Subversion ALM

Painless and powerful ALM integration of JIRA with Subversion

©Kinto Soft Ltd

Contents

©Kinto Soft Ltd

©Kinto Soft Ltd

Introduction

It provides a painless and powerful integration of JIRA with Subversion. All the web applications required in order to enhance Subversion traceability are bundled with the plugin. They have been fully customized to support the JIRA plugin environment and fully integrated out of the box. The bundled applications are:

- Atlassian JIRA Subversion plugin
- Polarion Web Client for Subversion

With Subversion ALM, JIRA users are able to trace relationships among issues and commits from a single point, explore Subversion repositories and create powerful reports on JIRA by using JQL fully integrated with Subversion.

Major Features

- Painless installation and configuration
- Multi domain repository (Subversion repositories can be located anywhere)
- Multi connection protocol: file, http, https, svn and svn+ssh
- Displays commits related to issues and projects on JIRA
- Bundled a customized Polarion's Web Client for Subversion fully integrated out of the box
- Full Commit Graphs providing the backward and forward history of any artifact with support for JIRA issues
- Real time Subversion statistics for users and artifacts (commit calendar views)
- JQL Functions supporting Subversion attributes
- Internal Office and PDF HTML 5viewer

©Kinto Soft Ltd

Installation & Configuration

Install the add-on jar file

It supports the Atlassian V2 plugin model, hence no JIRA re-start is required. The plug-in is available to download from the Marketplace. It can be installed by following any of the usual methods:

- From within JIRA (Administration > add-ons > Find new add-ons)
- By downloading the binaries (.jar file) from the Marketplace and uploading it into JIRA: Administration > Manage > Installed > Upload a new add-on

Compatibility

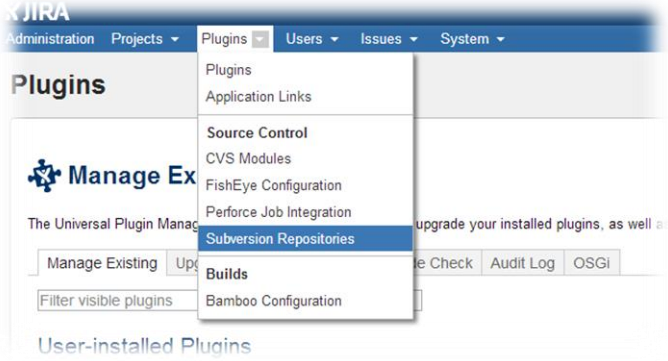
It is fully compatible with the Atlassian's JIRA Subversion plug-in as well as Fisheye. As Subversion ALM includes a improved version of the Atlassian's JIRA Subversion plug-in it has not sense keep them installed on the same JIRA instance unless during the evaluation period. Regarding Fisheye, you would get a lot of features overlapping, but while FishEye puts the focus on reposting, Subversion ALM put the focus on traceability.

Accessing to the configuration page

The Subversion ALM configuration page on JIRA requires *Administration* permission.

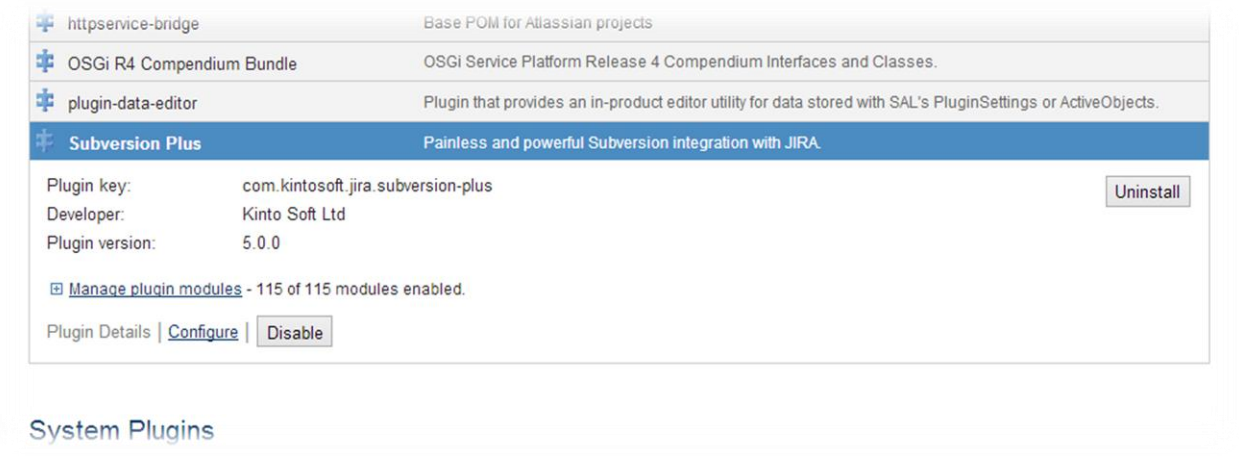
The configuration page can be accessed from two locations:

A) The Administration > Add-ons > Source Control > *Subversion Repositories* menu:



From the Administration > add-ons > manage > Installed > *Subversion ALM* > *Configure* link:

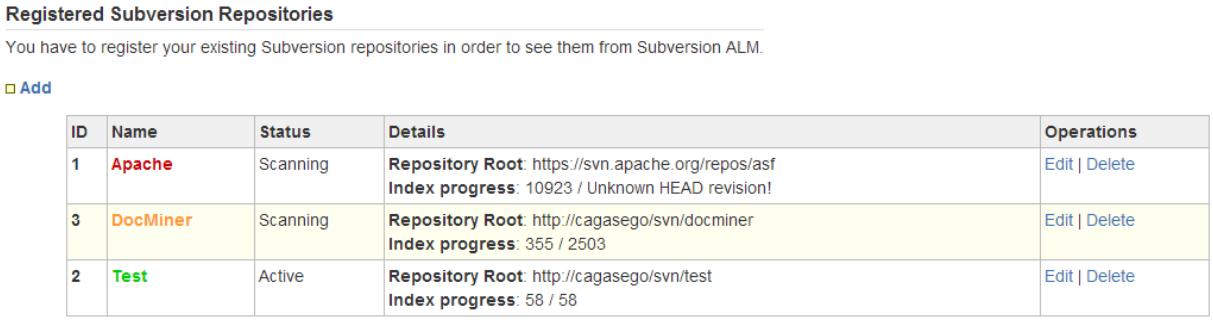
©Kinto Soft Ltd



©Kinto Soft Ltd

Registering Subversion repositories

Subversion ALM supports any Subversion 5.0+ repository version and all the connection protocols (file, http, https, svn, svn+ssh). The repositories have to be registered in the *Registered Subversion Repositories* panel at the bottom of the configuration page on JIRA. The table lists the already registered repositories:



Each repository has:

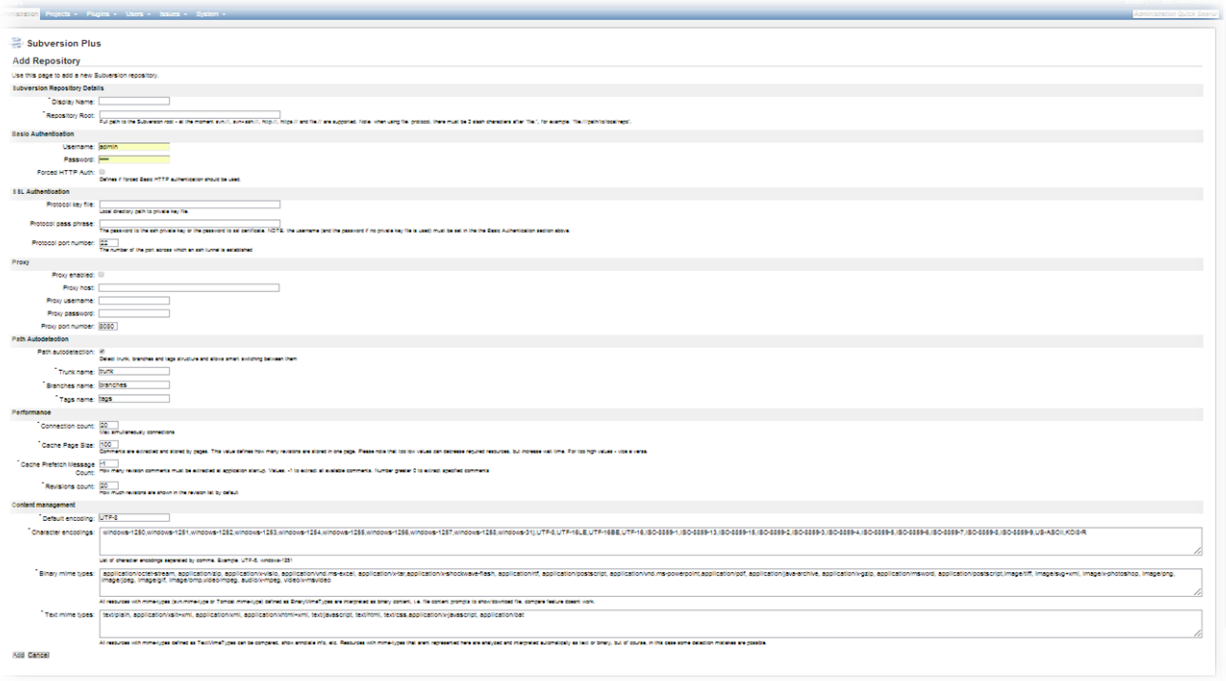
- **ID**: an unique integer > 0 assigned by the database to each repository automatically.
- **Name**: it will be displayed at the GUI front end to help users to easily identify the repository (so, please, use user friendly names). The color name varies accordingly the indexing progress:
  - ***Red:*** *Error*.
  - ***Orange:*** *In progress*, the *latest indexed* revision is behind the repository HEAD revision.
  - ***Green*****:** *Up to date*.
- **Status**:
  - ***Scanning*****:** There is a thread fetching the log history for the repository and the repository data are being cached on the database.
  - ***Active*****:** No error. The repository is accessible but there is not a thread scanning it because it is up to date or queued (see *MAX_INDEXING_THREADS* parameter).
  - ***Inactive*****:** Some error has occurred and the repository is not accessible.
- ***Details***:
  - **Repository Root**
  - **Index progress**: latest indexed revision vs total revisions in the repository (HEAD).

The *Operation* columns show the available actions for each repository. Usually they are:

- *Edit*: allows changing most of the repository configuration data.
- *Delete:* It un-registers the repository and deletes all its data from the H2 database.

To register a new Subversion repository click on the *Add* link. It will open a new page asking for the repository configuration data:

©Kinto Soft Ltd



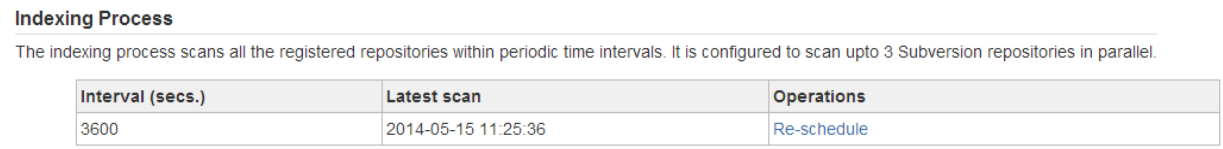
Most of the configuration is automatically populate with default values. Hence, the amount of the required data to be filled is really small: the repository *name*, the *url*, username *credentials* and the connection *protocol* only in most cases.

©Kinto Soft Ltd

The Indexing Process

The registered repositories are periodically scanned in order to index the new commits. Subversion ALM will scan each repository incrementally, since its latest indexed commit. The maximum amount of parallel indexing repositories can be configured by changing the MAX_INDEXING_THREADS value at the APPLICATION_PROPERTIES table on the internal database. The default value is 3.

