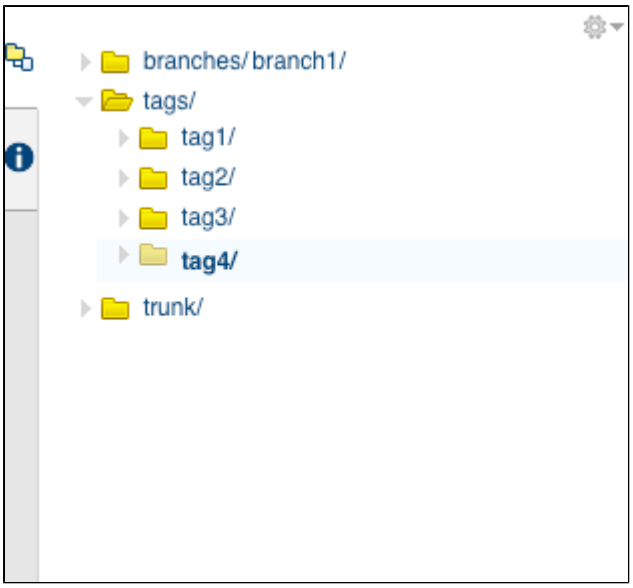
In general, it's not good practice to make changes in the tag areas of a Subversion repository. Such changes can easily get lost if they are not applied to trunk or a current branch. It is preferable to make the change in trunk or a branch and then create a new tag to capture the update. Nevertheless, since Subversion tagging is merely a convention, this is sometimes convenient. FishEye handles this situation as described above.

Conclusion

In general a lot of systems have a large number of tags which would make the drop-down unworkable. This is the reason the tag field is a text-entry box below the branch drop-down menu in FishEye.

Since tags and branches are based on location convention in Subversion, the constraint is less effective than on other SCMs. You can always see the tag or branch you are interested in, based on its location in the repository. For example, the subdirectory list here shows all tags:

Screenshot: Subdirectory Listing in FishEye



If you want to constrain to a tag, enter the tag name in the tag field of the constraint filter.

Support Policies

Welcome to the support policies index page. Here, you'll find information about how Atlassian Support can help you and how to get in touch with our helpful support engineers. Please choose the relevant page below to find out more.

- Bug Fixing Policy
- How to Report a Security Issue
- New Features Policy
- Patch Policy
- Security Advisory Publishing Policy
- Security Patch Policy
- Severity Levels for Security Issues

To request support from Atlassian, please raise a support issue in our online support system. To do this, visit support.atlassian.com, log in (creating an account if need be) and create an issue under FishEye. Our friendly support engineers will get right back to you with an answer.

Bug Fixing Policy

Summary

- Atlassian Support will help with workarounds and bug reporting.
- Critical bugs will generally be fixed in the next maintenance release.
- Non critical bugs will be scheduled according to a variety of considerations.

Raising a Bug Report

Atlassian Support is eager and happy to help verify bugs — we take pride in it! Please open a support request in our support system providing as much information as possible about how to replicate the problem you are experiencing. We will replicate the bug to verify, then lodge the report for you. We'll also try to construct workarounds if they're possible.

Customers and plugin developers are also welcome to open bug reports on our issue tracking systems directly. Use http://jira.atlassian.com for the stand-alone products and http://studio.atlassian.com for JIRA Studio and Atlassian OnDemand.

When raising a new bug, you should rate the priority of a bug according to our JIRA usage guidelines. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

How Atlassian Approaches Bug Fixing

Maintenance (bug fix) releases come out more frequently than major releases and attempt to target the most critical bugs affecting our customers. The notation for a maintenance release is the final number in the version (ie the 1 in 3.0.1).

If a bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions) then it will be fixed in the next maintenance release provided that:

- The fix is technically feasible (i.e. it doesn't require a major architectural change).
- It does not impact the quality or integrity of a product.

For non-critical bugs, the developer assigned to fixing bugs prioritises the non-critical bug according to these factors:

- How many of our supported configurations are affected by the problem.
- Whether there is an effective workaround or patch.
- How difficult the issue is to fix.
- Whether many bugs in one area can be fixed at one time.