The indexing interval can be scheduled from the *Indexing Process* section:

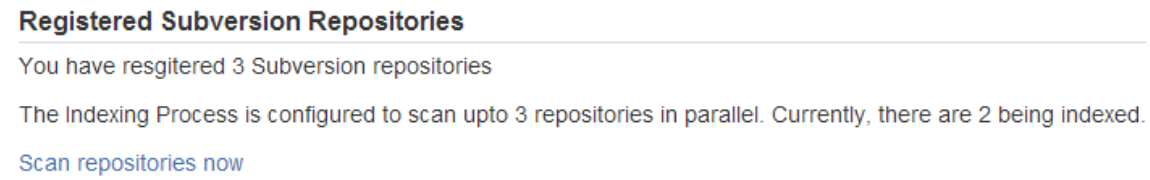


The default value is 3.600 seconds (1 hour) and the minimal supported value is 10 seconds.

The indexing interval can be modified by clicking on the *Re-schedule* link. This will open a new page:

The new interval value has to be in second units.

All the indexing information has been moved into the *Subversion Registered Repositories* section:

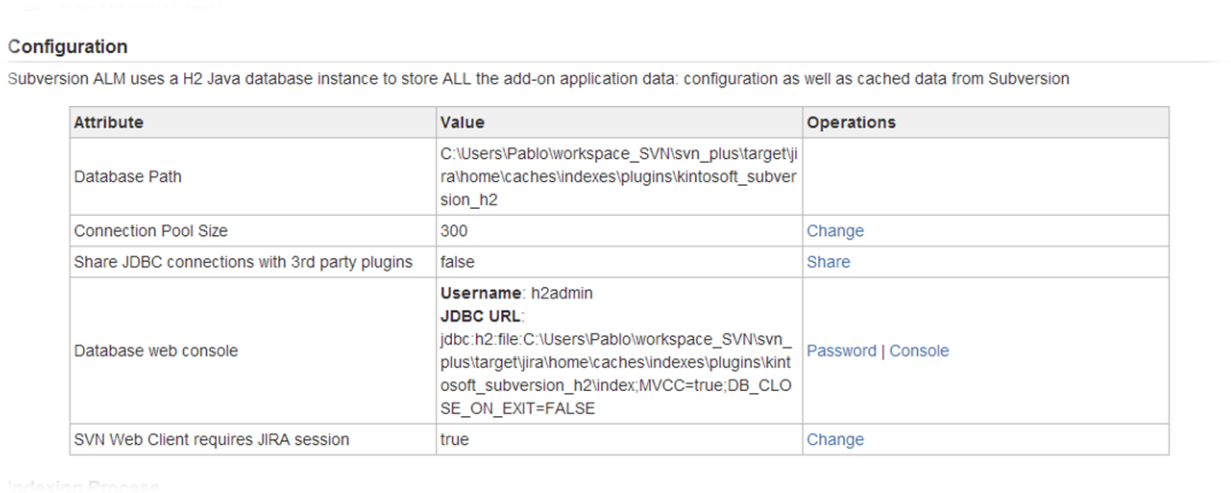


The *Scan repositories now* link is dynamic and it will displayed only when it has sense to start the Indexing Process manually without waiting for be fired by the scheduler

©Kinto Soft Ltd

Configuration parameters

All the application data, including the plugin's own configuration, the repositories configuration as well as any data retrieved from Subversion, are stored in the H2 database instance. The database location is shown in *The Internal Database* section > *Database Path* entry.



Configuration

Subversion ALM uses a H2 Java database instance to store ALL the add-on application data: configuration as well as cached data from Subversion

Attribute	Value	Operations
Database Path	C:\Users\Pablo\workspace_SVN\svn_plus\target\jira\home\caches\indexes\plugins\kintosoft_subversion_h2	
Connection Pool Size	300	Change
Share JDBC connections with 3rd party plugins	false	Share
Database web console	**Username**: h2admin **JDBC URL**: jdbc:h2:file:C:\Users\Pablo\workspace_SVN\svn_plus\target\jira\home\caches\indexes\plugins\kintosoft_subversion_h2\index;MVCC=true;DB_CLOSE_ON_EXIT=FALSE	Password | Console
SVN Web Client requires JIRA session	true	Change

Re-initializing the Subversion ALM configuration

The H2 database instance can be deleted to fully reset Subversion ALM configuration:

1. Disable the plug-in
2. Delete the *Database Path* directory
3. Enable the plug-in

By following the steps above, a new clean database would be created. The entire previous configuration as well the Subversion indexed data will be permanently lost.

Exporting and importing the registered repositories

Before deleting the database files, you might want to export all the registered repositories in order to import them after resetting the configuration.

Export to CVS

You have to export the following tables:

- REPOSITORIES
- REPOSITORIES_CONFIGURATION

By running the SLQ commands below from the database web console:

- CALL CSVWRITE('repos.csv', 'SELECT * FROM REPOSITORIES')
- CALL CSVWRITE('repos_conf.csv', 'SELECT * FROM REPOSITORIES_CONFIGURATION')

NOTE: The Subversion credentials will be exported in plain text. Be careful with the repos.csv file and keep it in a SECURE location.

©Kinto Soft Ltd

Import from CVS

Start the plug-in, this will create an empty database schema (tables, indexes, etc.). Then run:

- INSERT INTO REPOSITORIES (select * from csvread('repos.csv'));
- INSERT INTO REPOSITORIES_CONFIGURATION (select * from csvread('repos_conf.csv'));

And re-start the plug-in.

The Subversion timeouts

Sometimes login into a Subversion repository or fetching the commits might take a long time. By default, the Connection and Read timeouts are two minutes (120000 milliseconds) but you might want to change them:

NOTE: The add-on must be re-started after modifying any timeout.

The pool size

By default the plug-in creates 100 database connections during the start up. The code has also been written to ensure that each user action consumes one database connection only. If no free connections are available the add-on will wait for ten seconds before raise an error. If during that time interval some connection is released, then it will be assigned to some process (user action) waiting for a connection. Depending on your user base, it might require increasing the amount of available connections.

You can increase the *Connection Pool Size* value by clicking on the *Change* action:

The Database Web Console

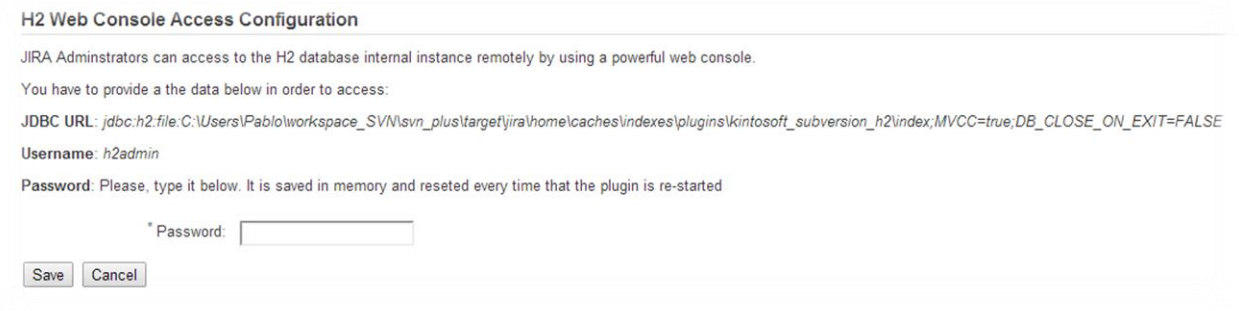
The H2 database is automatically started and stopped by Subversion ALM. It is opened in the *embedded* mode and the database file is locked while Subversion ALM is enabled: no any external process can access/open the H2 database.

The database can be accessed remotely through the *Database Web Console*. It requires JIRA Administer privileges.

The Database Web Console allows to explore the data model (tables) as well as read/modify any data. Please, use it with caution.

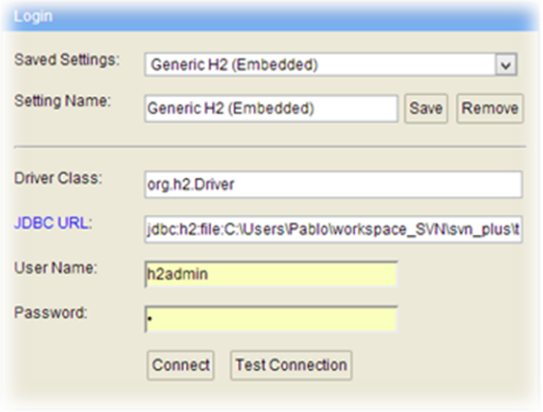
©Kinto Soft Ltd

Prior to access to the web console, setting a password is required. Click on the *Password* action and provide a valid one:



Then save it and copy the JDBC URL into the system clipboard because it will be required in next steps. Also please pay attention to the *h2admin* username.

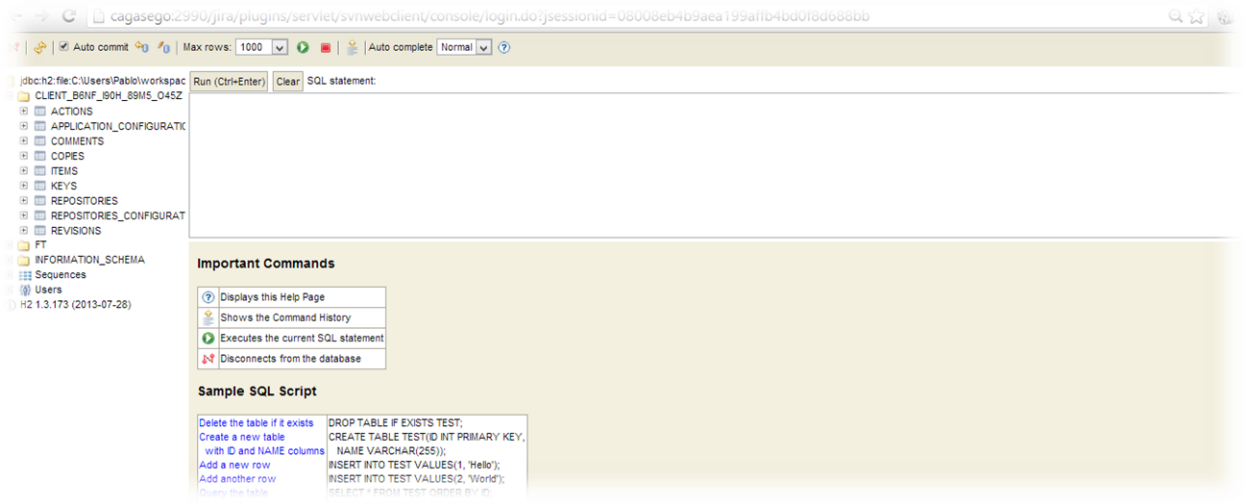
JIRA administrators can click on the *Console* link to open the Web Console:



And fill out the required connection data:

Type the *JDBC URL* from and the user credentials: fill the *User Name* with the *h2admin* default value and also fill the *Password* field with the value that you introduced in the previous steps. Click on the *Connect* button to access to the H2 database:

©Kinto Soft Ltd



The console allows exploring the database data and **modifying, inserting and deleting** new records. It is really powerful as you can use it to fully administer your H2 database instance.

Compacting the database file size

A new parameter is supported to allow compacting the database file size on database shutdown (upgrading, disabling and deleting repositories):

Compact database on plugin exit (it may take a long time and it should enabled only from time to time for maintenence)	false	Switch

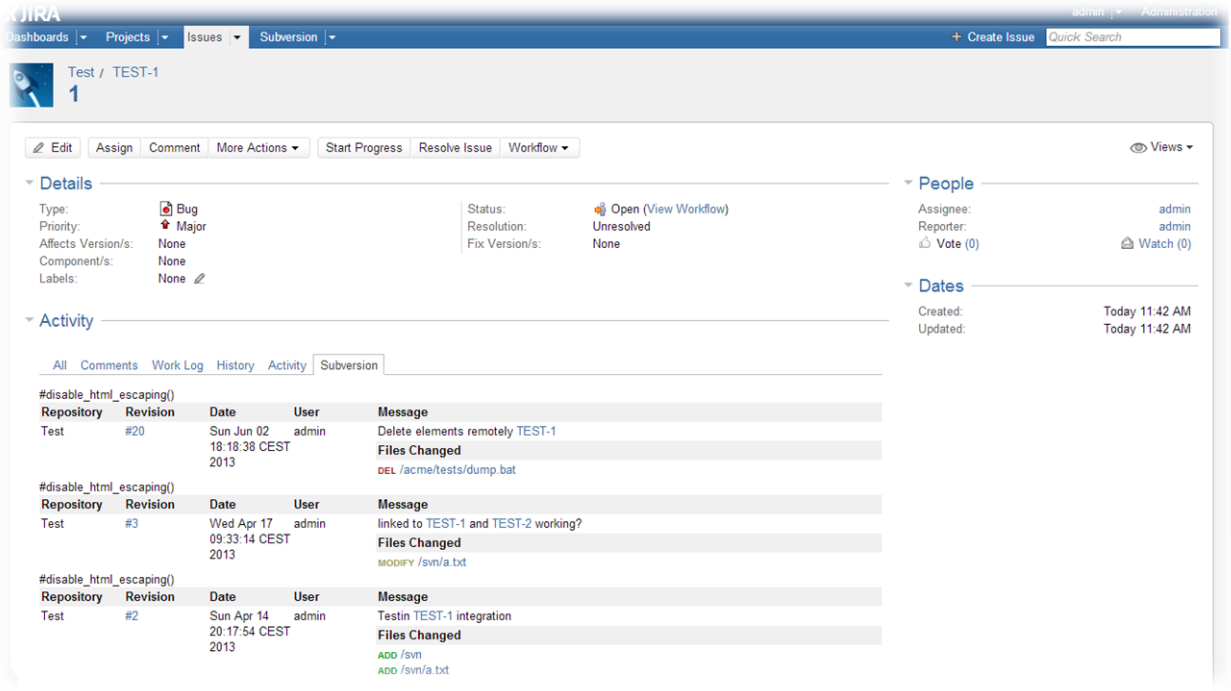
The default value is *false* and usually it is not necessary compacting the database files. However, you may want to enable it from time to time and set it to false (default value) after re-enabling the plugin as compacting rebuilds the database indexes and it could take a long time.

©Kinto Soft Ltd

Displaying Subversion commits on JIRA

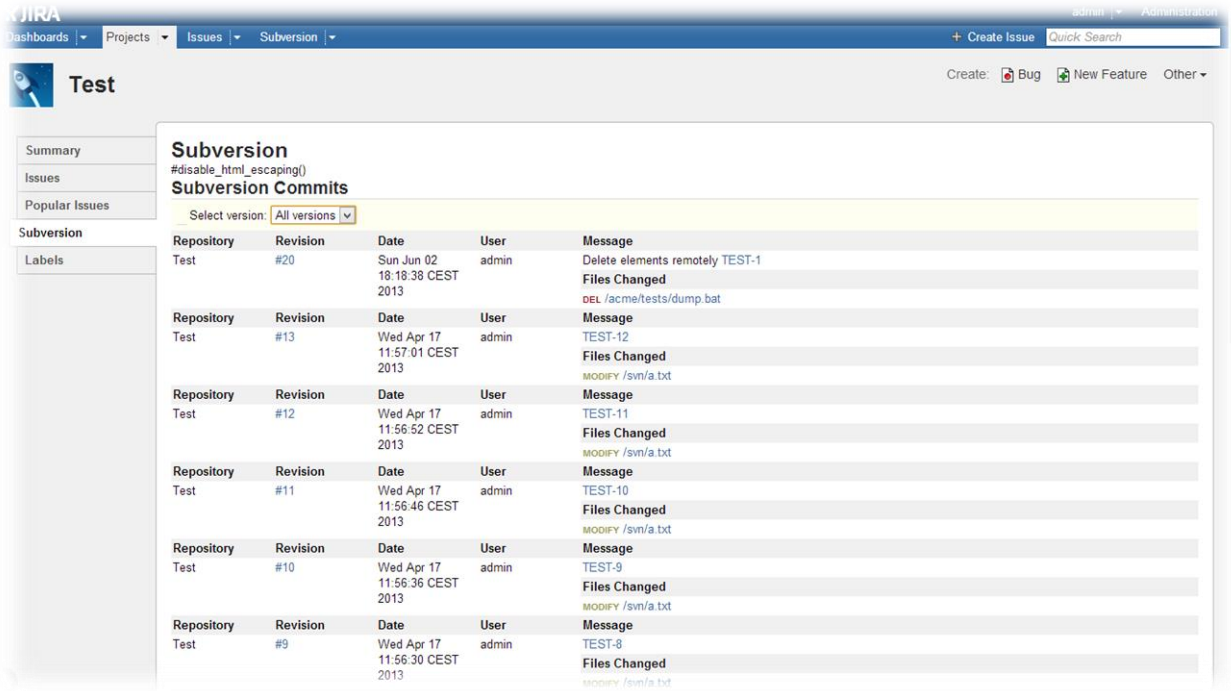
The indexing process scans the repositories looking for JIRA issue keys embedded in commit messages and stores those relationships in the H2 database. They can be explored from the JIRA from two locations:

A) *The issue page*: Subversion ALM adds a new *Subversion* tab in the JIRA issue page. It lists the commits related to the issue:



B) *The project page*: It also adds a *Subversion* tab on the JIRA project page:

©Kinto Soft Ltd



In this case, a combo box is displayed at the top of the view allowing users to filter by project *version*.

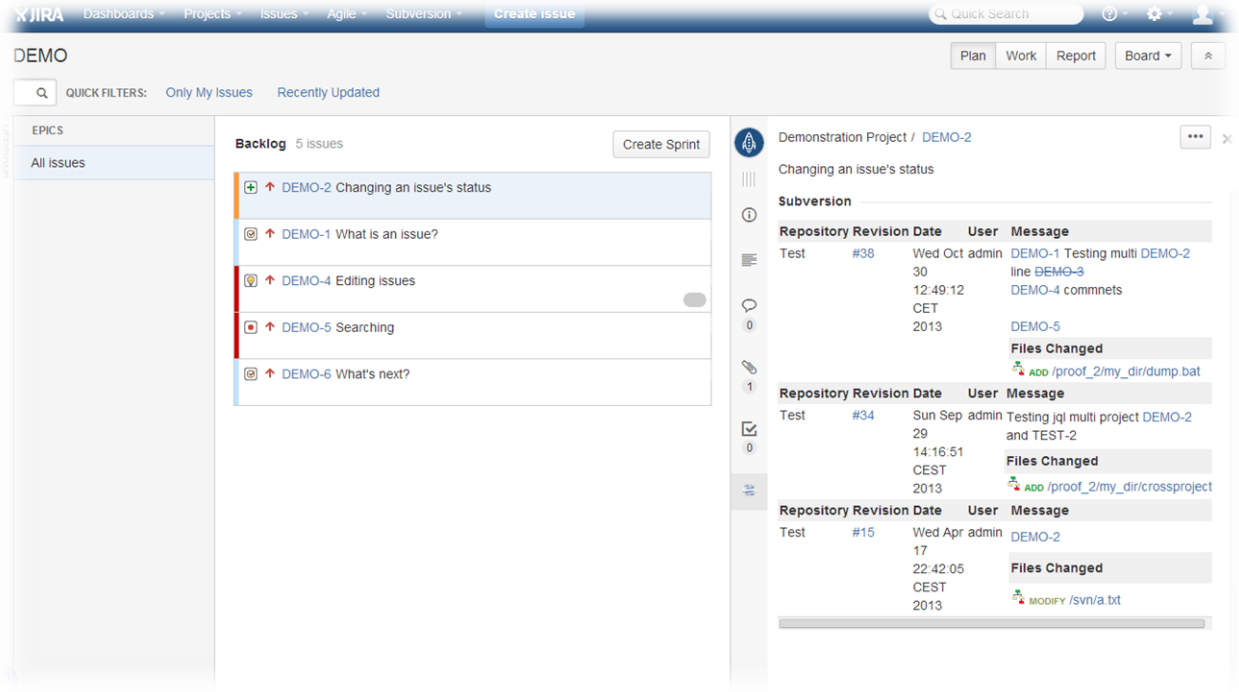
IMPORTANT: Those views require that the user is granted with the **View Development Tools**, ***View Issue Source Tab*** or ***View Version Control*** permission accordingly to your JIRA version.

IMPORTANT2: The permissions above have been discontinued in JIRA 7. As there is no chance to verify them, the add-on ignores them.

©Kinto Soft Ltd

Displaying Subversion commits on Agile (Greenhopper)

Since the 5.8.0 version, the 10 latest commits are displayed on the JIR Agile Scrum Plan and Kanban Work boards:

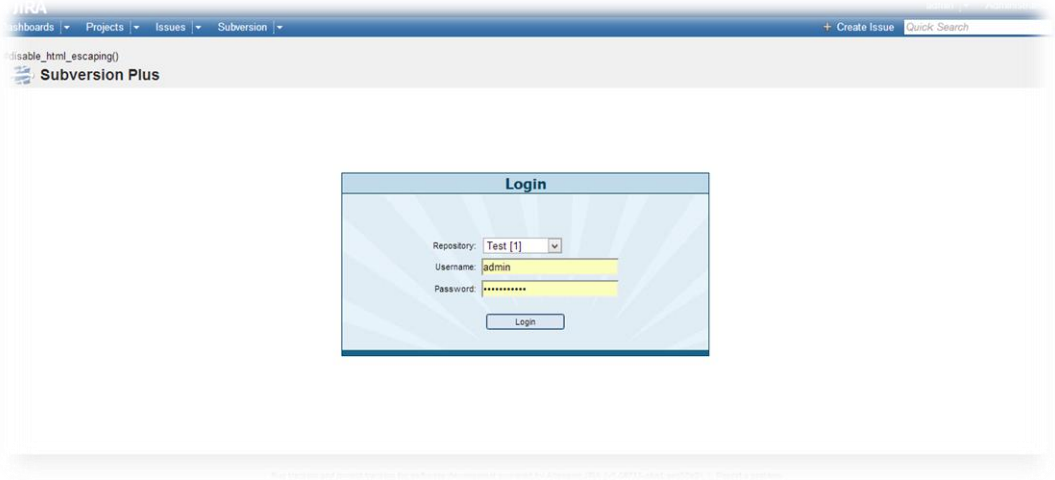


©Kinto Soft Ltd

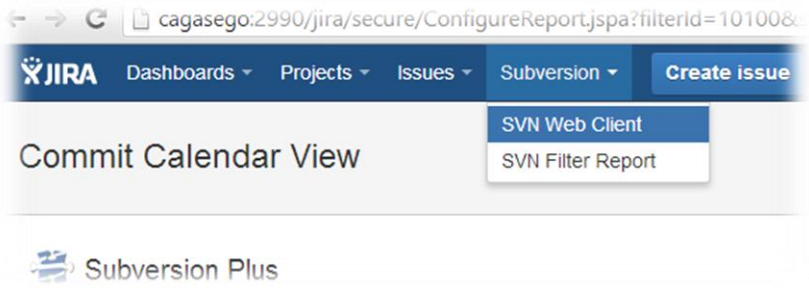
The Web Client for Subversion

JIRA integration

Since the 2.0 version, a customized version of the *Polarion Subversion Web Client* is bundled with the plug-in. It is automatically deployed and configured by Subversion ALM to *work* fully integrated with JIRA *out-of-the-box*:



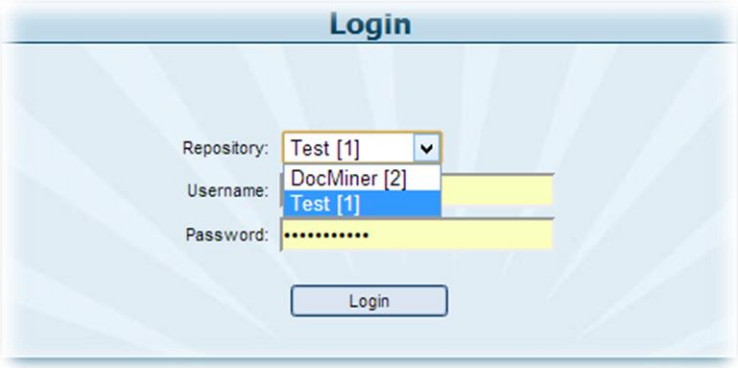
The Web Client is a full web application and it can be accessed from the Subversion menu on the JIRA top action bar:



The *Subversion* top menu is displayed if the user has the enough privileges (just the same required to show the Subversion tab on the issue and project pages).