The developers responsible for bug fixing also monitor comments on existing bugs and new bugs submitted in JIRA, so you can provide feedback in this way. We give high priority consideration to security issues.

When considering the priority of a non-critical bug we try to determine a 'value' score for a bug which takes into account the severity of the bug from the customer's perspective, how prevalent the bug is and whether roadmap features may render the bug obsolete. We combine this with a complexity score (i.e. how difficult the bug is). These two dimensions are used when developers self serve from the bug pile.

Further reading

See Atlassian Support Offerings for more support-related information.

How to Report a Security Issue

Finding and Reporting a Security Issue

If you find a security issue in the product, open an issue on https://jira.atlassian.com in the relevant project.

- Set the **security level** of the bug to '**Reporters and Developers**'.
- Set the priority of the bug to 'Blocker'.
- Provide as much information on reproducing the bug as possible.

All communication about the security issue should be performed through JIRA, so that Atlassian can keep track of the issue and get a patch out as soon as possible.

If you cannot find the right project to file your issue in, email the details to security@atlassian.com.

When reporting a security vulnerability, please keep in mind the following:

We need a technical description that allows us to assess exploitability and impact of the issue.

- Provide steps to reproduce the issue, including any URLs or code involved.
- If you are reporting a cross-site scripting (XSS), your exploit should at least pop up an alert in the browser. It is much better if the XSS exploit shows user's authentication cookie.
- For a cross-site request forgery (CSRF), use a proper CSRF case when a third party causes the logged in victim to perform an action.
- For a SQL injection, we want to see the exploit extracting database data, not just producing an error message.
- HTTP request / response captures or simply packet captures are also very useful to us.

Please refrain from sending us links to non-Atlassian web sites, or reports in PDF / DOC / EXE files. Image files are ok. Make sure the bug is exploitable by someone other than the user himself (e.g. "self-XSS").

Without this information it is not possible to assess your report and it is unlikely to be addressed.

We are not looking for the reports listing generic "best practice" issues such as:

- Specific cookies being not marked as Secure or HTTPOnly
- Presence or absence of HTTP headers (X-Frame-Options, HSTS, CSP, nosniff and so on)
- Clickjacking
- Mixed HTTP and HTTPS content
- Auto-complete enabled or disabled
- SSL-related issues

We are also not looking for reports on the following bug classes:

- Username enumeration using login or password reset features. While username enumeration can be a vulnerability in web applications, most of Atlassian products and web sites include a number of social features. As a result, usernames can be discovered by design in a number of ways.

Further reading

See Atlassian Support Offerings for more support-related information.

New Features Policy

Summary

- We encourage and display customer comments and votes openly in our issue tracking system, http://jira.atlassian.com.
- We do not publish roadmaps.
- Product Managers review our most popular voted issues on a regular basis.
- We schedule features based on a variety of factors.
- Our Atlassian Bug Fixing Policy is distinct from our Feature Request process.
- Atlassian provides consistent updates on the top 20 feature/improvement requests (in our issue tracker systems).

How to Track what Features are Being Implemented

When a new feature or improvement is scheduled, the 'fix-for' version will be indicated in the JIRA issue. This happens for the upcoming release only. We maintain roadmaps for more distant releases internally, but because these roadmaps are often pre-empted by changing customer demands, we do not publish them.

How Atlassian Chooses What to Implement

In every major release we *aim* to implement highly requested features, but it is not the only determining factor. Other factors include:

- **Customer contact**: We get the chance to meet customers and hear their successes and challenges at Atlassian Summit, Atlassian Unite, developer conferences, and road shows.
- **Customer interviews**: All product managers at Atlassian do customer interviews. Our interviews are not simply to capture a list of features, but to understand our customers' goals and plans.
- **Community forums**: There are large volumes of posts on answers, of votes and comments on jira.atlassian.com, and of conversations on community forums like groups on LinkedIn.
- **Customer Support**: Our support team provides clear insights into the issues that are challenging for customers, and which are generating the most calls to support
- **Atlassian Experts**: Our Experts provide insights into real-world customer deployments, especially for customers at scale.
- **Evaluator Feedback**: When someone new tries our products, we want to know what they liked and disliked and often reach out to them for more detail.
- **In product feedback**: The JIRA Issue Collectors that we embed our products for evaluators and our Early Access Program give us a constant pulse on how users are experiencing our product.
- **Usage data:** Are customers using the features we have developed?
- **Product strategy:** Our long-term strategic vision for the product.