The users can log out from the Subversion Web Client at any time by clicking on the *logout* link at the top right of the view. It will show the login view and the users can choose any registered repository from the *Repository* combo box:

©Kinto Soft Ltd



The combo box is populated by Subversion ALM with the names of all the registered repositories.

The repository Id is shown beside the repository name. This is very useful in order to use the JQL functions as many of them required the repository Id as input parameter.

Subversion Web Client in *Standalone* mode

By default, all the users have to login JIRA prior browsing on Subversion. This behavior can be changed from the plugin configuration view:

Attribute	Value	Operations
Database Path	C:\Users\Pablo\workspace_SVN\svn_plus\target\jira\home\caches\indexes\plugins\kintosoft_subversion_h2	
Connection Pool Size	300	Change
Share JDBC connections with 3rd party plugins	false	Share
Database web console	**Username**: h2admin **JDBC URL**: jdbc:h2:file:C:\Users\Pablo\workspace_SVN\svn_plus\target\jira\home\caches\indexes\plugins\kintosoft_subversion_h2\index;MVCC=true;DB_CLOSE_ON_EXIT=FALSE	Password | Console
SVN Web Client requires JIRA session	true	Change

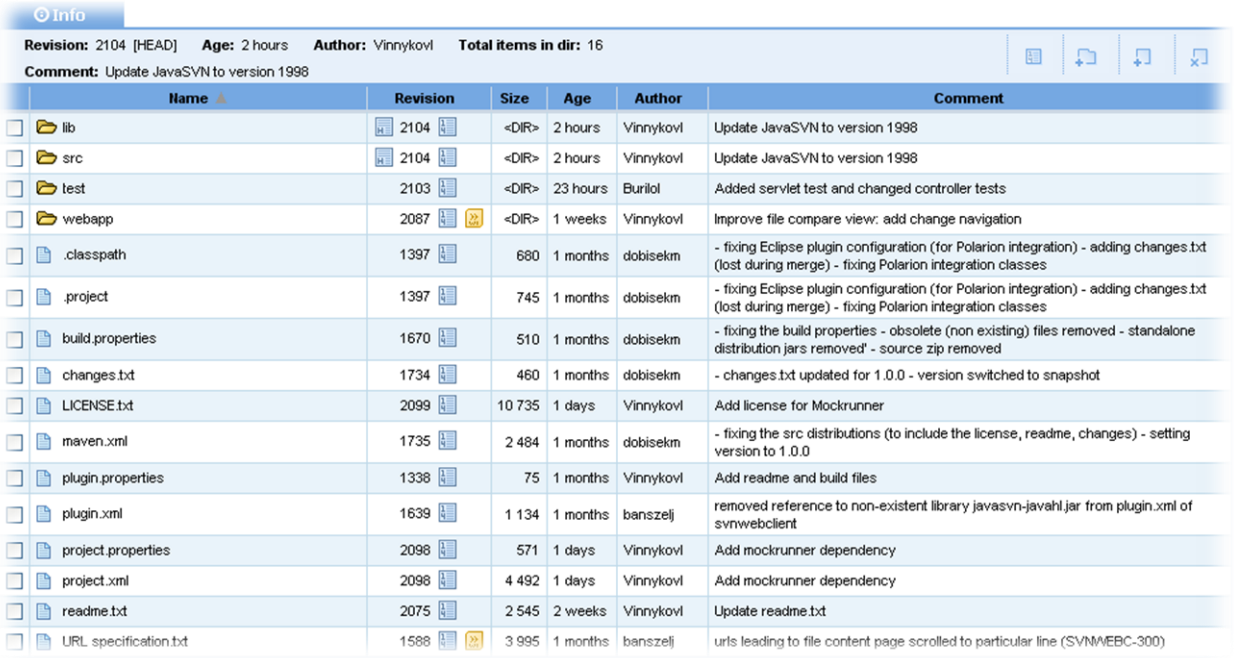
If the *SVN Web Client requires JIRA Session* attribute is set to *false,* then the plug-in does not validate any JIRA session allowing browsing on Subversion to any user logged in Subversion. In this mode, Subversion ALM can be used as a pure Subversion web client and users have not be aware about JIRA. The Subversion Web Client can be accessed from the following URL directly:

*<your JIRA base URL>**/plugins/servlet/svnwebclient/login.jsp***

©Kinto Soft Ltd

Major features

Repository browse



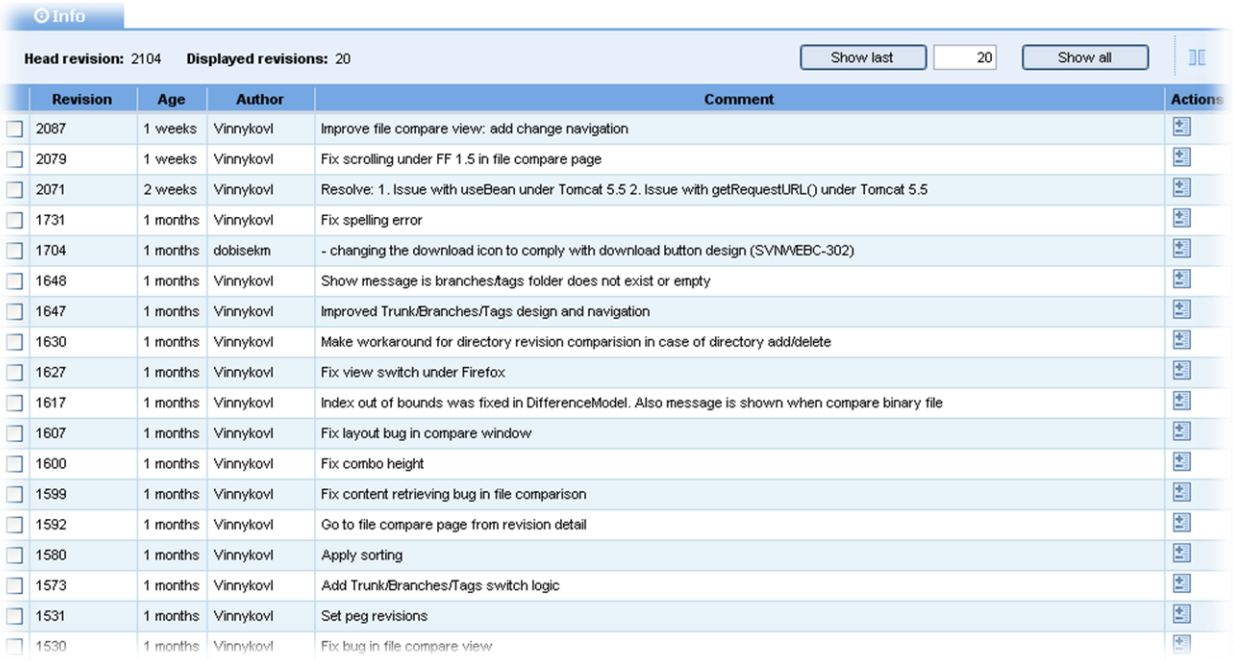
Info

Revision: 2104 [HEAD] **Age:** 2 hours **Author:** Vinnykovl **Total items in dir:** 16

Comment: Update JavaSVN to version 1998

Name	Revision	Size	Age	Author	Comment
lib	2104	<DIR>	2 hours	Vinnykovl	Update JavaSVN to version 1998
src	2104	<DIR>	2 hours	Vinnykovl	Update JavaSVN to version 1998
test	2103	<DIR>	23 hours	Burilol	Added servlet test and changed controller tests
webapp	2087	<DIR>	1 weeks	Vinnykovl	Improve file compare view: add change navigation
.classpath	1397	680	1 months	dobisekm	- fixing Eclipse plugin configuration (for Polarion integration) - adding changes.txt (lost during merge) - fixing Polarion integration classes
.project	1397	745	1 months	dobisekm	- fixing Eclipse plugin configuration (for Polarion integration) - adding changes.txt (lost during merge) - fixing Polarion integration classes
build.properties	1670	510	1 months	dobisekm	- fixing the build properties - obsolete (non existing) files removed - standalone distribution jars removed' - source zip removed
changes.txt	1734	460	1 months	dobisekm	- changes.txt updated for 1.0.0 - version switched to snapshot
LICENSE.txt	2099	10 735	1 days	Vinnykovl	Add license for Mockrunner
maven.xml	1735	2 484	1 months	dobisekm	- fixing the src distributions (to include the license, readme, changes) - setting version to 1.0.0
plugin.properties	1338	75	1 months	Vinnykovl	Add readme and build files
plugin.xml	1639	1 134	1 months	banszelj	removed reference to non-existent library javasvn-javahl.jar from plugin.xml of svnwebclient
project.properties	2098	571	1 days	Vinnykovl	Add mockrunner dependency
project.xml	2098	4 492	1 days	Vinnykovl	Add mockrunner dependency
readme.txt	2075	2 545	2 weeks	Vinnykovl	Update readme.txt
URL specification.txt	1588	3 995	1 months	banszelj	urls leading to file content page scrolled to particular line (SVNWEBC-300)

Revision List



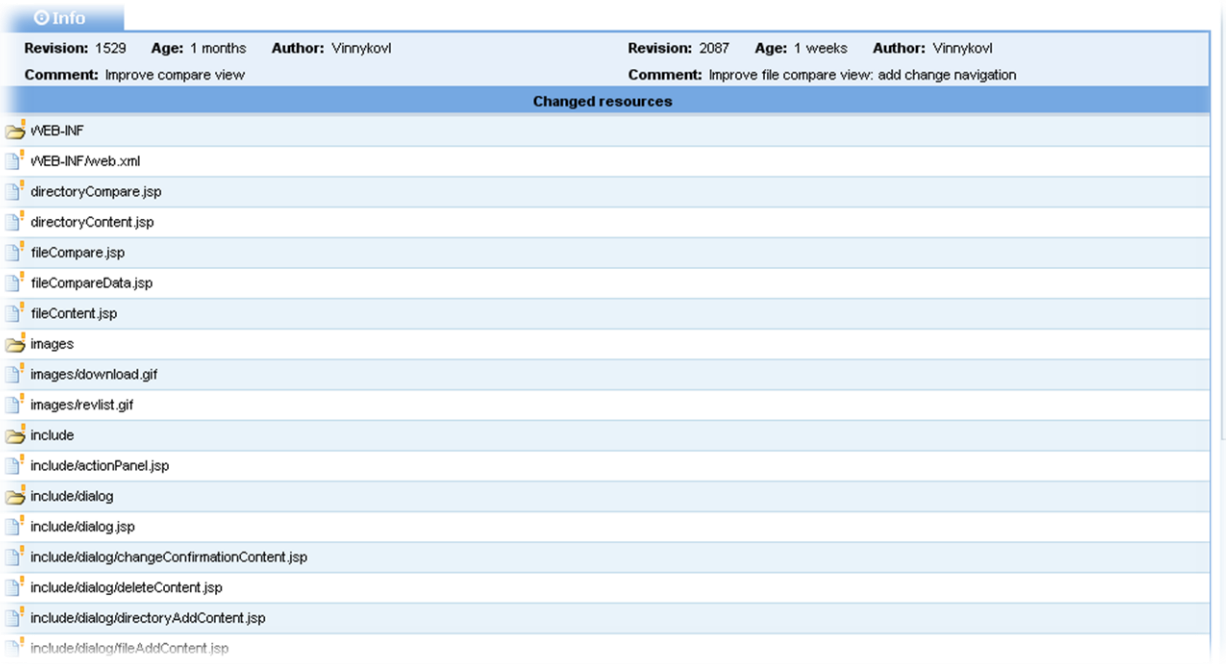
Info

Head revision: 2104 **Displayed revisions:** 20 Show last 20 Show all

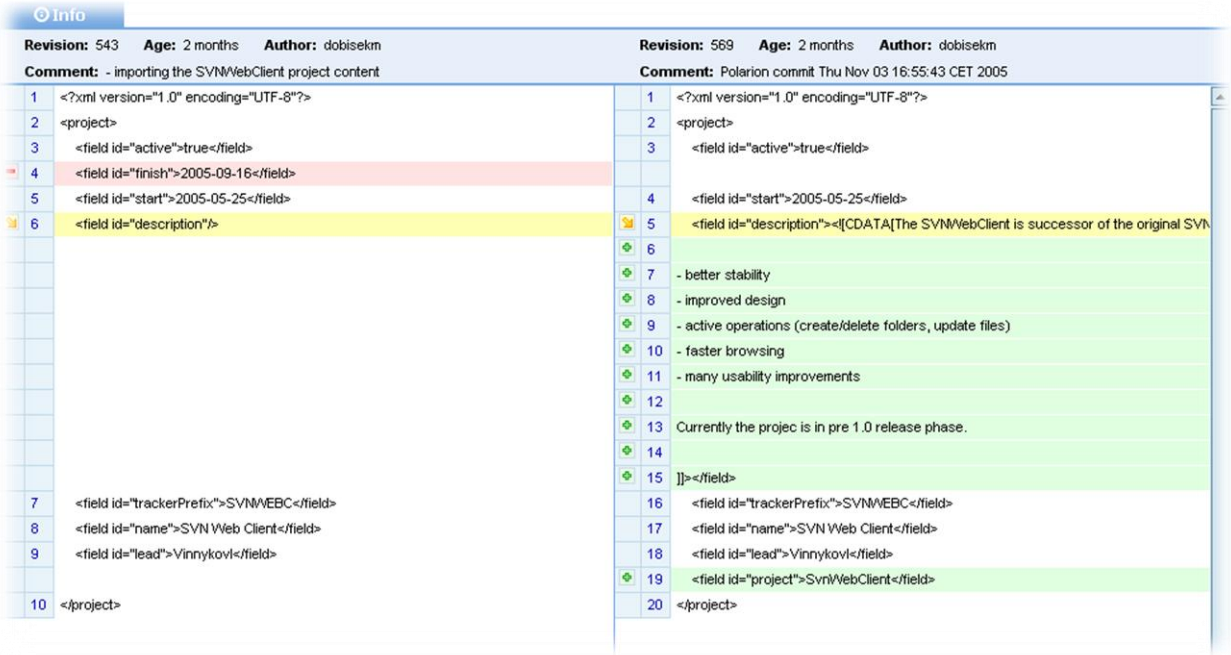
Revision	Age	Author	Comment	Actions
2087	1 weeks	Vinnykovl	Improve file compare view: add change navigation	
2079	1 weeks	Vinnykovl	Fix scrolling under FF 1.5 in file compare page	
2071	2 weeks	Vinnykovl	Resolve: 1. Issue with useBean under Tomcat 5.5 2. Issue with getRequestURL() under Tomcat 5.5	
1731	1 months	Vinnykovl	Fix spelling error	
1704	1 months	dobisekm	- changing the download icon to comply with download button design (SVNWEBC-302)	
1648	1 months	Vinnykovl	Show message is branches/tags folder does not exist or empty	
1647	1 months	Vinnykovl	Improved Trunk/Branches/Tags design and navigation	
1630	1 months	Vinnykovl	Make workaround for directory revision comparision in case of directory add/delete	
1627	1 months	Vinnykovl	Fix view switch under Firefox	
1617	1 months	Vinnykovl	Index out of bounds was fixed in DifferenceModel. Also message is shown when compare binary file	
1607	1 months	Vinnykovl	Fix layout bug in compare window	
1600	1 months	Vinnykovl	Fix combo height	
1599	1 months	Vinnykovl	Fix content retrieving bug in file comparison	
1592	1 months	Vinnykovl	Go to file compare page from revision detail	
1580	1 months	Vinnykovl	Apply sorting	
1573	1 months	Vinnykovl	Add Trunk/Branches/Tags switch logic	
1531	1 months	Vinnykovl	Set peg revisions	
1530	1 months	Vinnykovl	Fix bug in file compare view	

©Kinto Soft Ltd

Revision changes

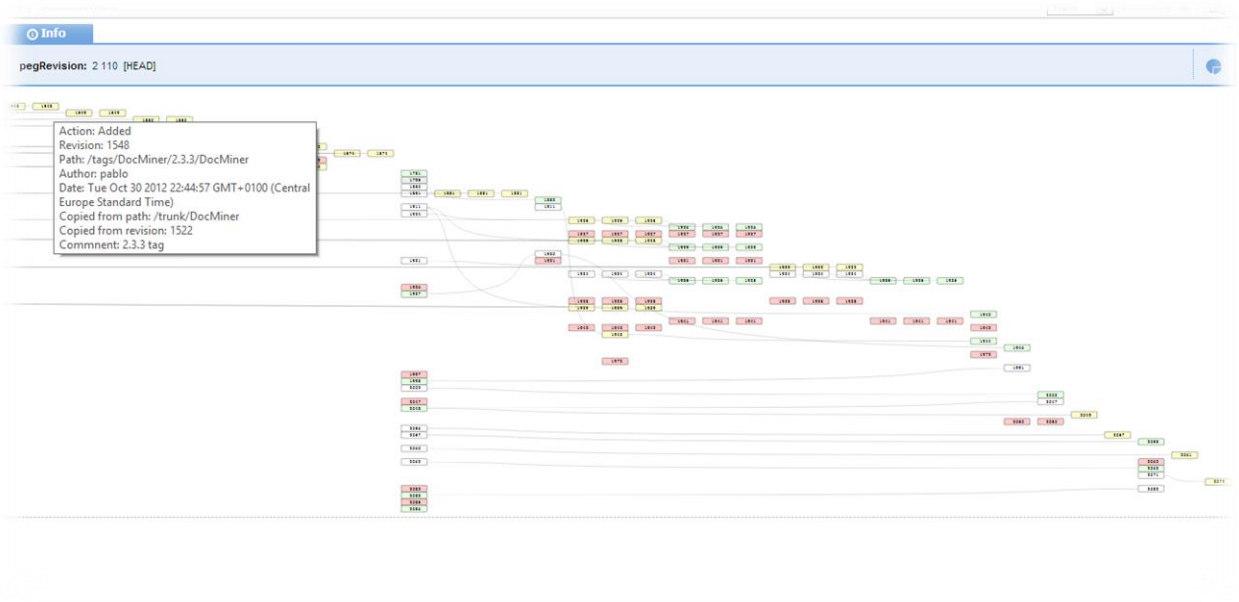


Compare files

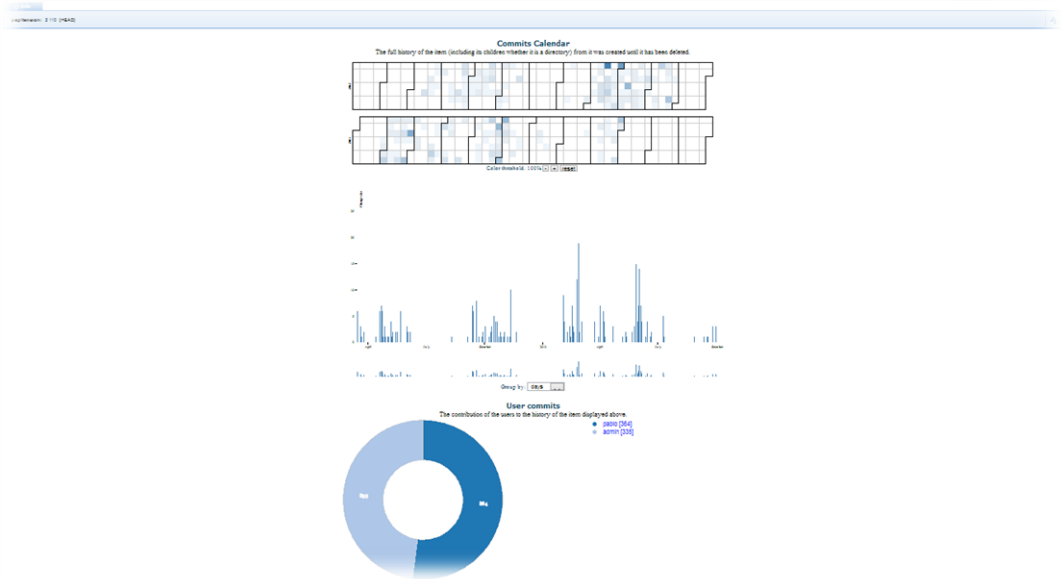


©Kinto Soft Ltd

Commit graphs



Statistics



©Kinto Soft Ltd

Reporting

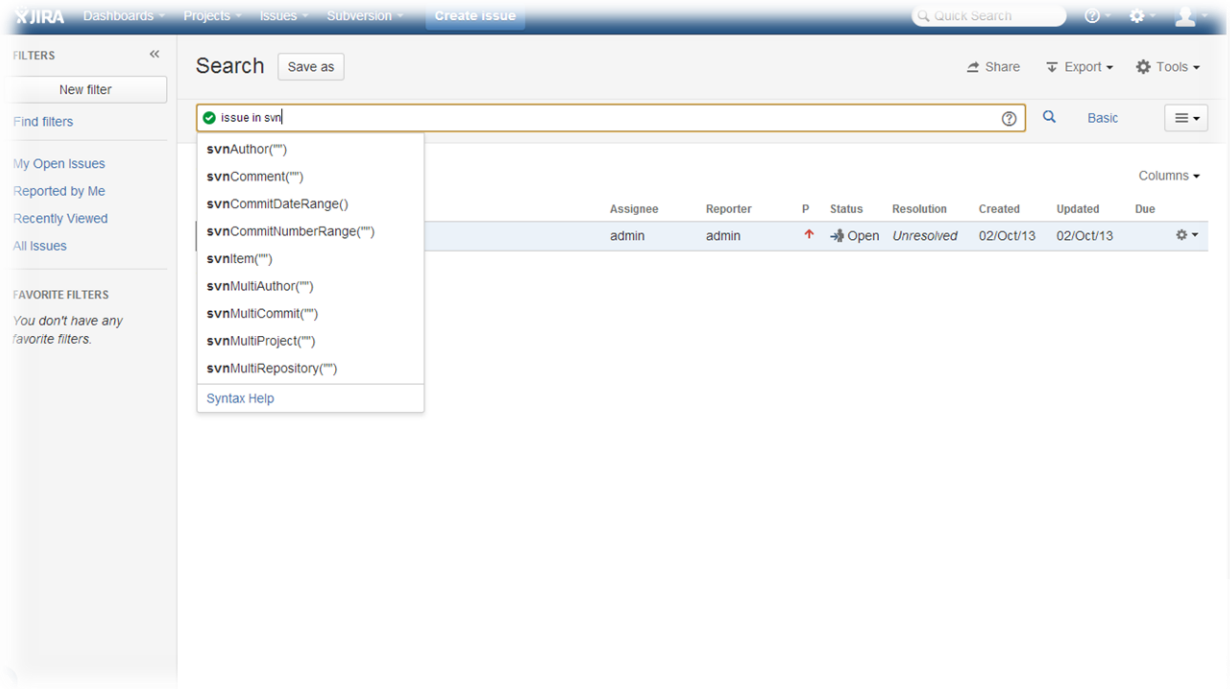
Subversion ALM supports a very powerful and flexible reporting based on:

- JQL
- Commit Calendar View

The JQL functions provided by Subversion ALM allows searching issues filtering by Subversion attributes like author, date range, commit range, words in comment, files and subdirectories, etc. while the Commit Calendar View shows commits in an interactive calendar.

JQL functions

Several JQL functions are supported by Subversion ALM allowing users to search issues on JIRA by using Subversion attributes like the *commit author, commit date range, committed files and directories, any words in the commit message*, etc.