How to Contribute to Feature Development

Influencing Atlassian's release cycle
We encourage our customers to vote on feature requests in JIRA. The current tally of votes is available online in our issue tracking system, http://jira.atlassian.com. Find out if your improvement request already exists. If it does, please vote for it. If you do not find it, create a new feature or improvement request online.

Extending Atlassian Products
Atlassian products have powerful and flexible extension APIs. If you would like to see a particular feature implemented, it may be possible to develop the feature as a plugin. Documentation regarding the plugin APIs is available. Advice on extending either product may be available on the user mailing-lists, or at Atlassian Answers.

If you require significant customisations, you may wish to get in touch with our partners. They specialise in extending Atlassian products and can do this work for you. If you are interested, please contact us.

Further reading

See Atlassian Support Offerings for more support-related information.

Patch Policy

Patch Policy

Atlassian will only provide software patches in extremely unusual circumstances. If a problem has been fixed in a newer release of the product, Atlassian will request that you upgrade your instance to fix the issue. If it is deemed necessary to provide a patch, a patch will be provided for the current release and the last maintenance release of the last major version only.

Patches are issued under the following conditions:

- The bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions) AND a patch is technically feasible (i.e., it doesn't require a major architectural change)
  OR
- The issue is a security issue, and falls under our Security Patch Policy.

Atlassian does not provide patches for non-critical bugs.

Provided that a patch does not impact the quality or integrity of a product, Atlassian will ensure that patches supplied to customers are added to the next maintenance release. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

Patches are generally attached to the relevant http://jira.atlassian.com issue.

Further reading

See Atlassian Support Offerings for more support-related information.

Security Advisory Publishing Policy

Publication of Security Advisories

When a critical severity security vulnerability in an Atlassian product is discovered and resolved, Atlassian will inform customers through the following mechanisms:

- We will post a security advisory in the latest documentation of the affected product at the same time as releasing a fix for the vulnerability.
- We will send a copy of all posted security advisories to the **'Technical Alerts' mailing list** for the product concerned.
  Note: To manage your email subscriptions and ensure you are on this list, please go to my.atlassian.com and click 'Communications Centre' near the top right of the page.
- If the person who reported the vulnerability wants to publish an advisory through some other agency, such as CERT, we will assist in the production of that advisory and link to it from our own.

If you want to track non-critical severity security vulnerabilities, you need to monitor the issue trackers for the relevant products on http://jira.atlassian.com. For example, https://jira.atlassian.com/browse/JRA for JIRA and https://jira.atlassian.com/browse/CONF for Confluence. Security issues in trackers will be marked with a "security" label. All security issues will be listed in the release notes of the release where they have been fixed, similar to other bugs.

One of the ways to monitor updates to security issues is subscribing to the results of a sample search via email or RSS.

Further reading

See Atlassian Support Offerings for more support-related information.

Security Patch Policy

Product Security Patch Policy

Atlassian makes it a priority to ensure that customers' systems cannot be compromised by exploiting vulnerabilities in Atlassian products.

Scope

This page describes when and how we release security patches and security upgrades for our products. It does not describe the whole of disclosure process that we follow. This policy excludes OnDemand and Bitbucket, since these services are always patched by Atlassian without additional notifications.

Critical vulnerabilities

When a **Critical** security vulnerability is discovered by Atlassian or reported by a third party, Atlassian will do all of the following:

- Issue a new, fixed release for the current version of the affected product as soon as possible, usually in a few days.
- Issue a binary patch for the current release.
- Issue a binary patch for the latest maintenance release of the previous version of the product.
- Patches for older versions or releases normally will not be issued.

Patches will be attached to the relevant JIRA issue. You can use these patches as a "stop-gap" measure until you upgrade your installation in order to fully fix the vulnerability.