All the Subversion ALM JQL functions:

- Start with the **svn** prefix.
- Show a useful quick guide on JIRA when some parameter has an invalid value.
- Support the following two parameters:

©Kinto Soft Ltd

A) *repoId*: it restricts the search the repository with the provided Id (Id>0). An Id with the zero value always means all the repositories.

B) *limit:* It is very important understanding that the JIRA JQL engine always re-orders the JQL results by some Issue attribute (priority, status, …). Therefore, is not possible displaying on JIRA the issues sorted by any Subversion attribute like the revision number, commit date, etc. To deal with this, when the *limit* filter is set, the results are internally sorted by the *commit day* before they are sent back to JIRA. A limit = 10 means the 10 issues related to the latest (newer) commits matching the query, but they will be re-ordered by JIRA when they are displayed. Regardless they are re-sorted by JIRA you can be confident about they are the most recent issues.

To quick start working with the Subversion ALM JQL functions type the query below in the JIRA Issue Advanced Search:

issue in svn

Currently the Subversion ALM JQL functions are grouped into two main groups:

- *Attribute functions*: they allow searching issues on JIRA by using Subversion attributes:
  - ***svnCommitNumberRange***
  - ***svnCommitDateRange***
  - ***svnAuthor***
  - ***svnComment***
  - ***svnItem***
- *Aggregation functions*: they allow searching issues on JIRA by aggregating (counting/amount) some Subversion attribute:
  - ***svnMultiCommit***
  - ***svnMultiAuthor***
  - ***svnMultiProject***
  - ***svnMultiRepository***

Attribute JQL functions

svnCommitNumberRange					
It returns the issues related to commits between a revision range on an specific repository					
#	**Name**	**Type**	**Required**	**Values**	**Description**
1	repoId	Integer	Yes	>0	The repository Id
2	start	Integer	No	>=0	The start revision of the range
3	end	Integer	No	>= 0	The end revision of the range
4	limit	Integer	No	>=1	Limits the number of returned issues

©Kinto Soft Ltd

Example:

issue in ***svnCommitNumberRange****(5,4364,"",10)*

Meaning:

It returns maximum 10 issues related to the latest commits from the 4364 (included) revision to the HEAD in the repository Id=5.

svnCommitDateRange					
It returns the issues related to commits between a date range on all the repositories					
#	**Name**	**Type**	**Required**	**Values**	**Description**
1	repoId	Integer	Yes	>=0	Zero means all the repositories.
2	start	String	No	yyyy-MM-dd	The start date of the range
3	end	String	No	yyyy-MM-dd	The end date of the range
4	limit	Integer	No	>=1	Limits the number of returned issues

Example:

issue in ***svnCommitDateRange*** *(0,"2013-02-01","",99)*

It returns maximum 99 issues related to the latest commits on any repository from the February, 1st 2013.

svnAuthor					
It returns the issues related to commits made for an specific user					
#	**Name**	**Type**	**Required**	**Values**	**Description**
1	username	String	No	Any	An empty string means no username (null).
2	repoId	Integer	No	>=0	Zero means all the repositories.
3	limit	Integer	No	>=1	Limits the number of returned issues

Example:

issue in ***svnAuthor*** *("sally",0,1)*

It returns one issue (if any) related to the latest Sally's commit on any repository.

©Kinto Soft Ltd

svnComment					
It returns issues related to commits containing all the exact words in their commit comment messages					
#	**Name**	**Type**	**Required**	**Values**	**Description**
1	Lucene Query	String	Yes	Any	A Lucene Query
2	repoId	Integer	No	>=0	Zero means all the repositories.
3	limit	Integer	No	>=1	Limits the number of returned issues

Example:

issue in ***svnComment*** *(”merge* AND r1034”, 3)*

It returns all the issues related to any commit (without any specific order) matching the Lucene query for the repository Id=3. In this example “merg**ed** in revision r1034” would match the query

svnItem					
It returns the issues related to commits modifying a file or subdirectory					
#	**Name**	**Type**	**Required**	**Values**	
1	Item	String	Yes	Path	A path or file name
2	Action	String	No	A,M,D,R,””	The action on the item
3	repoId	Integer	No	>=0	Zero means all the repositories.
4	limit	Integer	No	>=1	Limits the number of returned issues

Examples:

1) *issue in **svnItem**(“myfile.txt”,”D”,3)*
2) *issue in **svnItem**(“/foo/myfile.txt”,””,3,5)*
3) *issue in **svnItem**(“/foo”,”A”,3)*
4) *issue in **svnItem**(“/foo/”,””,3,10)*

1) It returns all the issues related to the commit which deleted a “myfile.txt” file in the repository 3 regardless the file location. As no path is set, it will search for the file in all the branches.
2) It returns the issues related to the latest 5 commits changing the exact “/foo/myfile.text” file in the repository 3. As the path is set to “/foo” it will look for the file in that exact branch.

©Kinto Soft Ltd

3) It returns all the issues related to the commit which created the exact “/foo” branch. Notice that the branch name does not end with a slash “/” character, so the subdirectories are ignored.
4) It returns 10 issues related to all the commits modifying something under the “/foo” branch. Notice that the branch name ends with a slash “/”, hence all the subdirectories will be taken in consideration during the search.

©Kinto Soft Ltd

Aggregation JQL functions

Sometimes managers want to track some potential risks. Subversion ALM supports some useful JQL functions in order to help them:

svnMultiCommit					
It returns issues related to a minimal amount commits					
#	**Name**	**Type**	**Required**	**Values**	
1	count	Integer	Yes	> 0	Minimal amount of commits
2	limit	Integer	No	>=1	Limits the number of returned issues

Example:

issue in ***svnMultiCommit*** *(5, 10)*

What are the most recent 10 issues which have been related to 5 or more commits on any repository?

svnMultiAuthor					
It returns issues with commits from several authors					
#	**Name**	**Type**	**Required**	**Values**	
1	count	Integer	Yes	> 0	Minimal amount of different authors
2	limit	Integer	No	>=1	Limits the number of returned issues

Example:

issue in ***svnMultiAuthor*** *(3, 10)*

What are the 10 most recent issues which have commits from more than 3 different users?

svnMultiRepository					
It returns issues with commits on different repositories					
#	**Name**	**Type**	**Required**	**Values**	
1	count	Integer	Yes	> 0	Minimal amount of different repositories
2	limit	Integer	No	>=1	Limits the number of returned issues

©Kinto Soft Ltd

Example:

issue in ***svnMultiRepository*** *(2)*

What are all the issues having commits on 2 or more repositories? (Cross repository issues are not allowed in this company).

svnMultiProject					
It returns issues with commits affecting to different JIRA projects					
#	**Name**	**Type**	**Required**	**Values**	
1	commits	Integer	Yes	> 0	A path or file name
2	limit	Integer	No	>=1	Limits the number of returned issues

Example:

issue in ***svnMultiProject*** *(2, 10)*

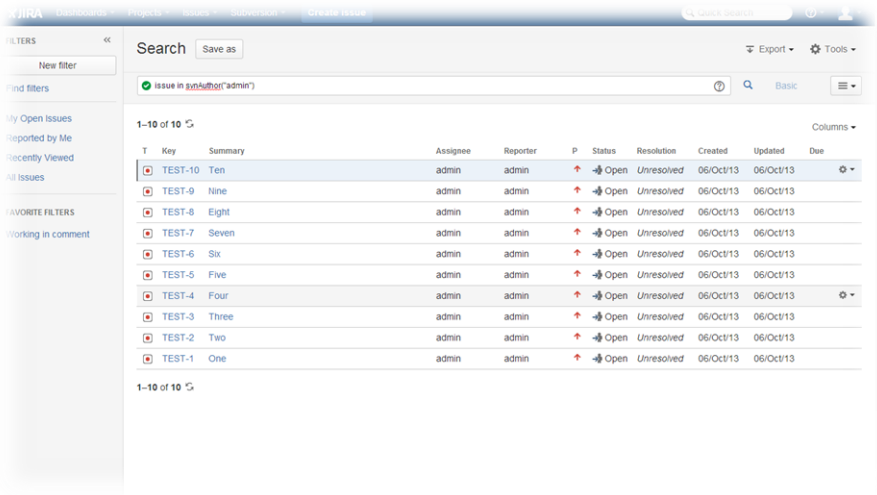
What are the recent issues that have commits on 2 on more JIRA projects?

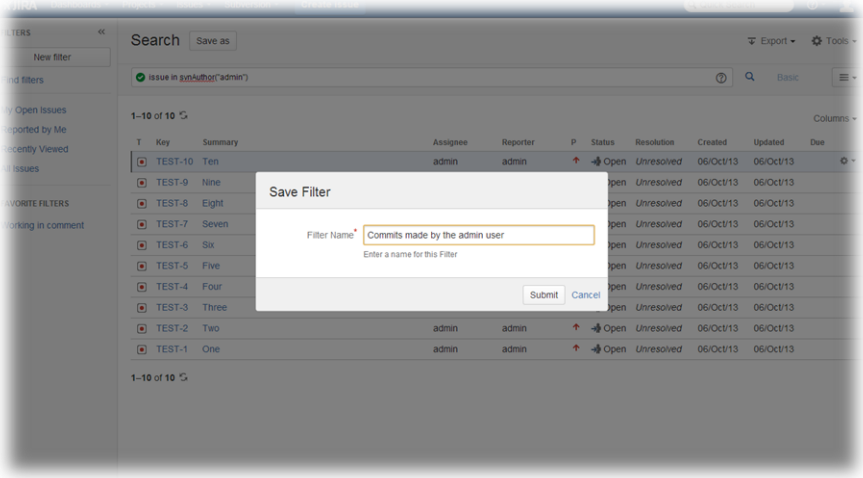
©Kinto Soft Ltd

Commits Calendar Report

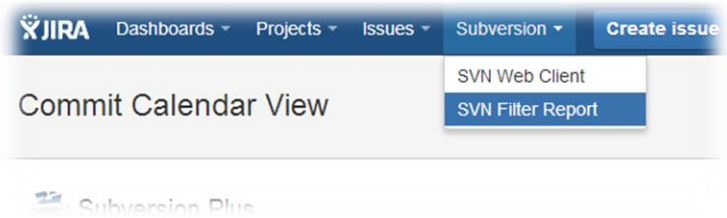
The Commits Calendar Report allows visualizing the commits related to the issues returned by any filter.

Prior to use it, a JQL query has to be saved:

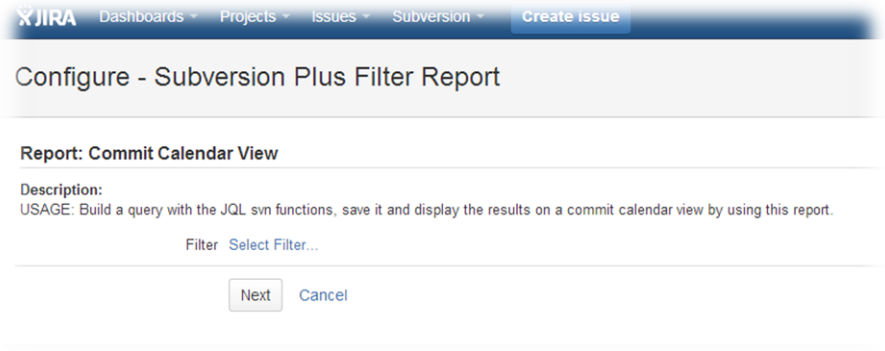




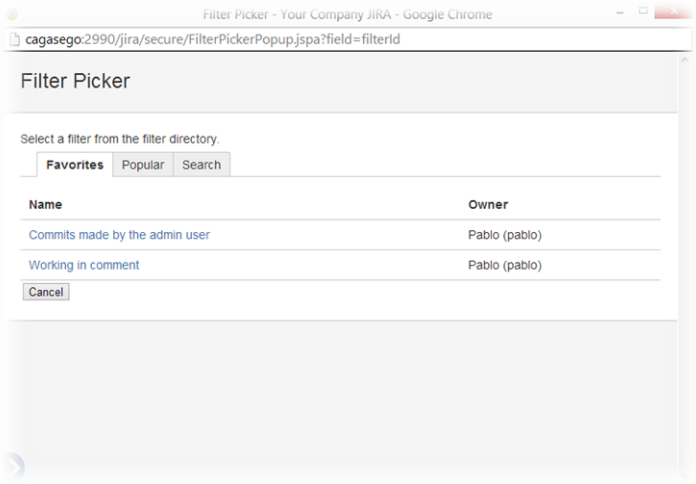
From the top menu: *Subversion* > *SVN Filter Report*



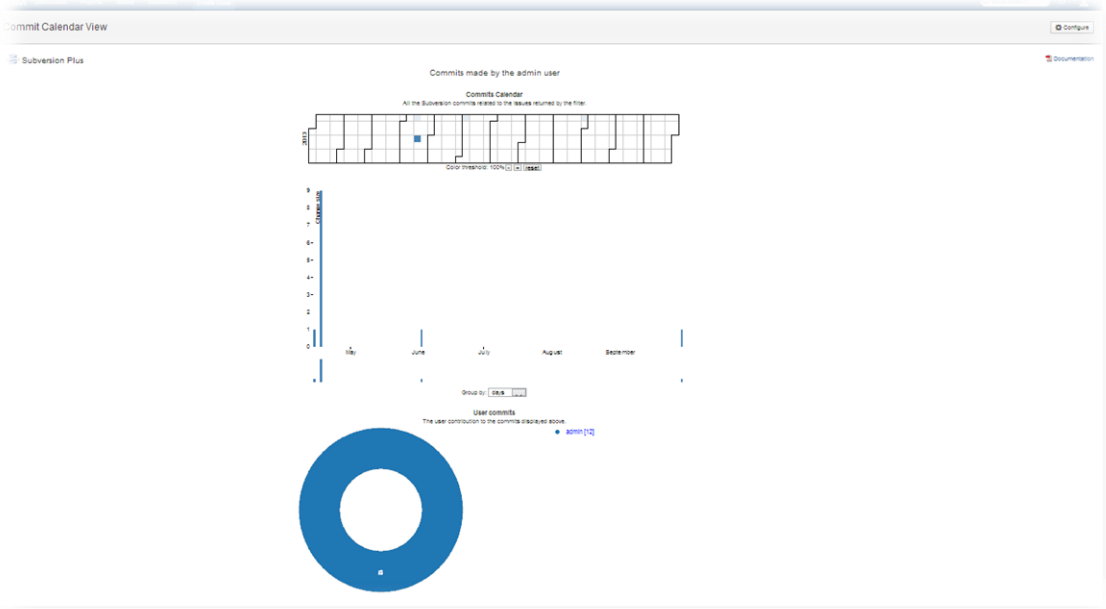
©Kinto Soft Ltd



Click on the *Select Filter…* link:



Select a filter and click on the *Next* button:

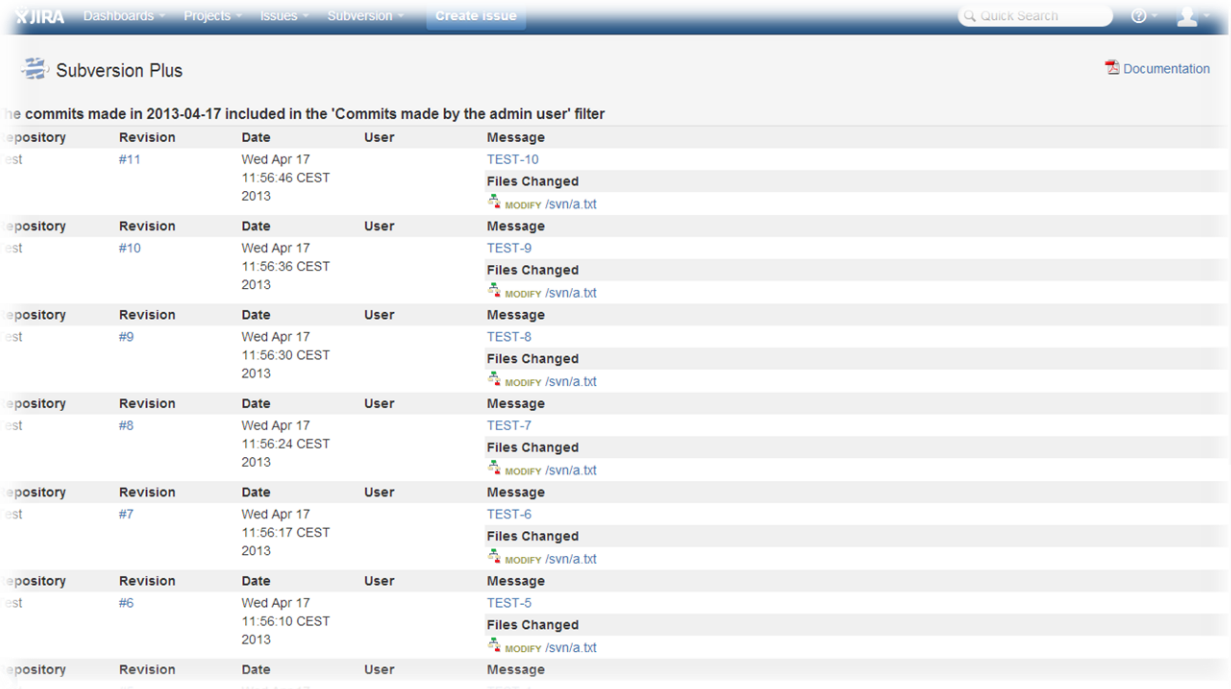


©Kinto Soft Ltd

This page shows:

- a *calendar* at the top of the page displaying the commits related to the JIRA issues along the time
- a *bar chart* at the middle showing the amount of commits vs time. By default the commits are displayed grouped by day, although they can be re-grouped in weeks: there is an interactive thumbnail view below allowing to select a range of days.
- a *pie chart* at the bottom with the contribution of each author.

You can click on any day on the top calendar as well as on the authors' names links in the pie chart to see the issues on JIRA:



©Kinto Soft Ltd

Extending the functionality with 3rd party plugins

Since the 5.5 version, sharing the database connection among 3rd party plug-ins is supported. And since the 5.8 version it publishes commit events allowing subscription.

How to use the JDBC connections from a 3rd party plug-in

This brings a lot of power to users and allows them to enhance the current functionality by creating custom plug-ins in order to:

- Writing a GUI supporting to change JIRA issue keys related to commits when users forget or type a wrong key related to a commit
- Capture JIRA moved issue events to keep aligned the commits
- Build powerful reports
- ...

1. Add a copy of the "SWCPublicInterface" Java interface into your sources plugin:

com.kintosoft.svnwebclient.jira.public_interface.SWCPublicInterface

the interface above provides a "getConnection()" method returning an standard JDBC connection

2. Import the implementation of the interface provided by the Subversion ALM plugin

by adding the component-import element below into your atlassian-plugin.xml descriptor:

```
<component-import key="svnalm-public">
        <interface>
        com.kintosoft.svnwebclient.jira.public_interface.SWCPublicInterface
        </interface>
</component-import>
```

3. Import the "com.kintosoft.svnwebclient.jira.public_interface" from your Maven *pom.xml* file to make it visible from your plugin:

```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-
v4_0_0.xsd">

    <build>

        <plugins>
```

©Kinto Soft Ltd

```
            <plugin>
                <configuration>
                    <instructions>
                        <Import-Package>
com.kintosoft.svnwebclient.jira.public_interface*;version="0.0.0"
                        </Import-Package>
                        ....
```

4. Use injection in the constructor to get an instance of the:

```
public class  Foo{
       final private SWCPublicInterface swci = null;
       public Foo(SWCPublicInterface swci){
              this.swci = swci;
       }
}
```

4. Get a JDBC connection and do not forget make a commit/rollback and CLOSE THE CONNECTION in order to return it to the pool:

```
Connection conn = null;
PreparedStatement ps = null;
try {
       try {
              int commits = 0;
              conn = swc.getConnection();
              String sql = "select count(*) as count from KEYS where project=?";
              ps = conn.prepareStatement(sql);
              ps.setString(1, key);
              ResultSet rs = ps.executeQuery();
              if (rs.next()) {
```

```
                commits = rs.getInt("count");
            }
            rs.close();
            startingParams.put("commits", commits);
        } finally {
            if (ps != null) {
                ps.close();
            }
            if (conn != null) {
                conn.rollback(); // rollback or commit accordingly
                conn.close(); // return the connection to the pool
            }
        }
} catch (SQLException ex) {
        System.out.println(ex.getMessage());
}
```

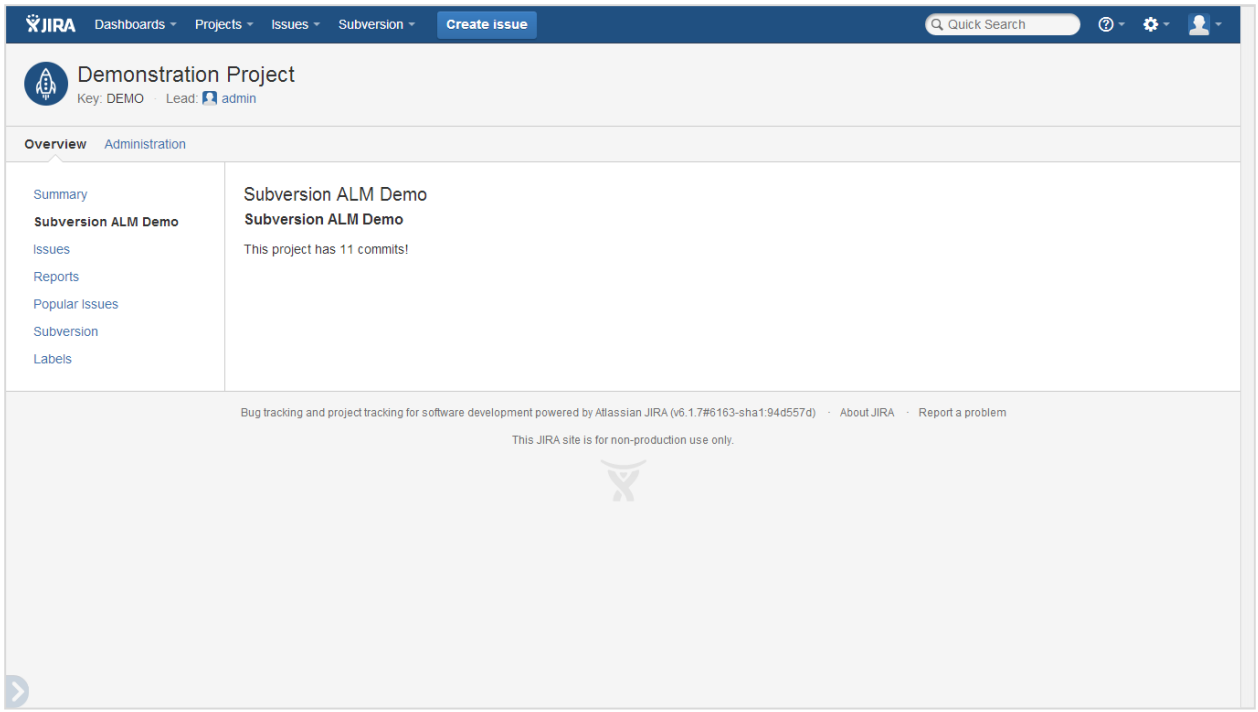
Example

There is available an open source public example available on Google Code released under the Apache license:

https://code.google.com/p/jira-svn-alm-db-demo/

This example creates a *Subversion ALM Demo* tab on JIRA projects and shows the amount of commits:

©Kinto Soft Ltd

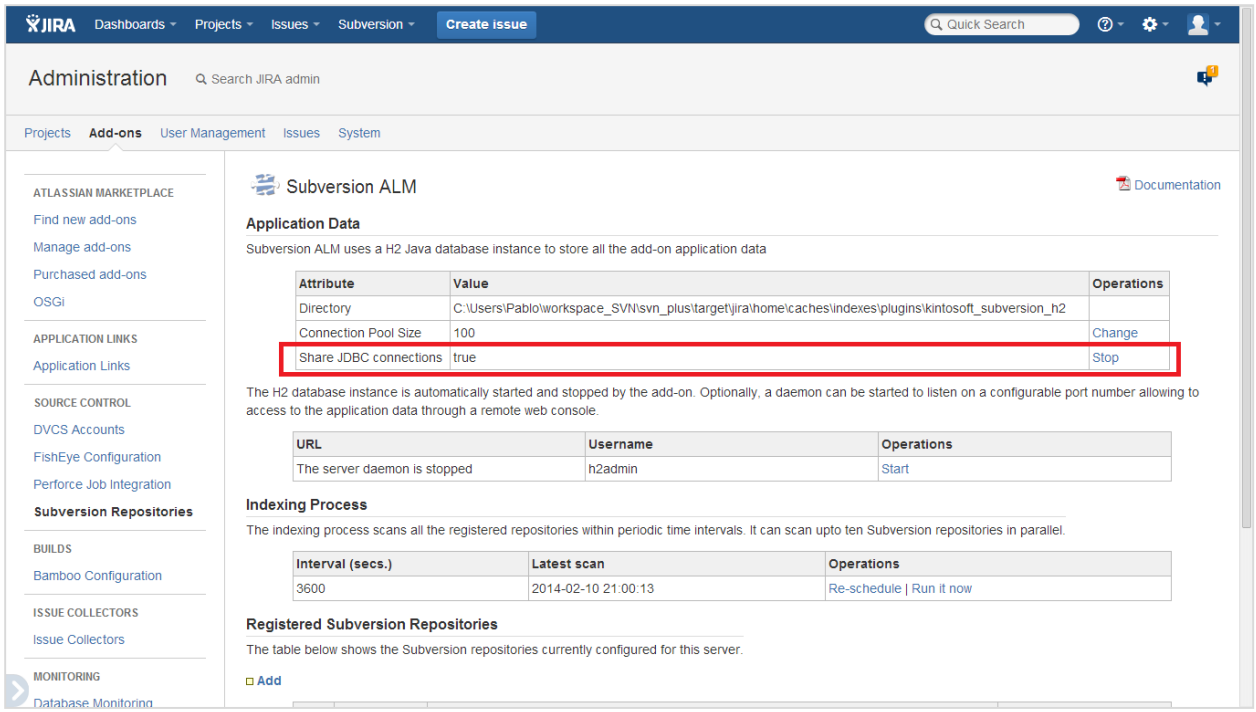


Security

Exposing the JDBC connection brings potential security risks as a malicious 3rd party plug-in could gain access to the Subversion ALM database and get the passwords of all the registered Subversion repositories!

This would be pretty difficult as all the plugins on the Marketplace are validated by Atlassian. However, to bring some peace of mind to JIRA administrators, a new feature was introduced in the 5.5.1 version: Sharing the JDBC connection must be explicitly enabled by a JIRA administrator:

©Kinto Soft Ltd



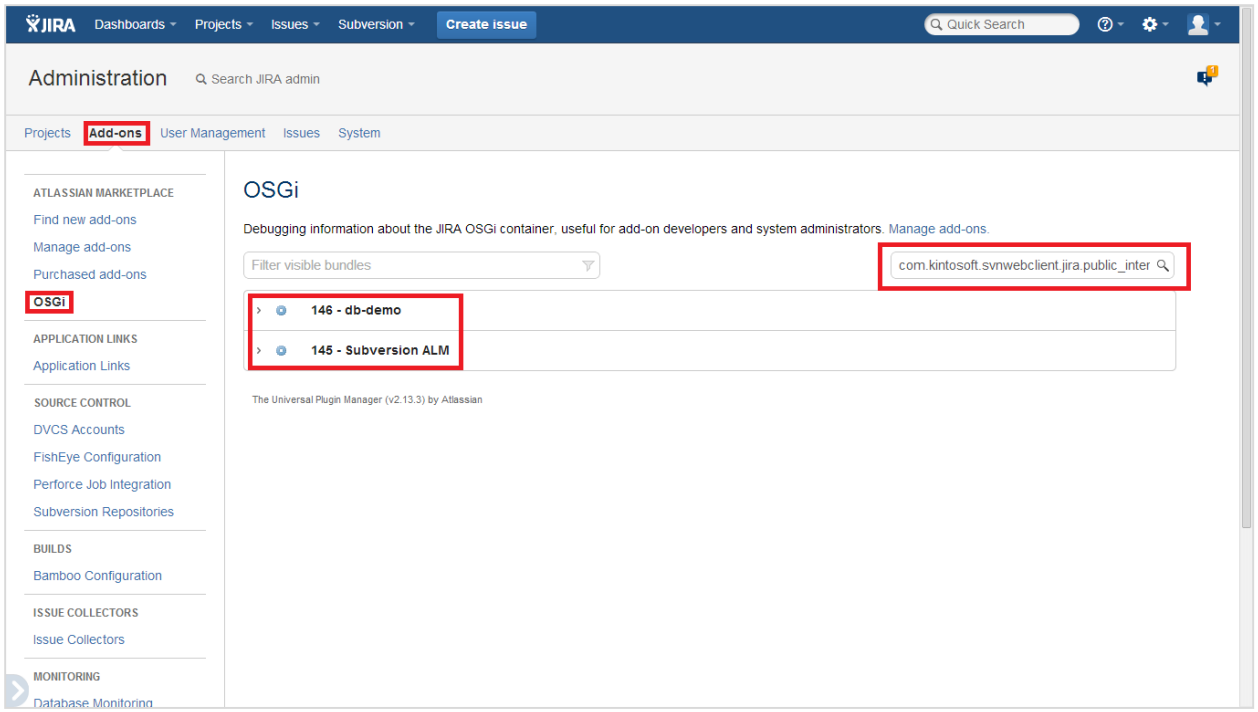
Before sharing any connection, the Subversion ALM plug-in checks the *Share JDBC connection* value. If it is false, then an exception is raised aborting the process and nobody can get a JDBC connection.

Audit

How to know what plug-ins are accessing to the Subversion ALM internal database?

All the plug-ins have to explicitly import the `com.kintosoft.svnwebclient.jira.public_interface` Java package in order to access to the Java object serving JDBC connections. Fortunately, JIRA tracks such information and allows administrators to check who is accessing to those objects from the OSGi console:

©Kinto Soft Ltd



Copy the package name and paste it into the box at the top right. This will unveil the add-ons using the Subversion JDBC connection ☺

Subscribing to Commit Events

Commits Events is a powerful feature supported from the 5.7.0 version. It allows to 3rd party plug-ins listen to all the commits fetched from Subversion and extend the functionality to create review issues or Smart Commits among other features.

In order to implement your own solution you have to follow the steps below:

1. Add a dependency into your pom.xml for the Google's Guava jar library. This library is provided by the JIRA System plugin and (among other features)

it provides the Eventbus component for event publish/subscription:

```
<dependency>
        <groupId>com.google.guava</groupId>
        <artifactId>guava</artifactId>
        <version>10.0.1</version>
        <scope>provided</scope>
</dependency>
```

©Kinto Soft Ltd

2. Import the *svnalm-commits-eventbus* component shared by Subversion ALM.

```
<component-import key="svnalm-commits-eventbus">
  <interface>com.kintosoft.svnwebclient.jira.public_interface.CommitsEventbusService
  </interface>
</component-import>
```

3. Add a copy of the *com.kintosoft.svnwebclient.jira.public_interface.CommitEvent* class into your plug-in. Subversion ALM creates and dispatches a new *CommitEvent* object for each commit fetched from Subversion. This class supports the repositoryId, the commit number (revision) and the timestamp with the milliseconds when the *CommitEvent* was fired.

4. Create your own listener (subscriber) class. Have a look at the *com.kintosoft.jira.svn.MyPluginComponentImpl.CommitEventListener* class for an example.

Accordingly to the Eventbus specification, you have to add a method with the *@Subscribe* annotation for *CommitEvents* handling:

```
@Subscribe
public void handler(CommitEvent event) {
        log.info(event.getRevision() + " revision commited in repository "
                        + event.getRepoId());
}
```

5. Add copy of the com.*kintosoft.svnwebclient.jira.public_interface.CommitsEventbusService* interface into your project. This interface allows to (un) subscribe (from) to the Eventbus provided by Subversion ALM. There is an example available in the com.kintosoft.jira.svn.MyPluginComponentImpl class. The *CommitsEventbusService* can be injected in the constructor:

```
public void register() {
        eventBus.startListenting(commitListener);
        log.info("MyPluginComponentImpl commit listener registered");
}
```

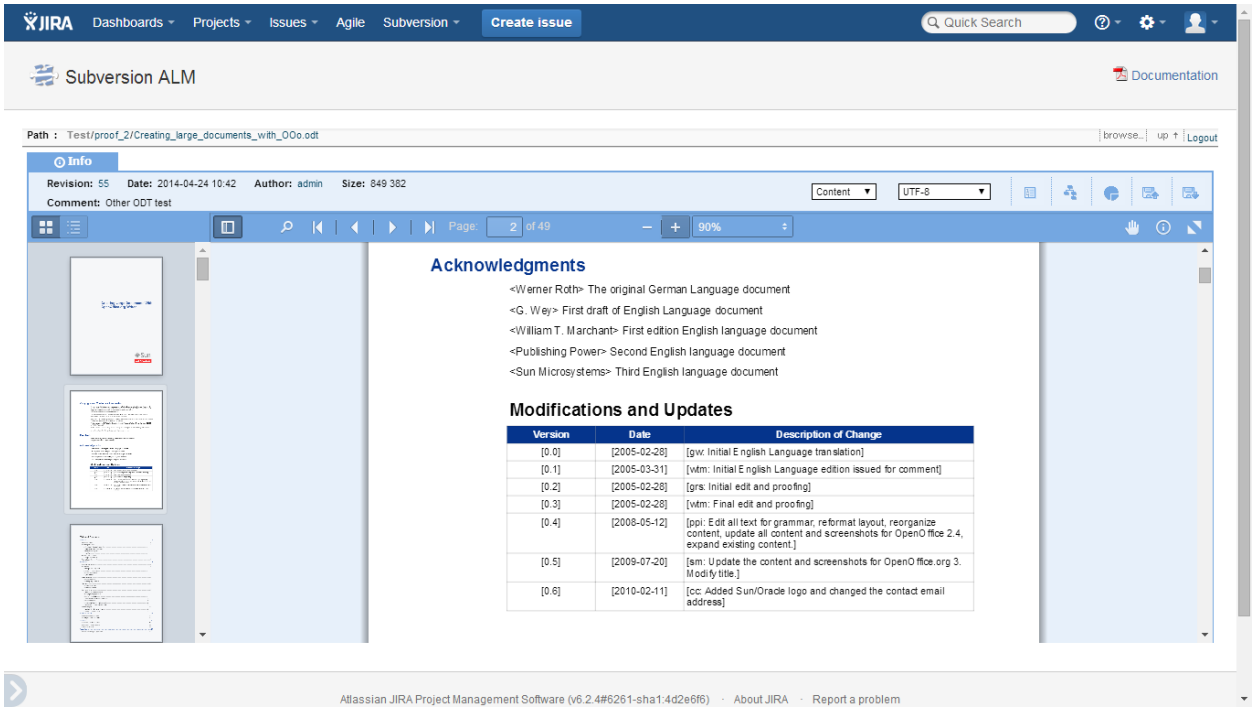
©Kinto Soft Ltd

```
@PreDestroy
public void unregister() {
        log.info("MyPluginComponentImpl destroying");
        eventBus.stopListenting(commitListener);
        log.info("MyPluginComponentImpl commit listener unregistered");
}
```

©Kinto Soft Ltd

Office and PDF documents

Since the 6.0 version, Microsoft Office and OpenOffice documents are supported. In the 6.0 version they will be displayed by using an internal viewer. In further versions new features like version comparison will be supported:



Documents are transformed into PDF files and stored in the plug-in cache *kintosoft_subversion_doc* under the JIRA HOME directory for performance.

©Kinto Soft Ltd