Non-critical vulnerabilities

When a security issue of a **High, Medium or Low** severity is discovered, Atlassian will do all of the following:

- Include the fix into the next scheduled release, both for the current and previous maintenance versions.
- Where practical, provide new versions of plugins or other components of the product that can be upgraded independently.

You should upgrade your installation in order to fix the vulnerability.

Other information

Severity level of vulnerabilities is calculated based on Severity Levels for Security Issues.

Visit our general Atlassian Patch Policy as well.

Examples

Example 1: A critical severity vulnerability is found in a (hypothetical current release) JIRA 5.3.2. The last bugfix release in 5.2.x branch was 5.2.3. In this case, a patch will be created for 5.3.2 and 5.2.3. In addition, new bugfix releases, 5.3.3 and 5.2.4, which are free from this vulnerability, will be created in a few days.

Example 2: A high or medium severity vulnerability is found in the same release as in the previous example. The fix will be included into the currently scheduled releases 5.3.3 and 5.2.4. Release schedule will not be brought forward and no patches will be issued. If the vulnerability is in a plugin module, then a plugin upgrade package may still be supplied.

Further reading

See Atlassian Support Offerings for more support-related information.

Severity Levels for Security Issues

Severity Levels

Atlassian security advisories include a severity level. This severity level is based on our self-calculated CVSS score for each specific vulnerability. CVSS is an industry standard vulnerability metric. You can learn more about CVSS at FIRST.org web site.

CVSS scores are mapped into the following severity ratings:

- Critical
- High
- Medium
- Low

An approximate mapping guideline is as follows:

CVSS score range	Severity in advisory
0 – 2.9	Low
3 – 5.9	Medium
6.0 – 7.9	High
8.0 – 10.0	Critical

Below is a summary of the factors which illustrate types of vulnerabilities usually resulting in a specific severity level. Please keep in mind that this rating does not take into account details of your installation.

Severity Level: Critical

Vulnerabilities that score in the critical range usually have most of the following characteristics:

- Exploitation of the vulnerability results in root-level compromise of servers or infrastructure devices.
- The information required in order to exploit the vulnerability, such as example code, is widely available to attackers.
- Exploitation is usually straightforward, in the sense that the attacker does not need any special authentication credentials or knowledge about individual victims, and does not need to persuade a target user, for example via social engineering, into performing any special functions.

For critical vulnerabilities, is advised that you patch or upgrade as soon as possible, unless you have other mitigating measures in place. For example, if your installation is not accessible from the Internet, this may be a mitigating factor.

Severity Level: High

Vulnerabilities that score in the high range usually have some of the following characteristics:

- The vulnerability is difficult to exploit.
- Exploitation does not result in elevated privileges.
- Exploitation does not result in a significant data loss.

Severity Level: Medium

Vulnerabilities that score in the medium range usually have some of the following characteristics:

- Denial of service vulnerabilities that are difficult to set up.
- Exploits that require an attacker to reside on the same local network as the victim.
- Vulnerabilities that affect only nonstandard configurations or obscure applications.
- Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics.
- Vulnerabilities where exploitation provides only very limited access.

Severity Level: Low

Vulnerabilities in the low range typically have very little impact on an organisation's business. Exploitation of such vulnerabilities usually requires local or physical system access.

Further reading

See Atlassian Support Offerings for more support-related information.

FishEye Resources

Resources for Evaluators

- Free Trial
- Feature Tour

Resources for Administrators

- FishEye Knowledge Base
- FishEye FAQ
- Guide to Installing an Atlassian Integrated Suite
- The big list of Atlassian gadgets

Downloadable Documentation

- FishEye documentation in PDF, HTML or XML formats

Plugins

- FishEye Developer Documentation
- Add-ons for FishEye

Support

- Atlassian Support
- Support Policies

Forums

- FishEye Forum
- FishEye Developers Forum

Mailing Lists

- Visit http://my.atlassian.com to sign up for mailing lists relating to Atlassian products, such as technical alerts, product announcements and developer updates.

Feature Requests

- Issue Tracker and Feature Requests for FishEye